Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Факультет комп'ютерних наук

Спеціальність 125 «Кібербезпека»

Освітня програма «Безпека інформаційних та комунікаційних систем»

«Допущено до захисту»

Зав.кафедрою БІСТ

Сергій РАССОМАХІН

_____

« » 2022 р.

Пояснювальна записка

до кваліфікаційної роботи магістра

на тему: «Розробка та обґрунтування методики пошуку вразливостей у web-застосунках»

оцінка « » Керівник проф. Олійников Р. В.

Голова ЕК Рецензент Шумов О. І.

Доценко С.І. _____ Виконавець : студент групи КБ-61

Лахтін І. А

_____

Харків – 2022

РЕФЕРАТ

Кваліфікаційна робота магістра обсягом 76 сторінок, містить 1 рисунок, 2 таблиці, 39 літературних посилань та 2 додатки.

Мета роботи: розробка нової методики виявлення вразливостей у веб-застосунках, яка має покращити можливості для аудиту веб-застосунків за рахунок зменшення кількості помилкових спрацьовувань та помилок пов’язаних з людським фактором.

Методи дослідження: у цій роботі проведено огляд літератури в напрямку виявлення нових векторів атак, вразливостей, механізму виявлення, прогалин в дослідженнях, проведено порівняльний аналіз існуючих підходів до виявлення вразливостей веб-додатків, та на основі отриманих результатів розроблена нова методика, яка матиме меншу ймовірність помилкових спрацьовувань.

У роботі розробляється методика виявлення вразливостей веб-застосунків на основі існуючих методик та стандартів в цій галузі. Також в роботі наводиться приклад використання запропонованої методики.

Практичне значення роботи полягає у розробці методики, для пошуку вразливостей веб-застосунків.

Результати здійснених у кваліфікаційній роботі досліджень можуть бути використані при тестуванні веб-додатків.

Ключові слова: ВРАЗЛИВІСТЬ, ТЕСТУВАННЯ, ВЕБ-ЗАСТОСУНОК, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ.

ABSTRACT

The master's thesis is 76 pages long and contains 1 illustration, 2 tables, 39 literary references and 2 appendices.

Objective: to develop a new method for detecting vulnerabilities in web applications, which should improve the capabilities for auditing web applications by reducing the number of false positives and errors related to the human factor.

Research methods: the literature was reviewed in the direction of identifying new attack vectors, vulnerabilities, detection mechanism, research gaps, a comparative analysis of existing approaches to detecting Web Application Vulnerabilities was carried out, and based on the results obtained, a new method was developed that will have a lower probability of false positives.

The paper shows the development of a method for detecting vulnerabilities in web applications based on existing methods and standards in this area. The paper also provides an example of using the proposed methodology.

The practical significance of the work lies in the development of a methodology that can be used when searching for vulnerabilities in web applications.

The results of the research carried out in the thesis can be used in testing web applications.

Keywords: VULNERABILITY, TESTING, WEB APPLICATION, SOFTWARE, VULNERABILITY DETECTION.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 8

ВСТУП 10

1 СПОСОБИ ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ У ВЕБ-ЗАСТОСУНКАХ 12

1.1 Сканування вразливостей 12

1.2 Проведення тестування на проникнення 14

1.3 Постановка завдань досліджень 17

2 АНАЛІЗ СТАНДАРТІВ І МЕТОДИК В ОБЛАСТІ ПОШУКУ ВРАЗЛИВОСТЕЙ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 19

2.1 Базова термінологія і класифікація тестових впливів 19

2.2 Методика OSSTMM 21

2.3 Методика ISSAF 23

2.4 Методика OWASP 28

2.5 Стандарт PTES 29

2.6 Стандарт NIST SP 800-115 32

2.7 Методика BSI 34

2.8 Методика PETA 36

2.9 Методика PTF 37

2.10 Результати порівняльного аналізу стандартів і методик 38

3 РОЗРОБКА МЕТОДИКИ ПОШУКУ ВРАЗЛИВОСТЕЙ У WEB-ЗАСТОСУНКАХ 42

3.1 Планування 44

3.1.1 Визначення області досліджень 44

3.1.2 Екстрені контакти 45
3.1.3 Правила взаємодії 45
3.1.4 Визначення показників якості 45
3.1.5 Оцінка термінів 46
3.1.6 Вимоги безпеки 46
3.1.7 Визначення інструментів 46
3.2 Початковий збір інформації 47
3.2.1 Ідентифікація цілей 47
3.2.2 Підхід 47
3.2.2.1 Пасивний підхід 47
3.2.2.2 Напівпасивний підхід 47
3.2.2.3 Активний підхід 47
3.3 Сканування 48
3.3.1 Активне сканування 48
3.3.1.1 Сканування мережі 48
3.3.1.2 Сканування додатка 48
3.3.1.3 Ідентифікація вразливостей 49
3.3.2 Пасивне сканування 49
3.3.2.1 Моніторинг трафіку 49
3.3.2.2 Аналіз метаданих 49
3.3.3 Тестування 50
3.3.4 Перевірка коду 50
3.3.4.1 Аналіз вихідного коду 51
3.3.4.2 Аналіз запитів 51
3.3.5 Підтвердження результатів 51

3.4 Аналіз вразливостей 52
3.4.1 Аналіз загроз 52
3.4.2 Можливість експлуатації вразливостей 52
3.4.3 Аналіз доступності 53
3.4.4 Планування атак 53
3.5 Використання вразливостей 53
3.5.1 Експлойти 53
3.5.1.1 Загальнодоступні експлойти 54
3.5.2 Подальше проникнення до системи 55
3.5.2.1 Нові вразливості 55
3.5.3 Встановлення бекдорів 56
3.5.4 Очищення 56
3.6 Аналіз результатів 56
3.6.1 Огляд отриманих результатів 56
3.6.1.1 Огляд правил взаємодії 57
3.6.1.2 Огляд цілей 57
3.6.1.3 Огляд показників якості 57
3.6.2 Аналіз впливу тестування на систему 57
3.6.3 Аналіз тесту на проникнення 58
3.7 Звітність 58
3.7.1 Загальний звіт 58
3.7.2 Технічний звіт 59
3.7.2.1 Висновки 59
3.7.3 Додаткова інформація 60
3.8 Висновок 60

4 ПРИКЛАД ВИКОРИСТАННЯ РОЗРОБЛЕНОЇ МЕТОДИКИ 64
4.1 Середовище тестування 64
4.2 Відомості про додаток 64
4.3 Інструменти 65
4.3.1 Nmap 66
4.3.2 Metasploit 66
4.3.3 Burp Suite Community Edition 67
4.3.4 OWASP ZAP 68
4.3.5 Wireshark 68
4.3.6 WebInspect 69
4.4 Використовувана методика 69
4.5 Тестовий приклад - DVWA 71
4.5.1 Планування 71
4.5.2 Початковий збір інформації 72
4.5.3 Сканування 73
4.5.4 Аналіз вразливостей 74
4.5.5 Використання вразливостей 76
4.5.6 Аналіз результатів 78
4.5.7 Звітність 78
4.6 Підсумок тестового прикладу 78
ВИСНОВКИ 79
ПЕРЕЛІК ПОСИЛАНЬ 81
ДОДАТОК А 86
ДОДАТОК Б 96

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – База даних

ІБ – Інформаційна безпека

ІПВ – Інформаційно-психологічний вплив

ІТВ – Інформаційно-технічний вплив

НСД – Несанкціонований доступ

ОС – Операційна система

ПЗ – Програмне забезпечення

BSI – Bundesamt für Sicherheit in der Informationstechnik

BSQLi – Blind Structured Query Language Injektion

CSRC – Computer Security Resource Center

CVE – Common Vulnerabilities and Exposures

CVSS – Common Vulnerability Scoring System

DNS – Domain Name System

DoS – Denial of Service

DVWA – Damn Vulnerable Web Application

HTTP – HyperText Transfer Protocol

ISECOM – Institute for Security and open Methodologies

ISSAF – Information System Security Assessment Framework

IT – Information Technology

NIST	−	National Institute of Standards and Technology
NIST SP	−	National Institute of Standards and Technology Special Publication
NVD	−	National Vulnerability Database
OISSG	−	Open Information Systems Security Group
OSINT	−	Open source intelligence
OSSTMM	−	The Open Source Security Testing Methodology Manual
OWASP	−	Open Web Application Security Project
PETA	−	Methodology of Information Systems Security Penetration Testing
PTES	−	Penetration Testing Execution Standard
PTF	−	Penetration Testing Framework
SQL	−	Structured query language
URL	−	Uniform Resource Locator
VoIP	−	Voice over IP
VPN	−	Virtual Private Network
WASSEC	−	Web Application Security Scanner Evaluation Criteria
WEP	−	Wired Equivalent Privacy
WPA	−	Wi-Fi Protected Access

ВСТУП

В даний час управління інформаційною безпекою починає ставати пріоритетом для більшості компаній. Основна мета – запобігти несанкціонованому доступу та використанню секретної інформації проти організації. Успішні кібератаки можуть призвести до значних втрат щодо секретної інформації, прямих економічних втрат і репутаційного збитку. Необхідність визначення пріоритетів уразливості в організаціях широко визнається і систематично зростає; таким чином, в останні роки з'явилися нові способи звітування та оцінки цих вразливостей.

Наше суспільство більш технологічно залежне, ніж будь-коли раніше, але кіберзлочинність також зростає і стає все більш досконалою. Cybersecurity Ventures очікує, що глобальні витрати на збитки від кіберзлочинності зростуть на 15 відсотків на рік протягом наступних п'яти років, досягнувши до 2025 року 10,5 трлн доларів США на рік порівняно з 3 трлн доларів США в 2015 році [1]. Порушення даних постійно збільшуються, надаючи зловмисникам доступ до конфіденційних даних, інформації про особу, захищеної медичної інформації, особистої інформації, інтелектуальної власності, даних державних та галузевих інформаційних систем.

На жаль, сучасного підходу до безпеки може бути недостатньо. Якщо організація дійсно хоче суттєво знизити ризики, відділу IT безпеки необхідно виявляти всі типи вразливостей шляхом систематичного та активного тестування системи безпеки на наявність вразливостей. Це можна зробити за допомогою різних методологій, таких як тестування на проникнення або аналіз вразливостей.

Безпека веб-додатків пов'язана з атаками безпеки на рівні додатків. Тому завдання полягає в тому, щоб захистити сервери веб-додатків від різних загроз

безпеки, які використовують вразливості додатків. Поєднання набору інструментів і належної методики дозволяє охопити більш широкий спектр проблем безпеки в веб-додатку.

Процес тестування на проникнення можна розділити на п'ять загальних етапів: планування і збір інформації, на яких визначаються обсяг і мета тестів і збираються дані, необхідні для розуміння того, як працює об’єкт досліджень і його потенційні слабкі місця; етап сканування, на якому тестувальник зрозуміє, як додаток може реагувати на різні вторгнення шляхом сканування за допомогою статичного або динамічного аналізу; третій етап – це експлуатація, використання виявлених вразливостей тестувальником. Після цих трьох початкових етапів проводиться аналіз результатів, щоб переконатися, що виявлені недоліки є реальними вразливими місцями, а не помилковими спрацьовуваннями. Останній етап звіт міститиме важливі та детальні дані, які включають доступ до інформації з обмеженим доступом, конкретні використані вразливості, час, протягом якого пентестер залишався непоміченим у системі, та іншу необхідну інформацію.

1 СПОСОБИ ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ У ВЕБ-ЗАСТОСУНКАХ

Тестування програмного забезпечення – це процес пошуку помилок у реалізації програми [2]. Організувати тестування таким чином, щоб виявити всі помилки програми майже неможливо. Можна стверджувати, що правильність і абсолютну свободу від помилок програми гарантують:

- підготовлені всі можливі набори вхідних даних (в тому числі і некоректні) ;
- виконання програми при всіх можливих перестановках і поєднаннях цих вхідних даних;
- аналіз всіх вихідних даних і установка факту, що кожен тестовий вихідний набір правильний.

Для реалізації всіх вищенаведених пунктів потрібні колосальні витрати часу, фінансові та людські ресурси, адже навіть для невеликих програм з парою вхідних параметрів число тестових випадків може обчислюватися мільйонами комбінацій. Це доводить, що вичерпне тестування програмного забезпечення неможливе. Отже, неможливо виявити всі помилки в реалізації програми. Саме тому інтуїтивний процес тестування не приносить, як правило, великої користі, тим більше що в умовах жорсткого обмеження часу і коштів на розробку вибір маленької підмножини тестових ситуацій часто є вимушеною необхідністю. Повний вибір необхідно робити не спонтанно, а на підставі певної стратегії, якої слід дотримуватися при тестуванні продукту. Це доводить, що дослідження сучасних методів, що дозволяють проводити тестування систематично, а не інтуїтивно, є актуальною проблемою.

1.1 Сканування вразливостей

Сканер безпеки додатків – це інструмент, який налаштований для запиту певних інтерфейсів для виявлення прогалин у безпеці та продуктивності. Ці

інструменти покладаються на документовані Інструменти та сценарії для перевірки відомих слабких місць. Сканери вразливостей моделюють різні сценарії "якщо-то" для оцінки дій користувача і системних конфігурацій, які можуть сприяти експлойту. Ефективно налаштоване пасивне сканування веб-безпеки допомагає перевіряти програми та мережі, а потім надає журнал слабких місць, які необхідно усунути в порядку пріоритету.

Переваги інструментів сканування вразливостей:

1) Швидкі результати. Головна перевага автоматизованого інструменту сканування полягає в тому, що він відносно швидко видає результат. Таким чином, можна отримати уявлення про свою безпеку в будь-який зручний час.
2) Повторюваність. Автоматичне сканування вразливостей легко повторити. Користувач сам вирішує, чи хоче він виконувати перевірку щодня, щотижня або щомісяця і отримувати оновлену інформацію про виявлені зміни і вразливості.
3) Зручний для користувача. Більшість інструментів сканування вразливостей мають зрозумілий інтерфейс і тому прості у використанні. Таким чином, бар'єр для системних адміністраторів та інших осіб, які використовують їх, невеликий. Однак слід зазначити, що результати роботи інструментів містять досить докладні відомості. Це означає, що фахівець з безпеки як і раніше необхідний для інтерпретації результатів і вжиття заходів.
4) Постійний моніторинг. Засіб сканування вразливостей також може бути ефективно розгорнуто для постійного моніторингу, наприклад, при виконанні великої кількості розгортань. Крім того, він надає системним адміністраторам безперервну інформацію про стан інфраструктури.

Недоліки інструментів сканування вразливостей:

1) Засіб сканування вразливостей не виявить майже всіх вразливостей. Оскільки засіб сканування вразливостей також пропускає вразливості. Це одне з найбільших обмежень всіх інструментів сканування, оскільки все ще можуть існувати уразливості, якими можуть скористатися хакери. Для цього є дві можливі причини:
   - Сканер не знає про вразливість, наприклад, тому, що вона була виявлена тільки що.
   - Вразливість занадто складна, щоб її можна було виявити за допомогою автоматизованого інструменту, оскільки автоматизувати атаку нетривіально.
2) Потрібні постійні оновлення. Щоб переконатися, що виявлені найостанніші вразливості, необхідно переконатися, що інструмент постійно оновлюється.
3) Помилкове спрацьовування. Особливо, якщо велика IT-інфраструктура, безліч серверів і служб, може бути важко зрозуміти вплив вразливостей. В результаті інструмент сканування часто буде стикатися з помилковими спрацьовуваннями. Розпізнати їх непросто, що робить інтерпретацію результатів справою, що забирає багато часу. Більш того, якщо помилкові спрацьовування не відфільтровуються, інструмент не стає розумнішим і буде продовжувати видавати помилкові результати.
4) Наслідки вразливості незрозумілі. Якщо виявляється вразливість, іноді важко оцінити, що це означає для бізнес-операцій. Як це вплине на різні відділи, співробітників і процеси. Автоматизований інструмент не скаже вам про це, і системний адміністратор, як правило, буде більше зосереджений на технічному аспекті уразливості.

1.2 Проведення тестування на проникнення

Кібербезпека – це практика захисту комп'ютерів, серверів, мобільних пристроїв, електронних систем, мереж та даних від шкідливих атак [3]. Це також відоме як безпека інформаційних технологій або електронна

інформаційна безпека. Цей термін застосовується в різних контекстах, від бізнесу до мобільних обчислень, і може бути розділений на кілька загальних категорій. Багато організацій зрозуміли, що застосування тих самих методів, процедур та інструментів, які використовуються зловмисниками, було б ефективною формою тестування та оцінки ризиків та загроз для їх систем [3].

Ці проблеми можуть бути пов'язані з мережевою безпекою, безпекою додатків, інформаційною безпекою, операційною безпекою або навчанням кінцевих користувачів. Для ефективного вирішення цих проблем формується політика безпеки, відповідно до вимог та цілей якої забезпечується достатній рівень захисту від різних атак [4].

Атака – це будь-яка ворожа, агресивна або зловмисна дія, що виконується з використанням недоліків своєї інфраструктури або служб проти самого себе, а зловмисник – це якась сутність, яка виконує атаку [5].

Існує два типи кібератак: пасивні та активні [5]. Пасивні атаки важко виявити. Вони лише відстежують і сканують трафік між комп'ютерами. Активні атаки – це спроби внести несанкціоновані зміни в систему. Їх легко виявити, але зазвичай вони призводять до значних пошкоджень, таких як зміни даних, що передаються та зберігаються, або нові потоки даних, створені для доступу до ресурсів [5].

Тестування на проникнення – це попереджувальний підхід до забезпечення безпеки, при якому фахівці з безпеки намагаються безпечно використовувати вразливості, такі як різні типи SQL-ін'єкцій, міжсайтові сценарії, підробка міжсайтових запитів і міжсайтові запити. Після виявлення вразливостей організації, як правило, імітують і розуміють дії зловмисника. Групи безпеки проводять тести на проникнення для оцінки ефективності механізмів безпеки та відповідності політикам безпеки. Для цього тестувальники імітують робочий процес зловмисника, покладаючись на існуючі вразливості і підвищення привілеїв для доступу до системних даних.

Потім вони описують докладні звіти про інформацію, отриману в результаті тесту, які потім використовуються для точного налаштування засобів контролю безпеки [6].

Переваги тестування на проникнення:

1) Вони можуть виявити цілий ряд вразливостей. Підприємства піддаються безлічі потенційних загроз, і кожна з них може використовувати сотні різних вразливостей. Такі уразливості відкриті для потенційно руйнівних атак, таких як впровадження SQL, і такі, здавалося б, безпечні речі, як сторінки з помилками, можуть надати зловмисникам достатньо інформації, щоб використовувати менш очевидну і набагато більш шкідливу вразливість.
2) Вони можуть виявляти вразливі місця з високим ризиком, що виникають в результаті поєднання більш дрібних вразливостей. Взяті самі по собі, невеликі уразливості можуть здатися незначними, але хакери часто шукають ці слабкі місця для створення послідовностей вторгнень, які вимагають невеликих, постійних зусиль, щоб розкрити проломи в безпеці в набагато більшу слабкість. Ці прогалини часто не враховуються компанією або автоматизованими системами безпеки, але, враховуючи, що тестери пера копіюють методи хакера, вони зможуть ідентифікувати такі точки входу.
3) У звітах будуть міститися конкретні рекомендації. Заключним етапом тесту на проникнення є повідомлення про вразливості. На відміну від автоматично генеруються звітів з інструментів, які пропонують загальні рекомендації щодо усунення неполадок, звіти з тестів на проникнення можуть ранжувати і оцінювати уразливості відповідно до масштабом ризику і бюджетом компанії.

Недоліки тестування на проникнення:

1) Якщо вони не будуть виконані правильно, вони можуть завдати великої шкоди. Тести, які не виконуються належним чином, можуть призвести до збою серверів, розкриття конфіденційних даних, пошкодження важливих виробничих даних або викликати безліч інших побічних ефектів, пов'язаних з імітацією кримінального злому.
2) Ви повинні довіряти тестеру проникнення. Тестування на проникнення, по суті, означає, що Ви запрошуєте когось зламати ваші системи, тому ви покладаєтеся на те, що Тестувальник не зловжить своїми навичками і знаннями. Якщо ви не наймаєте когось, кому ви можете довіряти, для виконання цієї роботи, ваші спроби забезпечення безпеки можуть мати неприємні наслідки.
3) Якщо ви не використовуєте реалістичні умови тестування, результати будуть оманливими. Співробітники, швидше за все, готуються до випробування, яке, як вони знають, має відбутися, а це означає, що організація виглядає сильніше, ніж є насправді. Справжня атака відбудеться без попередження, причому творчими способами, які важко спланувати.

1.3 Постановка завдань досліджень

Методика – це, алгоритм, процедура щодо будь-яких націлених дій. У науці спосіб та порядок дослідження предмета для отримання найбільш повного та відповідного істині результату. Методика відрізняється від методу конкретизацією прийомів та завдань. Основні види логічних та наукових методик: аналіз, синтез, індукція, дедукція [7].

Для розробки та обґрунтування ефективної методики пошуку вразливостей у веб-застосунках необхідно:

1) Проаналізувати існуючи стандарти та методики в області пошуку вразливостей програмного забезпечення;

2) Враховуючи досвід та виявлені недоліки існуючих підходів розробити власну методику пошуку вразливостей у веб-застосунках;
3) Для обґрунтування розробленої методики навести приклад використання цієї методики на конкретному веб-застосунку.

2 АНАЛІЗ СТАНДАРТІВ І МЕТОДИК В ОБЛАСТІ ПОШУКУ ВРАЗЛИВОСТЕЙ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У міжнародній практиці, проведення тестів на проникнення регламентується стандартами і методиками, які регламентують етапи тестування, порядок випробувань тестованих об'єктів, порядок взаємодії аудитора з замовником і т. д. До широко поширеним зарубіжним стандартам і методикам відносяться:

1) Методика OSSTMM - The Open Source Security Testing Methodology Manual [8];
2) методика ISSAF - Information System Security Assessment Framework [9];
3) методика OWASP - Open Web Application Security Project [10];
4) стандарт PTES - Penetration Testing Execution Standard [11];
5) стандарт NIST SP 800-115 - Technical Guide to Information Security Testing and Assessment [12];
6) методика BSI - Study a penetration testing Model [13];
7) методика PETA - Methodology of Information Systems Security Penetration Testing [14];
8) методика PTF - penetration testing Framework [15].

2.1 Базова термінологія і класифікація тестових впливів

Введемо базову термінологію, яку будемо використовувати надалі.

Об'єкт – інформаційна система, інформаційно-телекомунікаційна мережа, автоматизована система управління, в відношення яких проводиться аудит ІБ [16].

Тестування – перевірка виконання вимог до об'єкта за допомогою спостереження за його роботою в кінцевому наборі спеціально обраних ситуацій [16].

Тест – окремий захід з дослідження об'єкта або спосіб вивчення процесів його функціонування [16].

Інформаційно-технічний вплив – вплив на інформаційний ресурс, інформаційну систему, інформаційну інфраструктуру, на технічні засоби або на програми, що вирішують завдання формування, передачі, обробки, зберігання і відтворення інформації, з метою викликати задані структурні або функціональні зміни [16].

Тестовий інформаційно-технічний вплив – вплив на інформаційний ресурс, інформаційну систему, інформаційну інфраструктуру, на технічні засоби або на програми, що вирішують завдання, формування, передачі, обробки, зберігання і відтворення інформації, з метою виявити уразливості об'єкта на яке здійснюється вплив [16].

Інформаційно-психологічний вплив – інформаційний, психотронний або психофізичний вплив на психіку людини або групи людей, що впливає на сприйняття ними реальної дійсності, в тому числі на їх поведінкові функції, а, в деяких випадках, і на функціонування органів і систем людського організму [16].

Тестовий інформаційно-психологічний вплив – інформаційне, психотронне або психофізичний вплив на психіку людини або групи людей, що робить вплив на сприйняття ними реальної дійсності, в тому числі на їх поведінкові функції, а, в деяких випадках, і на функціонування органів і систем людського організму, з метою виявити уразливості об'єкта на яке проводиться вплив [16].

Тестування на проникнення – експериментальна перевірка з метою оцінювання стану ІБ і виявлення вразливостей об'єкта тестування (тестованої

системи) шляхом інтегрального і цілеспрямованого застосування проти нього спеціальних засобів і способів ІТВ і ІПВ [16].

Збиток – еквівалентна вартість всіх видів втрат (фінансових, репутаційних, матеріальних та ін.), які понесе об'єкт або його власник в результаті інциденту [16].

Інцидент – факт порушення властивостей ІБ в процесах формування, передачі, обробки, зберігання і відтворення інформації на об'єкті та/або припинення функціонування об'єкта, в тому числі те, що сталося в результаті впливу ІТВ або ІПВ [16].

Вразливість – недолік об'єкта, експлуатація якого робить можливим реалізацію інциденту, нанесення об'єкта пошкоджень будь-якої природи, або зниження ефективності його функціонування [16].

Експлойт – потенційно нешкідливий набір даних або послідовності дій, які некоректно обробляється інформаційною системою, внаслідок помилок в ній. Результатом некоректної обробки такого набору даних або послідовності дій може бути перехід об'єкта в вразливий стан [16].

В рамках тестування на проникнення повинні реалізовуватися сценарії поетапного інтегрального застосування засобів і способів ІТВ і ІПВ, які з високим ступенем ймовірності будуть застосовуватися реальними зловмисниками, що дозволить провести всеосяжний аналіз вразливостей тестованих об'єктів, а також сформувати пропозиції щодо вдосконалення системи захисту.

2.2 Методика OSSTMM

Методика OSSTMM - The Open Source Security Testing Methodology Manual розроблена Інститутом ISECOM (Institute for Security and open Methodologies), який є відкритим співтовариством вчених і практиків в області ІБ [8].

Методика OSSTMM є у високому ступені формалізованим і добре структурованим документом регламентує практично всі аспекти тестування на проникнення, орієнтована на тестування переважно комп'ютерних мереж. Методика періодично оновлюється. З недоліків варто відзначити малу кількість інформації з практичних дій та інструментарію тестування [8].

Методика OSSTMM містить наступні розділи:

1) первинні відомості про документ;
2) визначення тестування на проникнення, рамок тестування, ролей і процесів;
3) аналіз безпеки об'єкта;
4) показники (метрики) безпеки об'єкта;
5) аналіз соціальних процесів в персоналі об'єкта тестування;
6) процес тестування;
7) тестування стійкості персоналу до ІПВ та соціальної інженерії;
8) тестування безпеки фізичної інфраструктури;
9) тестування безпеки бездротових технологій;
10) тестування безпеки Телекомунікаційних технологій;
11) тестування безпеки даних;
12) рекомендації щодо дотримання національних стандартів та угод;
13) підготовка звіту про тестування.

Методика OSSTMM визначає, так звану «карту безпеки» – візуальне відображення основних категорій ІБ, які оцінюються в процесі тестування:

- інформаційна безпека;
- безпека соціальних процесів;
- безпека інформаційних процесів;
- безпека Інтернет-технологій;
- безпека каналів зв'язку;
- безпека бездротових технологій;

- безпека фізичної інфраструктури.

Як такої, класифікації вразливостей в цій методиці немає. Поняття "вразливість" в методиці вводиться як обмеження безпеки – це дефект або помилка, яка забороняє доступ авторизованим користувачам або процесам до інформаційних ресурсів або дозволяє несанкціонований доступ (НСД) неавторизованих користувачів або процесів до ресурсів [8].

Цю методику можна використовувати як на етапі попередньої оцінки захищеності об'єктів в інтересах перевірки можливості їх використання в складі будь-якої інформаційної системи, так і на етапі розробки об'єктів для перевірки окремих можливостей і функцій ІБ.

2.3 Методика ISSAF

Методика ISSAF - Information System Security Assessment Framework розроблена консорціумом OISSG (Open Information Systems Security Group) як стандарт внутрішнього аудиту організацій цього консорціуму [9]. При даному аудиті виконується оцінка наступних аспектів ІБ:

- оцінка політик і процедур ІБ організації, а також ступінь їх відповідності IT-стандартам і вимогам нормативних документів в області ІБ;
- виявлення та оцінка "залежності" бізнес-процесів організацій від IT-інфраструктури;
- проведення оцінки вразливостей і тестів на проникнення для виділення вразливостей в системі, які можуть призвести до потенційних ризиків інформаційних ресурсів;
- вказівка моделей оцінки за доменами безпеки;
- знаходження і усунення неправильних конфігурацій апаратнопрограмних засобів;
- ідентифікація та зниження ризиків, пов'язаних з IT;

- ідентифікація та зниження ризиків, пов'язаних з персоналом або бізнес-процесами;
- посилення безпеки існуючих процесів і технологій;
- впровадження кращого досвіду забезпечення ІБ в практику і процедури бізнес-процесів.

Методика ISSAF включає в себе велику кількість питань, пов'язаних з тестуванням ІБ, а матеріал методики організований у вигляді двох частин [9]:

- рекомендації для менеджменту;
- рекомендації з тестування.

Матеріал методики ISSAF поділений на 14 підрозділів [8].

1) Управління проектом з тестування.
2) Основні принципи і кращі практики в проведенні тестування.
3) Схема процесу тестування.
4) Огляд політики безпеки і способів підвищення ІБ.
5) Методологія оцінки ризиків.
6) Тестування технічних аспектів ІБ:
    a) тестування криптостійкості паролів;
    b) тестування безпеки операційної системи (ОС) Unix / Linux;
    c) тестування безпеки ОС Windows;
    d) тестування безпеки ОС Novell Netware;
    e) тестування безпеки баз даних (БД);
    f) тестування безпеки бездротових мереж і комунікацій;
    g) тестування безпеки комутаторів;
    h) тестування безпеки маршрутизаторів;
    i) тестування безпеки брандмауерів;
    j) тестування безпеки систем виявлення вторгнень;
    k) тестування безпеки приватних віртуальних мереж VPN;
    l) тестування безпеки антивірусних систем;

m) тестування безпеки розподілених систем зберігання даних;
n) тестування безпеки інтернет-комунікацій;
o) тестування безпеки користувачів;
p) тестування безпеки вихідного коду;
q) тестування безпеки бінарного коду.

7) Тестування соціально-психологічних аспектів ІБ.
8) Тестування фізичної інфраструктури.
9) Аналіз інцидентів.
10) Звітність за результатами аудиту та тестування.
11) Забезпечення безперервності бізнес-процесів і відновлення після інцидентів.
12) Підвищення повноти моніторингу та навчання в області ІБ.
13) Аутсорсинг проведення тестування та забезпечення ІБ.
14) База знань:
    a) правові аспекти тестування та аудиту ІБ;
    b) рекомендації щодо складання договору про нерозголошення інформації;
    c) рекомендації щодо складання договору на тестування та аудит;
    d) шаблони типових документів;
    e) контрольний список перевірки ОС Windows;
    f) контрольний список перевірки ОС Linux;
    g) контрольний список перевірки ОС Solaris;
    h) порти за замовчуванням у брандмауерів;
    i) порти за замовчуванням у систем виявлення вторгнень;
    j) посилання на інші документи та ресурси;
    k) рекомендації щодо програмного забезпечення (ПЗ), яке можна використовувати для проведення тестування.

У методиці ISSAF представлені 3 етапи, які необхідно реалізувати для коректного проведення тестів на проникнення.

1) Планування та підготовка. Отримання початкової вихідної інформації про об'єкт тестування, планування і підготовка до тестів. Перед тестуванням сторонам необхідно буде підписати формальну угоду, яка забезпечить основу для проведення тестування і взаємний правовий захист. У ньому також буде вказано порядок взаємодії, точні дати, тривалість тестування, способи проведення тестування і т. д. [9].
2) Оцінка. На цьому етапі проводиться виконання тестування. Передбачені наступні підетапи проведення тестування [9]:
   a) збір інформації. Для збору інформації в методиці ISSAF рекомендується використовувати Інтернет. При цьому одержувана інформація ділиться на дві групи: Технічна (DNS/WHOIS) і нетехнічна (пошукові системи, групи новин і т.д.). Даний етап дозволяє виділити «точки уразливості», які будуть використовуватися в подальшому;
   b) мережеве картографування. Застосування спеціальних технічних засобів для визначення структури мережі та її ресурсів;
   c) ідентифікація вразливостей. Перед цією стадією, аудитор визначає вразливі об'єкти і способи їх тестування. У процесі тестування методикою ISSAF передбачається виконання наступних заходів:
      - ідентифікація вразливостей поштових сервісів;
      - виконання поглибленого сканування мережевих інформаційних ресурсів і мережевих сервісів на предмет пошуку відомих вразливостей. Інформація про відомі вразливості береться з відкритих баз даних вразливостей;
      - верифікація отриманої інформації про вразливості шляхом порівняння і перевірки інформації про вразливості, отриманих з різних джерел або різними способами;
      - документування виявлених вразливостей;
      - класифікація знайдених вразливостей;
      - визначення сценаріїв ІТВ і сценаріїв використання експлойтів.

Відзначимо, що в методиці ISSAF для вразливостей визначається два типи ризиків: технічний ризик і бізнес-ризик. У свою чергу кожен з них ділиться на 3 рівня: Низький, Середній, Високий [9].

3) Безпосередньо тестування на проникнення.
4) Отримання доступу або розширення привілеїв. Отримання мінімальних привілеїв доступу можливо через доступ до непривілейованих акаунтів за допомогою таких способів:
   a) підбір комбінацій логін / пароль шляхом атаки зі словником;
   b) пошук порожніх або стандартних паролів в системних акаунтах;
   c) виявлення експлойтів в стандартних налаштуваннях мережевого обладнання;
   d) пошук публічних сервісів, що допускають певні операції в системі (запис/створення/читання файлів).

Кінцевою метою аудитора на даному етапі є отримання доступу до аккаунту адміністратора мережі. Часто в мережі дозволені тільки акаунти з мінімальною кількістю привілеїв. В цьому випадку виконується складання карти локальних вразливостей, проводиться розробка або отримання коректного експлойта, потім він тестується в ізольованому середовищі і застосовується до компрометованої системі.

5) Додаткові тести, наприклад, отримання зашифрованих паролів для їх подальшого злому в режимі off-line, перехоплення трафіку і його аналіз і т.д.
6) Компрометація віддалених користувачів, інформаційних ресурсів, об'єктів мережі.
7) Підтримка несанкціонованого доступу до мережі.
8) Приховування слідів роботи.

Методика ISSAF є найбільш докладною, з розглянутих в даній статті, методикою тестування на проникнення як в теоретичному, так і в практичному

плані. Цю методику можна використовувати як на етапі попередньої оцінки захищеності об'єктів мережі в інтересах перевірки можливості їх використання в складі будь-якої інформаційної системи, так і на етапі розробки об'єктів для перевірки окремих можливостей і функцій ІБ.

2.4 Методика OWASP

Методика OWASP – Open Web Application Security Project створена спільнотою OWASP в 2004 р. і розвивається по теперішній час міжнародною групою незалежних експертів-ентузіастів. Методика орієнтована на тестування веб-додатків. Організація OWASP зареєстрована в США і Бельгії (OWASP Europe VZW). Методика докладно описує тестування веб-додатків і фактично є єдиною подібною методикою, вузько орієнтованою саме на веб-додатки [10].

У 2020 р вийшла проміжна версія керівництва OWASP Web Security testing Guide v. 4.1. У посібнику з тестування є посилання на перелік заходів з тестування на проникнення (checklist), а також розкривається вміст цих заходів.

Методика OWASP містить наступні розділи [10]:

1) Вступ;
2) посібник з тестування OWASP;
3) тестування на проникнення веб-додатків;
4) керівництво по складанню звітів.

У розділі 3 "тестування на проникнення веб-додатків" описаний набір тестів, орієнтованим на перевірку наступних аспектів ІБ:

- збір інформації;
- тестування керування конфігурацією та розгортанням;
- тестування управління ідентифікацією;
- тестування аутентифікації;

- тестування авторизації;
- тестування управління сесіями;
- тестування перевірки введення;
- тестування обробки помилок;
- тестування на криптографічну стійкість;
- тестування бізнес-процесів;
- тестування клієнтської сторони.

Методику OWASP можна використовувати як на етапі попередньої оцінки захищеності веб-додатків в інтересах перевірки можливості їх використання в складі будь-якої інформаційної системи, так і на етапі розробки веб-додатків для перевірки окремих можливостей і функцій ІБ.

2.5 Стандарт PTES

Стандарт проведення тестування на проникнення PTES – Penetration Testing Execution Standard розроблена в 2009 р. міжнародною групою незалежних експертів-ентузіастів в області ІБ [11]. PTES якості стандарту офіційно зареєстрована тільки в США. З моменту появи стандарт отримав розвиток у вигляді версії 1.1 в 2017 р.

Стандарт PTES передбачає 7 основних етапів проведення тестування на проникнення, описаних у відповідних розділах [11]:

1) етап початкового спілкування;
2) збір інформації;
3) моделювання загроз;
4) аналіз вразливостей;
5) експлуатація;
6) пост-експлуатація;
7) звітність.

До даного стандарту додається технічне керівництво (PTES Technical Guidelines) докладно викладає основні технічні аспекти тестування [11]:

1) інструментарій тестування:
   a) ОС і ПЗ;
   b) апаратні засоби;
   c) радіотехнічні засоби;
2) збір інформації про об'єкт тестування:
   a) збір інормації за відкритими джерелами (OSINT);
   b) використання ІПВ та соціальної інженерії;
   c) аналіз соціальних мереж і контактів;
   d) аналіз email і телефонних контактів;
   e) сканування мережі;
   f) аналіз використовуваного ПЗ і ОС;
   g) аналіз периметра безпеки;
   h) аналіз фізичної інфраструктури;
3) аналіз вразливостей:
   a) аналіз вразливостей ОС і ПЗ;
   b) аналіз вразливостей БД;
   c) аналіз вразливостей VPN;
   d) аналіз вразливостей транспортної мережі та мережевих протоколів;
   e) аналіз вразливостей бездротової мережі;
   f) аналіз вразливостей інтернет-підключень;
   g) аналіз вразливостей аутентифікації і кріптоустойчівості паролів;
   h) формування цільових ІТВ комп'ютерної розвідки;
4) експлуатація вразливостей (формування забезпечують і атакуючих ІТВ, орієнтованих на проникнення за захищається периметр організації за рахунок експлуатації виявлених вразливостей):
   a) атаки на ОС і ПЗ;
   b) атаки на аутентифікацію за стандартними паролями;

c) атаки на мережеві протоколи;
d) атаки на VPN;
e) DoS-атаки;
f) атаки на протоколи сімейства WEP і WPA бездротових мереж;
g) атаки на шлюзи з мережею Інтернет;
h) ІПВ і соціальна інженерія;
i) атаки на інфраструктуру контролю периметра (відеоспостереження, пропускна система, Облік пересувань персоналу і т. д.);

5) постексплуатація (формування атакуючих ІТВ, орієнтованих на розширення привілеїв і несанкціоновані дії після проникнення за захищається периметр організації):
   a) експлуатація вразливостей ОС, ПЗ і БД;
   b) формування закладок і вразливостей для подальшої експлуатації;
   c) отримання доступу і робота з системними файлами;
   d) отримання доступу до важливих файлів;
   e) отримання доступу до інформації авторизації користувачів;
   f) обхід внутрішніх засобів захисту;
6) звітність.

Дані етапи охоплюють практично всі дії, пов'язані з тестом на проникнення – від початкового спілкування і обгрунтування завдання на тестування до етапу формування звіту, в якому весь процес фіксується найбільш ергономічним для замовника чином, а також формуються рекомендації щодо підвищення захищеності тестованої системи [11].

Цей стандарт можна використовувати як на етапі попередньої оцінки захищеності об'єктів в інтересах перевірки можливості їх використання в складі будь-якої інформаційної системи, так і на етапі розробки об'єктів для перевірки окремих їх можливостей і функцій ІБ.

2.6 Стандарт NIST SP 800-115

Стандарт NIST SP 800-115 – Technical Guide to Information Security Testing and Assessment розроблений і підтримується в актуальному стані одним з підрозділів національного інституту стандартизації NIST (National Institute of Standards and Technology), а саме – центром комп'ютерної безпеки CSRC (Computer Security Resource Center), що об'єднує фахівців федеральних служб, університетів і найбільших IT-компаній США [12].

Матеріал даного стандарту організований у вигляді послідовності наступних розділів [12]:

1) огляд тестування та експертизи безпеки;
2) огляд методів;
3) визначення мети і техніки аналізу;
4) техніки оцінки вразливостей об'єктів;
5) планування оцінки безпеки;
6) виконання оцінки безпеки;
7) пост-тестові заходи.

У розділі «техніки оцінки вразливостей об'єктів», в якості однієї з технік описуються типові тести на проникнення, а саме їх етапи і логіка проведення. Відповідно до цього стандарту, тести на проникнення рекомендується проводити в наступних випадках [12]:

- для визначення стійкості і захищеності об'єкта до реально існуючих ІТВ;
- для визначення трудомісткості подолання периметра захисту об'єкта;
- для з'ясування вразливостей існуючих заходів і засобів захисту;
- для визначення здатності системи захисту об'єкта, своєчасно виявляти реальні ІТВ і адекватно реагувати на них.

Відповідно до стандарту NIST SP 800-115, в тестуванні на проникнення виділяють наступні етапи [12].

1) Планування. На даному етапі визначаються правила тестування, затверджуються і документується управління тестуванням, визначаються мета і приватні завдання тестування.
2) Дослідження. Даний етап включає в себе два підетапи:
   - комп'ютерна розвідка об'єкта тестування в інтересах збору доступної інформації про об'єкт тестування, що захищається периметрі і т. д.;
   - аналіз вразливостей в інтересах формування переліку перспективних вразливостей які можуть використовуватися в якості цілей для тестових ІТВ.
3) Атака. Реалізація ІТВ на раніше визначені уразливості. Якщо ІТВ закінчується інцидентом, то уразливості присвоюється статус «актуальна» і надалі визначаються заходи щодо її усунення.
4) Звіт. Відповідно до Методики, формування звітних документів проводиться на всіх вищевказаних етапах. Під час етапу «планування» розробляється план тестування. Під час етапу «дослідження» і «атака» зберігаються лог-файли, створюються періодичні звіти для системних адміністраторів і фахівців. На закінчення тесту, створюється підсумковий звіт, який, як правило, містить описи виявлених вразливостей, оцінки ризиків, вказівок щодо усунення вразливостей і модернізації системи захисту.

Стандарт NIST SP 800-115 можна використовувати як на етапі попередньої оцінки захищеності об'єктів в інтересах перевірки можливості їх використання в складі будь-якої інформаційної системи, так і на етапі розробки об'єктів для перевірки окремих можливостей і функцій ІБ. Також цей стандарт можна використовувати як шаблон для розробки - які стандартні функції забезпечення ІБ повинні бути присутніми в об'єкті, що розробляється.

Недоліком стандарту NIST SP 800-115 є те, що він був прийнятий в 2008 р. і в даний час не повною мірою відображає сучасні підходи до тестування на проникнення.

2.7 Методика BSI

Методика BSI - Study a penetration testing Model розроблена німецькою державною організацією Federal Office for Information Security. У цій методиці описується проведення випробувань об'єкта на стійкість до ІТВ, при цьому докладно описуються не тільки послідовність проведення тестових ІТВ, а й необхідні вимоги по ІБ, а також правові аспекти тестування на проникнення [13].

Матеріал методики BSI організований в наступні розділи:

1) ІТ-безпека та тести на проникнення;
2) об'єкти тестів на проникнення та їх класифікація;
3) правові питання;
4) загальні вимоги;
5) методика проведення тестів на проникнення;
6) виконання тестів на проникнення.

Згідно з методикою BSI, виділяється три основних типи впливів:

1) ІТВ через мережу;
2) ІПВ та соціальна інженерія;
3) обхід фізичних заходів безпеки.

При цьому в методиці BSI визначені наступні основні етапи тестування об'єкта [13].

1) Підготовка до тестування. Замовник визначає об'єкти тестування. Визначаються ресурси, ризики, перевіряються вимоги по ІБ. Обговорюються правові аспект. Складається договір про проведення тестування.
2) Збір інформації. Це етап пасивного тестування, мета якого отримати якомога повнішу інформацію про об'єкт, встановлені операційні системи (ОС) і програмне забезпечення (ПЗ), дані про потенційні цілі

атакуючих ІТВ, а також про відомі недоліки ІБ. Даний етап включає в себе ряд підетапів:

a) пошук інформації про об'єкт тестування;
b) використання ІТВ для проведення комп'ютерної розвідки об'єкта;
c) визначення операційної системи і додатків;
d) виявлення вразливостей об'єкта.

3) Аналіз інформації та ризиків. Для успішної, прозорої та ефективної процедури тестування, зібрана інформація повинна бути проаналізована перед початком етапу тестування атакуючими ІТВ. Аналіз повинен включати в себе визначення цілей ІТВ, потенційні ризики для об'єкта, ймовірність заподіяння шкоди об'єкту, час, необхідний для проведення тестування атакуючими ІТВ, їх орієнтованість на виявлені на попередньому етапі уразливості об'єкта.
4) Спроби активного вторгнення. Тестування та аналіз можливостей експлуатації вразливостей об'єкта, виявлених на етапі збору інормації, шляхом реалізації атакуючих ІТВ.
5) Аналіз результатів. Кінцевий звіт повинен містити оцінку вразливостей об'єкта у вигляді формулярів потенційних ризиків, а також рекомендації щодо усунення вразливостей і ризиків. Звіт також повинен гарантувати прозорість тестів і розкриття вразливостей.
6) Документування. Це не окремий етап, а постійна процедура, що передбачає журналювання, запис, обробка і вироблення рекомендацій під час всіх вищеописаних етапів.

У додатках методики BSI містяться опис ПЗ, яке можна використовувати для тестування об'єктів, описаних в методиці [13].

Дану методику рекомендується використовувати для тестування кінцевого продукту. Методика BSI є досить докладною, а її розробники намагалися передбачити всі аспекти тестування на проникнення: технічні, організаційні, правові.

2.8 Методика PETA

Методика PETA – Methodology of Information Systems Security Penetration Testing є прикладом проектного підходу до організації тестування інформаційних систем. Дана методика пропонує наступну послідовність етапів тестування на проникнення [14].

1) Планування:
   - формування замовником вимог до результатів тестування;
   - формування домовленостей про проведення тестування між замовником та аудитором;
   - формування команди менеджерів проекту;
   - визначення області тестування;
   - визначення правил тестування;
   - формування групи тестування;
   - опис ролей;
   - проведення брифінгів та обговорень стратегії тестування.
2) Тестування:
   - збір інформації про об'єкт, що тестується;
   - аналіз периметра захисту об'єкта, що тестується;
   - проникнення за периметр;
   - аналіз мережі (використання забезпечують ІТВ);
   - аналіз вразливостей (використання ІТВ);
   - закріплення за периметром, розширення повноважень (використання атакуючих ІТВ);
   - отримання доступу до цільових інформаційних ресурсів, проведення несанкціонованих дій в системі (використання атакуючих ІТВ);
   - використання ІПВ та методів соціальної інженерії;
   - підтримання несанкціонованого доступу в актуальному стані;
   - приховування слідів.

3) Формування звіту:

- повернення тестованої системи в початковий стан, видалення наслідків ІТВ;
- аналіз отриманої в ході тестування інформації;
- формування підсумкового звіту замовнику;
- подання результатів тестування замовнику, підсумкове приймання ним результатів.

Вищевказані етапи в даній методиці формалізовані у вигляді процесної моделі, яка, однак розписані не дуже докладно. Методика є досить оглядовою і визначає тільки самі загальне підходи до проведення тестування на проникнення, залишаючи вибір конкретних цілей тестування, використовуваних тестових ІТВ, і інші параметри тестування на розсуд замовника і аудитора.

2.9 Методика PTF

Методика PTF – Penetration testing Framework, судячи з матеріалів сайту є детальним технічним керівництвом з проведення тестування на проникнення в технічній частині. Даний посібник не містить загальнотеоретичної інформації, подібно методикам OSSTMM або ISSAF, однак надає практично вичерпний перелік вразливостей об'єкта підлягають перевірці, в деяких випадках, із зазначенням рекомендованої Порядку проведення тестування та інструментарію для нього [15].

Матеріал методики PTF організований у вигляді наступних розділів:

1) збір інформації про об'єкт за рахунок проведення активного і пасивного моніторингу, пошуку інформація в мережі Інтернет, з використанням ІПВ і способів соціальної інженерії;
2) аналіз ОС і ПЗ об'єкта;
3) аналіз портів мережі;
4) перевірка паролів;

5) перевірка вразливостей:
   - уразливості віддаленого доступу;
   - вразливості внутрішнього доступу;
   - уразливості bluetooth;
   - вразливості телекомунікаційного обладнання Cisco;
   - уразливості системи Citrix;
   - уразливості транспортних мереж;
6) безпосередньо тестування на проникнення – використання виявлених вразливостей для реалізації таргетованих ІТВ щодо об'єкта тестування. При цьому окремо розглянуті:
   - проникнення на сервера;
   - оцінка стійкості VoIP-інфраструктури;
   - проникнення через бездротові мережі;
   - безпека фізичної інфраструктури.
7) формування підсумкового звіту.

Необхідно відзначити, що методика PTF, фактично є приватним проектом фахівця з ІБ K. Orrey [15]. Разом з тим, дана методика отримала велику кількість позитивних відгуків фахівців з тестування на проникнення, які використовували дану методику як першооснову для розробки свого варіанту тестування. У зв'язку з цим дана методика була включена в цей огляд.

2.10 Результати порівняльного аналізу стандартів і методик

Узагальнені результати аналіз вищевказаних методик тестування на проникнення за різними критеріями представлені в таблиці 2.1.

Таблиця 2.1 – Порівняльний аналіз стандартів і методик тестування

Методика / Характеристика	OSSTMM	ISSAF	OWASP	PTES	NIST SP 800-115	BSI	PETA	PTF
Рекомендації щодо обговорення з замовником цілей і завдань тестування	+	+	+	+	±	+	-	-
Рекомендації з підготовки договору на тестування	+	+	-	±	±	±	-	-
Законодавчі аспекти тестування	±	+	-	-	+	+	-	-
Рекомендації щодо збору інформації про об'єкт тестування	+	+	+	+	+	+	±	+
Детальні рекомендації з аналізу та оцінки вразливостей	±	+	+	+	±	+	-	+
Рекомендації щодо етапів тестування та їх змісту	+	+	+	+	+	+	+	+
Окремі рекомендації з тестування телекомунікаційних мереж	+	+	±	+	+	-	-	+
Окремі рекомендації з тестування бездротових мереж	+	+	-	+	+	-	-	+
Окремі рекомендації з тестування веб-додатків	±	±	+	+	-	-	-	±
Окремі рекомендації щодо перевірки безпеки фізичної інфраструктури	+	+	-	+	-	-	-	+
Окремі рекомендації з перевірки безпеки паролів	-	+	±	+	+	-	-	+
Окремі рекомендації щодо перевірки безпеки баз даних	-	+	±	-	-	-	-	-
Окремі рекомендації з перевірки безпеки вихідного коду програм	-	+	±	-	-	-	-	-

Продовження таблиці 2.1. – Порівняльний аналіз стандартів і методик тестування

Методика / Характеристика	OSSTMM	ISSAF	OWASP	PTES	NIST SP 800-115	BSI	PETA	PTF
Детальні рекомендації щодо використання конкретних ІТВ для тестування	-	+	+	+	±	+	-	+
Детальні рекомендації щодо використання конкретних ІПВ та соціальної інженерії для тестування	±	+	-	+	±	-	-	-
Рекомендації щодо конкретного ПЗ, що використовується для тестування	-	+	±	+	-	±	-	±
Рекомендації щодо формування звіту про тестування	+	+	+	+	-	+	-	+
Аналіз і рекомендації щодо усунення знайдених вразливостей	-	+	-	-	+	+	-	-

Де

«+» - є в повному обсязі;

«±» - мається на короткому викладі або згадується;

«-» - даний матеріал відсутній, або викладений таким чином, що не представляє цінності для аудитора.

Найбільш опрацьованою методикою тестування на проникнення як в теоретичному, так і в практичному плані є методика ISSAF. Методики OSSTMM, PETA і стандарти NIST SP 800-115, BSI носять більшою мірою теоретичний характер, при цьому NIST SP 800-115 і BSI фактично є стандартами країн-розробників, яких необхідно дотримуватися, проводячи

тестування на проникнення в цих країнах. Стандарт PTES і методика PTF є практико-орієнтованими і містять широкий набір технічних рекомендацій і конкретних вразливостей, які необхідно перевіряти в ході тестування на проникнення.

3 РОЗРОБКА МЕТОДИКИ ПОШУКУ ВРАЗЛИВОСТЕЙ У WEB-ЗАСТОСУНКАХ

Проаналізувавши існуючі методики та стандарти з галузі тестування безпеки додатків та визначивши їх слабкі та сильні сторони, пропонується наступна методика пошуку вразливостей у веб-застосунках. Наведена методика складається з семи етапів.

Першим етапом є етап планування. Мета цього етапу – визначити та спланувати тест на проникнення, визначивши область досліджень, зрозумівши правила взаємодії, дізнавшись про вимоги до тестування та визначивши інструменти, необхідні відповідно до цілей та дат.

Другим етапом методики є етап початкового збору інформації. На цьому етапі тестер повинен розглянути мету та правила взаємодії та зібрати основну, але конкретну інформацію про цілі. Для цього випробувач може використовувати будь-який тип підходу до збору інформації про цілі, щоб виявити додаткову інформацію про них і механізмах захисту, які можуть бути пов'язані з ними.

Третій етап методики – це етап сканування. На цьому етапі тестувальник повинен відігравати більш активну роль у процесі збору інформації, беручи до уваги інформацію, розкриту попередніми цілями, щоб почати пошук нових вразливостей. Тестер повинен просканувати додаток на наявність точок входу та недоліків під час цього процесу, а потім протестувати та підтвердити отримані результати. На цьому етапі тестувальник починає відігравати більш активну роль у тестуванні, маючи справу безпосередньо з додатком та його послугами.

Четвертим етапом методики є етап аналізу вразливості. На цьому етапі тестер зосередить свої зусилля на виявленні та оцінці результатів та

визначенні шляху виконання для експлуатації та тестування виявлених вразливостей. Для цього етапу методології немає вимог до програмного забезпечення, оскільки це потрібно робити переважно за допомогою оглядів процесів, ручного контролю та публічних досліджень.

П'ятим етапом є використання вразливостей. Основна мета цього етапу – встановити доступ до цілей в обхід обмежень безпеки. Якщо всі попередні етапи були виконані правильно, цей етап повинен бути досить простим, оскільки більшість точок входу встановлені, підхід і взаємодія сплановані, і вся додаткова інформація була розглянута для застосування корисних навантажень і експлойтів. Якщо вектор атаки добре визначений, то буде велика ймовірність успіху відповідно до атаки певних цілей. Цей етап може призвести до циклу між етапом експлуатації та етапом аналізу вразливостей, оскільки нові вразливості можуть бути виявлені під час виконання експлуатації попередніх результатів або під час подальшого проникнення та підвищення дозволів.

Шостий етап методики – це етап аналізу результатів. На цьому етапі тестувальник повинен проаналізувати весь процес, щоб перевірити, чи дотримувалися всі правила взаємодії, чи були досягнуті встановлені цілі, чи застосовні всі показники, а також проаналізувати загальний вплив тесту на проникнення на додаток. Цей етап розділений на три основні завдання: огляд встановлених правил взаємодії, цілей та показників, аналіз впливу тесту на проникнення та аналіз процедури тестування на проникнення в цілому.

Останнім етапом методики є етап звітність. Етап звітності – це кінцевий продукт тесту на проникнення. Після завершення всіх попередніх кроків і стадій тестувальник повинен підготувати звіт, в якому він представить замовнику всю інформацію про своєму підході, використовуваних інструментах і методах, короткий виклад результатів і пов'язаних з ними ризиків, вплив експлуатації сторонньої стороною, запобігання заходи, які необхідно вжити для усунення недоліків безпеки, вся технічна інформація, що

стосується тесту (з докладним описом області застосування, результатів, шляхів атаки і пов'язаних з ними ризиків), а також всі обмеження, виявлені в ході тестів. Підсумковий звіт повинен бути добре написаний, інформативний, легкий для розуміння і привабливий для виконавчого керівництва і технічного персоналу. Відповідно до цієї методології, остаточний звіт повинен бути розділений на три розділи: резюме, технічний звіт та додатковий розділ для отримання додаткової інформації (цей останній розділ не має вирішального значення для кінцевого продукту).

3.1 Планування

Мета цього етапу – визначити та спланувати тест на проникнення, визначивши сферу застосування, зрозумівши правила взаємодії, дізнавшись більше про вимоги до тестування та визначити інструменти, необхідні відповідно до цілей та дат. Етап планування складається з семи пунктів.

Перш за все має бути підписана угода про нерозголошення. Договір (угода) про нерозголошення – договір, укладеним між двома або більше сторонами з метою нерозголошення (зберігання), взаємного обміну знаннями, матеріалами, або іншою інформацією доступ, до якої обмежено третім особам, та (або) з метою зберігання (нерозголошення) конфіденційної інформації (комерційної таємниці) відносно третіх осіб. Цей документ зазвичай підписується до того, як відбудеться будь-яке обговорення щодо області досліджень, для захисту конфіденційності компанії.

3.1.1 Визначення області досліджень

Визначення області досліджень – це зустріч, що проводиться між тестувальниками та замовником, на якій слід визначити обсяг тестування та надати додаткову інформацію про систему та внутрішні процедури.

Це одна з найбільш важливих частин етапу планування, оскільки саме на ній визначається, що має бути протестовано. Цілі для пентесту повинні бути вказані, а також діапазони та домени, інформація про цілі або технічні деталі

(якщо це дозволено) та додаткова інформація, така як обмеження та вимоги. Тестувальник повинен враховувати складність встановлених вимог до тестування при визначенні області досліджень. Обсяг інформації, що розкривається безпосередньо пов'язаний з підходом до тестування програми.

3.1.2 Екстрені контакти

Дуже важливо встановити екстрені контакти, оскільки трапляються помилки, і краще перестрахуватися, ніж потім шкодувати. Для екстрених контактів Тестувальник повинен підготувати список з контактами, графіками доступності та формами для безпечної передачі даних у разі потреби.

3.1.3 Правила взаємодії

В області досліджень визначено, що буде тестуватися, і в цій частині визначено, як має відбуватися тестування. Для цього обидві сторони повинні прийняти рішення про процедуру обробки даних, терміни проведення цільових тестів, дозволи на тестування та обмеження підходу (конкретні методи, які не слід застосовувати). Цей набір правил гарантує, що система клієнта не піддається непотрібним ризикам в результаті дій тестувальника, і гарантує, що експлуатація і компрометація програми здійснюються відповідно до потреб клієнта.

3.1.4 Визначення показників якості

Результати вимірювання та процедури мають вирішальне значення для результатів тесту на проникнення. Вимірювання допоможуть тестувальнику і замовнику визначити і зрозуміти, який прогрес досягається в ході тестування безпеки. За допомогою певних показників тестер може пізніше використовувати цю інформацію та генерувати дані про тенденції, які можуть допомогти йому краще оцінити часові рамки для тестів. Зазвичай показники визначають відсоток охоплення цільових показників, відсоток хибнопозитивних вразливостей, вразливості кожної цілі, час, витрачений на виявлення вразливостей цілі, серйозність вразливостей.

3.1.5 Оцінка термінів

Після наради з визначення області застосування тестувальник повинен розглянути всю інформацію, щоб прийняти рішення про оцінку термінів для кожного завдання. Оцінка часу безпосередньо пов'язана з досвідом і знаннями тестувальника, оскільки цей досвід дозволить тестувальнику витрачати менше часу на виконання конкретних завдань, з якими йому може бути зручніше. Визначаючи оцінку часу, Тестувальник повинен бути обачним і розраховувати на більше часу, ніж він очікує; якщо під час тестів трапиться якась помилка, у нього все одно буде запас на додатковий час.

Визначення початкової та кінцевої дати тестувань не тільки захищає тестера від порушення встановлених термінів, але й захищає клієнта від затримок результатів. Певні дати повинні враховувати оцінки часу, встановлені раніше.

3.1.6 Вимоги безпеки

Вимоги безпеки визначають, як додаток працює з точки зору безпеки. Це означає розуміння управління користувачами, процедур автентифікації, процедур авторизації, конфіденційності даних та управління сеансами. Якщо ця інформація буде надана, це полегшить процес доступу; якщо ні, тестер повинен знайти спосіб виявити недоліки в механізмах безпеки.

3.1.7 Визначення інструментів

При прийнятті рішень щодо інструментів, які буде використовувати Тестувальник, він повинен враховувати область застосування, правила взаємодії, свої обмеження і користувальницький досвід. Для конкретних завдань можуть знадобитися спеціальні інструменти, тому з цієї причини тестер завжди повинен мати можливість використовувати автоматичні сканери та експоіти або використовувати ручну перевірку та індивідуальні експоіти. Якщо з боку замовника немає обмежень, ця частина етапу планування повністю знаходиться на виборі тестувальника.

3.2 Початковий збір інформації

3.2.1 Ідентифікація цілей

Незважаючи на те, що більша частина інформації повинна бути розкрита під час наради з визначення області дослідження, тестувальник завжди повинен визначати свої цілі. Це може допомогти визначити цілі, які не входили до початкової сфери, або навіть знайти додаткову інформацію про використовувані послуги чи протоколи. Щоб визначити свої цілі, тестувальник повинен розглянути архітектуру системи.

3.2.2 Підхід

Для отримання інформації щодо цілей випробувач може використовувати три підходи: пасивний, напівпасивний і активний підхід.

3.2.2.1 Пасивний підхід

При пасивному підході до збору інформації тестувальник може тільки збирати і використовувати архівовану або збережену інформацію. Жоден трафік не повинен спрямовуватися до цілі, оскільки передбачається, що він збирає інформацію, не виявляючи її. Це не повинно бути загальною процедурою, але конкретні завдання або клієнти можуть вимагати такого підходу.

3.2.2.2 Напівпасивний підхід

При зборі інформації з напівпасивним підходом мета тестувальника полягає в тому, щоб зібрати інформацію для профілювання цілі без шкоди для себе, посилаючи те, що може здатися звичайним трафіком або поведінкою. Тестер не повинен активно шукати конфіденційну інформацію, але повинен публікувати деталі, які можуть допомогти визначити цілі.

3.2.2.3 Активний підхід

Активний підхід полягає в тому, що тестер активно виявляє структуру мережі, сканує порти, відкриті служби, неопубліковані каталоги, файли та

сервери та, якщо можливо, збирає інформацію про системи баз даних та служби, що працюють у фоновому режимі. Мета цього підходу полягає в тому, щоб якомога більше перерахувати і нанести на карту цільові показники.

3.3 Сканування

Третій етап методики – це етап сканування. На цьому етапі тестувальник повинен просканувати додаток на наявність точок входу та недоліків під час цього процесу, а потім протестувати та підтвердити отримані результати. Це можна зробити за допомогою автоматичних або ручних інструментів для виявлення та оцінки безпеки щодо можливих вразливостей.

3.3.1 Активне сканування

3.3.1.1 Сканування мережі

Сканування мережі в основному здійснюється за допомогою автоматизованих інструментів, які допомагають тестувальнику отримати загальне уявлення про те, що може бути доступним у цільовій мережі. Цей крок до сканування програми зазвичай починається на етапі збору інформації, при виконанні сканування порту; однак на цьому етапі передбачається провести більш глибокий аналіз мережі, пошук неправильних налаштувань в мережі, вразливостей в технологіях передачі, тестування вразливостей використовуваних сервісів і протоколів.

3.3.1.2 Сканування додатка

Загальне сканування програми – це метод, який використовується для пошуку загальних недоліків у програмі. За допомогою ручного або автоматичного обходу можна знайти недоліки в додатку. Під час виконання сканування тестер може виявити поля форми для спроби ін’єкції SQL або XSS. При обході програми також можна знайти конфіденційну. Найбільш підходящими інструментами, що використовуються для цього, є BurpSuite [17] та OWASP ZAP [18], оскільки вони мають вбудовані сканери, які сканують програму на наявність вразливостей.

3.3.1.3 Ідентифікація вразливостей

Виконуючи попередні процедури активного сканування програми, тестер може зіткнутися з новими вразливими місцями в цій області. Прикладом такої ситуації є те, коли тестер виконує загальне сканування програми та виявляє поля форми, вразливі до ін’єкції SQL. Усі виявлені вразливості повинні бути проаналізовані та ідентифіковані, щоб використовувати їх на пізніх стадіях. Для цього тестер повинен зрозуміти, як була виявлена вразливість, як працює вразливість та який вплив може мати її використання.

3.3.2 Пасивне сканування

Пасивне сканування - це сканування, застосовуючи пасивний підхід.

В основному це робиться за допомогою комбінації автоматичних та ручних інструментів, оскільки автоматичні інструменти прагнуть знайти та зібрати інформацію, а ручні інструменти допоможуть тестувальнику проаналізувати та оцінити результати.

3.3.2.1 Моніторинг трафіку

Для пасивного сканування програми тестер може відстежувати трафік, що може допомогти визначити специфіку операційної системи або пристрою. Тестер може виявити неправильні налаштування, незахищену передачу даних або навіть захопити конфіденційну інформацію під час цього процесу. Найпоширенішими інструментами, що використовуються для цього, є Wireshark та Tcpdump.

3.3.2.2 Аналіз метаданих

Аналіз метаданих складається з аналізу та оцінки даних, які описують дані, а не самі дані. Наприклад, під час перевірки файлу інформація про метадані може містити такі відомості, як автор документа, коли документ був створений, та іншу інформацію, яка може включати метадані користувачів. У

цих метаданих можна знайти внутрішні адреси та шляхи до сервера, IP-адреси та іншу інформацію, що полегшує тестерам отримання додаткового доступу або інформації для виявлення нових точок входу. В основному це робиться вручну, але тестерам доступні такі інструменти, як FOCA [19] або MetaCrawler [20].

3.3.3 Тестування

Тестування на вразливість – це процес виявлення вразливостей та недоліків безпеки, пов'язаних з досліджуваним додатком. Це може бути зроблено під час виконання активного сканування програми і завжди має бути адаптовано до вимог тестування, правил взаємодії та кінцевої мети. Для цієї процедури тестер вивчить вразливості в межах встановленої глибини та широти вимог, гарантуючи, що результати оцінки відповідають очікуванням щодо цього. Це потрібно зробити за допомогою активних методів та інструментів, які допомагають тестувальнику перевірити справжність результатів. Для цієї процедури зазвичай використовуються такі інструменти, як OWASP ZAP або Burp Suite.

3.3.4 Перевірка коду

На етапі сканування також важливо вручну перевірити та переглянути код, пов'язаний із додатком, вихідний код або аналіз запитів, зроблених до програми.

Залежно від підходу до тестування, тестувальнику може бути дозволено переглядати вихідний код програми (у випадку тестування "білого ящика"), що, незважаючи на те, що це може бути обширна процедура, допоможе тестувальнику краще зрозуміти робочий процес програми. Перевірка також можлива за допомогою автоматичних інструментів, але більшість тестувальників погодяться, що ручний контроль нічим не замінити.

3.3.4.1 Аналіз вихідного коду

Аналіз вихідного коду – це процес, за допомогою якого тестер перевіряє вихідний код програми, шукаючи проблеми безпеки, оскільки багато недоліків безпеки можуть бути не виявлені сканерами вразливостей. Будь-яка інформація, пов'язана з недоліками безпеки, завжди міститься у вихідному коді. Маючи доступ до вихідного коду, тестувальник може точно визначити всі процедури всередині програми та виявити можливі недоліки в області безпеки.

Аналіз вихідного коду може бути складною процедурою, оскільки деякі перевірки повинні виконуватися вручну. Однак за допомогою таких інструментів, як SonarQube [21] або Appscan [22], цей процес може бути полегшений, оскільки ці інструменти аналізують та переглядають код, шукаючи помилки кодування, вразливості безпеки, пропонуючи виправлення для забезпечення стабільності та безпеки коду.

3.3.4.2 Аналіз запитів

Іншим способом пошуку конфіденційної інформації або вразливостей є спостереження, аналіз і маніпулювання запитами, що відправляються між додатком і сервером. HTTP-запити GET часто можуть розкривати конфіденційні дані, які можуть бути (або не бути) зашифровані, що дозволяє тестувальнику відкривати нові підходи до додатку. Цей аналіз можна здійснити за допомогою Burp Suite або за допомогою OWASP ZAP, оскільки обидва інструмента мають спеціальний проксі-сервер для перехоплення запитів, а також плагіни, які допомагають тестувальнику перетворювати закодовані або необроблені дані в корисну інформацію.

3.3.5 Підтвердження результатів

Цей крок слід враховувати при проведенні більшості тестів на проникнення, оскільки для отримання результатів дуже важливо перевірити кожне введення та виведення виконаних завдань та їх результатів. Ця

процедура має вирішальне значення для того, щоб допомогти тестувальнику зрозуміти кількість справжніх вразливостей. Це може бути зроблено або шляхом тестування вразливостей, або шляхом перевірки відповідного недоліку.

3.4 Аналіз вразливостей

3.4.1 Аналіз загроз

Після виконання всіх попередніх процедур сканування тестувальник повинен провести своє дослідження всіх результатів. Роблячи це, тестер отримає краще уявлення про те, як проводити атаки, який вплив вони можуть мати та обмеження в рамках правил взаємодії та певної області досліджень. Мета аналізу загроз полягає в тому, щоб дозволити тестувальнику визначити потенційні вектори атаки щодо кінцевої мети тестів і розглянути її вплив. Інтернет-платформи, такі як NVD (національна база даних про вразливості) [23], OSVDB (База даних про вразливості з відкритим кодом) [24], рекомендації щодо безпеки та засоби відстеження проблем – це чудові бази даних про вразливості, які дозволяють тестувальникам переглядати та знаходити більше інформації про проаналізовані вразливості. CVE також є чудовим джерелом інформації про вразливості, виявлені відповідно до системних компонентів, типів вразливостей та номерів CVE, які роблять дослідження більш точним, оскільки вони представляють ідентифікатор конкретних вразливостей.

3.4.2 Можливість експлуатації вразливостей

Виконуючи аналіз вразливостей, Тестувальник повинен також враховувати свої можливості щодо отримання або розробки експлойтів або корисних навантажень, необхідних для тестування середовища. Не тільки шляхом аналізу наявності та доступності таких експлойтів або корисного навантаження, а й шляхом розгляду використання третіми сторонами та налаштування конкретних методів. Цей крок має вирішальне значення для

планування етапу експлуатації, оскільки це допоможе тестувальнику підтвердити свої висновки та краще зрозуміти, як скористатися певними недоліками додатку. Що стосується можливих експлойтів, то широко використовуваним програмним забезпеченням є Metasploit, який містить колекцію з постійно зростаючою кількістю експлойтів, доступних для використання будь-яким тестувальником.

3.4.3 Аналіз доступності

Аналіз доступності полягає в доступі до конкретних вразливостей або точок входу для визначення точного сценарію і шляху виконання експлойту. Враховуючи механізми захисту та робочий процес цільових об'єктів, тестувальник повинен визначити всі вимоги та потреби для доступу до вразливості.

3.4.4 Планування атак

Беручи до уваги всі попередні завдання, планування наступних кроків повинно мати чіткий результат. Розгляд сфери досліджень і правил взаємодії, а також всіх результатів дозволяє тестувальнику визначити шлях виконання своїх експлойтів максимально інформативно, що деталізує цілі експлуатації, всі точки входу, як отримати до них доступ, які корисні навантаження або експлойти використовувати, інструменти і методи для використання і як налаштувати інструменти з урахуванням встановлених цілей.

3.5 Використання вразливостей

3.5.1 Експлойти

При виконанні експлойтів вектори атаки повинні бути точними і непоміченими. Це означає, що весь процес спрямований на імітацію реальної атаки. Етап експлуатації – це застосування всіх накопичених досліджень, проведених на цільовому застосунку, у поєднанні з заходами ухилення, щоб тестер міг виконувати точні атаки на ціль, не виявляючись під час тесту на проникнення.

В основному існує два типи експлойтів: загальнодоступні та спеціально розроблені експлойти; обидва мають свою придатність і повинні використовуватися відповідно до потреб тестувальника. Існує безліч інструментів для виконання подвигів, найпопулярнішими є Burp Suite, OWASP ZAP, Metasploit [25].

3.5.1.1 Загальнодоступні експлойти

Щоб максимізувати шанси на успішне використання вразливості цілі, Тестувальник захоче отримати доступ до якомога більшої кількості ресурсів. Зібравши всю інформацію, виконану на попередніх етапах, тестер може шукати загальнодоступні екплойти, доступні в інтернеті або в інших типах документації. Ці екплойти розробляються відповідно до конкретних вразливостей у конкретних апаратних або програмних конфігураціях, тому, якщо всі вимоги дотримані, вони можуть бути застосовані тестувальником. На цьому етапі Тестувальник повинен бути обережним з точки зору відповідальності та джерела вразливості, оскільки існує кілька підроблених екплойтів, призначених для заподіяння шкоди або знищення комп'ютера користувача. Деякі бази даних експлойтів, які вважаються надійними – це Exploit-DB, Searchsploit, Metasploit і навіть Google [26].

Оскільки кожна атака, як правило, відрізняється за способом використання, для досягнення успішних результатів тестеру може знадобитися адаптувати та налаштувати використання відповідно до сценарію тестування. Чітко розуміючи середовище тестування та застосовність експлуатації, тестувальник збільшить свої шанси на успішну атаку. Оскільки загальнодоступні експлойти можуть бути занадто специфічними для певних версій операційних систем або додатків, існує необхідність змінювати та налаштовувати експлойти для їх виконання. Цей процес може вимагати моделювання середовища для перевірки змін та гарантії його результатів.

Навіть якщо у тестувальника в руках вся зібрана інформація про систему, наявність робочої інфраструктури і системи для тестування експлойтів, це не спростить процес тестування. Це виправдано змінами адрес пам'яті, заснованими на пакетах оновлень або випусках нових версій. Для виконання цього завдання тестеру потрібно буде знати, як читати код, розроблений різними мовами програмування, і розуміти, як працюють корисні навантаження для кожної експлуатації, щоб адаптувати їх відповідно до своїх потреб.

3.5.2 Подальше проникнення до системи

Після виконання успішних використання експлойтів і залежно від правил взаємодії та обсягу тесту тестувальник може додатково перераховувати та отримувати доступ до інших систем в інфраструктурі клієнта. Використовуючи доступ, який був наданий під час експлуатації вразливостей, тестувальник може мати можливість виконувати дії всередині скомпрометованої системи, що дозволяє йому завантажувати інструменти у систему, перераховувати DNS внутрішньої мережі, виконувати атаки методом перебору, виконувати віддалені експлойти і зловживати скомпрометованими обліковими даними. Тестер також може використовувати скомпрометовану систему для підключення до внутрішньої мережі, налаштовувати переадресації портів, обмежувати доступ до інформації та навіть виконувати маніпулювання автентифікацією.

3.5.2.1 Нові вразливості

При використанні вразливостей або при подальшому проникненні в систему Тестувальник може виявити нові вразливості для тестування в міру збільшення дозволів і надання доступу до різних частин системи. Ці висновки потребують аналізу, щоб оцінити їх. Цей процес призведе до циклу між четвертим і п'ятим етапами цієї методики, оскільки були виявлені нові вразливості, Тестувальник повинен оцінити ризик і вплив результатів.

3.5.3 Встановлення бекдорів

Можливість змінювати конфігурації в компрометованих системах і маніпулювати ними дозволяє тестувальникам встановлювати бекдори для того, щоб залишатись в системі. Встановлені бекдори повинні вимагати автентифікації, для запобігання несанкціонованим атакам. Бекдори дозволяють тестувальникам обійти звичайний процес аутентифікації після компрометації системи. Це може бути зроблено для підтримки або полегшення доступу в майбутньому або для подальшої експлуатації системи. Щоб встановити бекдор, тестер може використовувати інструменти, призначені для цієї мети, або використовувати прості інструменти, такі як NetCat [27], основною метою якого є читання та запис даних через мережеві підключення. Завдяки цьому тестувальник дозволяє собі зберігати свій доступ до системи, навіть якщо він з якоїсь причини відключається.

3.5.4 Очищення

Перед виконанням очищення системи тестувальник повинен переконатися, що всі етапи експлуатації задокументовані і що в системі більше немає тестів для виконання. Очищення – це процедура, при якій тестер очищає систему від будь-яких слідів тесту на проникнення. Це включає в себе видалення всіх виконуваних файлів, повернення всіх системних налаштувань і конфігурацій до їх початкових значень, видалення всіх встановлених бекдорів, видалення облікових записів користувачів, створених для підключення до системи, і відновлення бази даних з резервної копії, щоб гарантувати, що дані не були пошкоджені під час процесу.

3.6 Аналіз результатів

3.6.1 Огляд отриманих результатів

Щоб оцінити відповідність раніше встановленим вимогам і правилам, їх необхідно переглянути, щоб підтвердити, чи пройшла вся процедура

відповідним чином. Для цього огляду тестувальнику потрібно перевірити, чи відповідає весь підхід до тестування на проникнення очікуваному.

3.6.1.1 Огляд правил взаємодії

Оскільки правила взаємодії встановлені для захисту обох сторін (тестувальника та замовника), тест на проникнення повинен оцінюватися відповідно до визначеного. Тестер повинен оцінити всі процедури, проведені під час тесту на проникнення, щоб перевірити, чи кожна дія була виконана належним чином. Після їх розгляду Тестувальник повинен повідомити результат аналізу у звіті на етапі звітності.

3.6.1.2 Огляд цілей

Весь процес тестування на проникнення був проведений з певної причини, і це визначається раніше встановленими цілями. Тестер повинен оцінити результати тесту на проникнення та підтвердити, чи відповідають вони певним цілям. Ця інформація має бути присутня у звіті, розробленому на етапі складання звітності.

3.6.1.3 Огляд показників якості

Огляд показників якості та їх кореляція з результатами має важливе значення для вимірювання ефективності тесту на проникнення. Припустимо, що показники якості тесту добре визначені і встановлені. У цьому випадку обидві сторони матимуть більш чітке та стисле уявлення про результати, не лише для того, щоб клієнт міг оцінити роботу, виконану тестувальником, а й для того, щоб тестер міг оцінити, чи пройшло дослідження відповідно до очікувань. Ця інформація має бути присутня у звіті, розробленому на етапі складання звітності.

3.6.2 Аналіз впливу тестування на систему

Після виконання всіх тестів та оцінок безпеки та очищення всіх слідів тесту на проникнення тестувальник повинен проаналізувати та оцінити вплив

всього процесу тестування на додаток. Якщо все зроблено правильно, це не матиме ніякого впливу на систему, але тестувальник повинен провести цю оцінку, щоб гарантувати цілісність системи та використовуваних методів.

3.6.3 Аналіз тесту на проникнення

Враховуючи все, етичний тестер повинен оцінити всі процедури, проведені під час тесту на проникнення. Що відповідало очікуванням, що не відповідало, які інструменти найкраще підходили для конкретних завдань, і вся додаткова інформація повинна бути проаналізована. В основному це процедура для тестувальника, що дозволяє оцінити його підхід і навички. Самооцінка є найважливішим компонентом вдосконалення і саморозвитку. Ця інформація може бути розкрита в остаточному звіті, але вона не є обов'язковою.

3.7 Звітність

3.7.1 Загальний звіт

У звіті підсумовуються загальні висновки оцінки. Мета цього розділу – ознайомити бізнес-менеджерів і власників систем про виявлені вразливості. Мова цього розділу не повинна бути надто технічною. У цьому розділі тестувальник також повинен включити інформацію про обсяг, терміни проведення тестів, список цілей, виявлені обмеження, звіт результатів.

Звіт складається з шести основних підрозділів:

1. Цілі проекту. У цьому підрозділі звіту тестувальник повинен окреслити всі цілі тесту на проникнення, а також використовувані показники та їх значення відповідно до результатів та очікуваного результату тестів;
2. Обсяг проекту. У цьому підрозділі звіту тестувальник повинен описати узгоджений обсяг тестів і те, як він дотримувався його в процесі тестування;

3. Терміни. У цьому підрозділі звіту тестувальник повинен вказати часові рамки випробувань та дати початку та завершення досліджень;
4. Список цілей. У цьому підрозділі звіту тестувальник повинен перерахувати цілі тестів та будь-яку додаткову інформацію, яка може мати відношення до теми;
5. Обмеження. У цьому підрозділі звіту тестувальник повинен описати всі обмеження, виявлені під час тестів. Наприклад, обмеження з точки зору методів, технічні проблеми, продуктивність, обмеженість інструментів та будь-яка додаткова інформація, що стосується цієї теми;
6. Висновки. У цьому підрозділі звіту тестувальник повинен описати кожну вразливість, виявлену під час тестів, з інформацією про ризики для системи.

3.7.2 Технічний звіт

Цей розділ звіту повинен бути чітким, стислим та більш технічним, ніж загальний звіт, і містити всю необхідну інформацію, щоб технічні групи могли зрозуміти недоліки безпеки, а також ступінь тяжкості кожної виявленої вразливості. Щоб дати краще розуміння, можуть бути включені скріншоти і командні рядки, що показують кроки, зроблені під час пошуку вразливостей.

3.7.2.1 Висновки

У цьому підрозділі тестувальник повинен включити детальну інформацію про свої висновки, включаючи всю необхідну інформацію для технічної команди, щоб зрозуміти та відтворити тестовий сценарій. Він також повинен включати рейтинг серйозності для всіх виявлених вразливостей та технічний опис проблеми та постраждалих об'єктів. Скріншоти і командні рядки можуть бути представлені для того, щоб дати краще уявлення про дослідження.

3.7.3 Додаткова інформація

Цей заключний розділ звіту не є обов'язковим. Однак тестер може захотіти додати додаткову інформацію, що стосується тестів, яка може бути самооцінкою процесу або будь-якою інформацією, що стосується раніше встановлених точок значущості.

3.8 Висновок

Ця методика була визначена і конкретизована з урахуванням деяких найбільш часто використовуваних методик та стандартів в галузі тестування веб-додатків. Ця методика може розглядатися як незавершена, оскільки вона вимагає подальшої перевірки у більшій кількості тестових випадків, щоб вважатися застосовною. Рекомендовані інструменти в основному з відкритим кодом, щоб методика була максимально доступною та всеохоплюючою, але деякі комерційні інструменти згадуються та рекомендуються [28].

У таблиці 3.1 представлено загальне уявлення щодо розробленої методики пошуку вразливостей у веб-застосунках та інструментів, які повинні використовуватись на кожному з етапів.

Таблиця 3.1 – Загальне уявлення розробленої методики пошуку вразливостей.

Етап методики	Підрозділ	Вхідні дані	Інструмент або метод для виконання	Результат
Планування	Визначення області досліджень	Цілі; Об’єкти досліджень; Задачі.	Обговорення з замовником	Встановлена область досліджень; Угода про нерозголошення
	Екстрені контакти	Контакти замовника; Лінії зв’язку	Обговорення з замовником	Екстрені контакти
	Правила взаємодії	Правила; Обмеження; Цілі	Обговорення з замовником	Правила взаємодії
	Визначення показників якості	Задачі; Цілі	Обговорення з замовником	Показники якості тестування

Продовження таблиці 3.1 – Загальне уявлення розробленої методики пошуку вразливостей.

Етап методики	Підрозділ	Вхідні дані	Інструмент або метод для виконання	Результат
Планування	Оцінка термінів	Область дослідження; Правила взаємодії	Експертне судження тестувальника	Терміни виконання дослідження; Дата початку; Дата завершення;
	Вимоги безпеки	Технічна документація;	Аналіз; Обговорення з замовником	Вимоги безпеки
	Визначення інструментів	Цілі; Задачі; Обмеження; Вимоги безпеки	Експертне судження тестувальника	Набір інструментів для пошуку вразливостей
Початковий збір інормації	Ідентифікація цілей	Область досліджень; Технічна документація	Файловий провідник; Браузер; Командна строка	Визначені цілі тестування
	Підхід	Технічна документація; Цілі; Задачі; Вимоги безпеки	Файловий провідник; BurpSuite; Wireshark	Пасивний, напівпасивний або активний підхід до збору іфнормації
Сканування	Активне сканування	Підключення до мережі Інтернет; Веб-застосунок	Nmap; OWASP ZAP; BurpSuite; Metasploit	Інформація щодо мережі; Карта веб-застосунку; Перелік вразливостей
	Пасивне сканування	Веб-застосунок	FOCA; MetaCrawler; Wireshark	Метадані; Перехоплений трафік
	Тестування	Веб-застосунок	OWASP ZAP; BurpSuite;	Перелік вразливостей
	Перевірка коду	Код веб-застосунку	OWASP ZAP; WebInspect; Ручна перевірка	Перелік вразливостей; Конфіденційні дані
	Підтвердження результатів	Перелік вразливостей	Metasploit; Ручна перевірка	Перелік підтверджених вразливостей

Продовження таблиці 3.1 – Загальне уявлення розробленої методики пошуку вразливостей.

Етап методики	Підрозділ	Вхідні дані	Інструмент або метод для виконання	Результат
Аналіз вразливостей	Аналіз загроз	Перелік підтверджених вразливостей	NVD; CVE; OSVDB	Інормація щодо загроз; Ризики; Потенційні вектори атак
	Можливість експлуатації вразливостей	Перелік підтверджених вразливостей	Metasploit; Пошук в Інтернеті	Перелік експолойтів
	Аналіз доступності	Перелік підтверджених вразливостей; Інормація щодо загроз	Експертне судження тестувальника; Пошук в Інтернеті	Можливості щодо проведення атак
	Планування атак	Перелік підтверджених вразливостей; Перелік експолойтів; Потенційні вектори атак; Набір інструментів для пошуку вразливостей; Цілі	Експертне судження тестувальника	План використання виявлених вразливостей для атак.
Використання вразливостей	Експлойти	Веб-зпстосунок; Вразливості; Загальнодоступні та приватні експлойти	BurpSuite; OWASP ZAP; Metasploit; Nmap; Exploit-DB; Searchploit; Google	Недоліки безпеки; Справжні вразливості
	Подальше проникнення до системи	Результати підрозділу «Експлойти»	Експертне судження тестувальника	Нові вразливості
	Встановлення бекдорів	Атакований веб-застосунок	NetCat	Встановлені на цілях бекдори
	Очищення	Атакована система; Експлойти; Бекдори; Налаштування системи	Ручна перевірка; Бекап системи	Відновлений веб-застосунок, без слідів атак

Продовження таблиці 3.1 – Загальне уявлення розробленої методики пошуку вразливостей.

Етап методики	Підрозділ	Вхідні дані	Інструмент або метод для виконання	Результат
Аналіз результатів	Огляд отриманих результатів	Правила взаємодії; Цілі; Показники якості	Ручна перевірка	Огляд правил взаємодії, цілей та показників якості
	Аналіз впливу тестування на систему	Перелік цілей; Результат використання вразливостей; Інформація щодо цілей	Ручна перевірка	Аналіз впливу тестування на систему
	Аналіз тесту на проникнення	Правила взаємодії; Показники якості; Цілі; Результати	Ручна перевірка	Аналіз тесту на проникнення
Звітність	Загальний звіт	Задачі проекту; Область дослідження; Терміни тесту; Перелік цілей; Результати	Текстовий редактор	Загальний звіт
	Технічний звіт	Знайдені вразливості; Недоліки безпеки; Технічна інформація	Текстовий редактор	Технічний звіт
	Додаткова інформація	Самооцінка проведених досліджень та тестувальника;	Текстовий редактор	Додаткова інформація

4 ПРИКЛАД ВИКОРИСТАННЯ РОЗРОБЛЕНОЇ МЕТОДИКИ

Для прикладу використання цієї методики було вирішено провести пентест до веб-застосунку DVWA.

У цьому розділі буде представлена додаткова інформація про налаштування середовища, опис DVWA та його інфраструктури, описані найбільш підходящі інструменти для цього процесу, проаналізовані різні методики тестування та продемонстровано застосування розробленої методики.

4.1 Середовище тестування

Налаштування середовища складалося з віртуальної машини з встановленою ОС Kali Linux. На етапі попередньої взаємодії було визначено, що сферою дії Pentest повинен бути лише певний порт встановлення програмного забезпечення з певною адресою для підключення, що вимагає доступу до віртуальної приватної мережі (VPN). Дозволи на використання програми надані, але з обмеженим доступом, що обмежувало Пряме підключення до бази даних через надані облікові дані. Повний доступ був наданий у веб-додатку, але не до всіх каталогів, включених до умов дозволів програми.

4.2 Відомості про додаток

DVWA (англ. Damn Vulnerable Web Application) – веб-додаток, заснований на PHP/MySQL, що має колосальну кількість вразливостей [29].

Головна мета існування такого додатка – допомогти спеціалістам з інформаційної безпеки оцінити свої навички, перевірити різні інструменти, не маючи проблем із законом; допомогти веб-розробникам краще зрозуміти механізм написання безпечного коду; а також дати можливість студентам та

викладачам дізнатися більше про безпеку веб-додатків у контрольованому середовищі [29].

DVWA надає можливість попрактикуватися в експлуатації найпопулярніших веб-вразливостей на різних рівнях складності. Програма має простий та зрозумілий інтерфейс. Також розробники програми попереджають, що всередині програми можлива наявність непередбачених та недокументованих вразливостей [29].

DVWA можна запустити як серед Windows, так і в Linux. Оскільки DVWA є PHP/MySQL веб-додатком, їй потрібна відповідна основа. Для Windows оптимальним рішенням є XAMPP, щоб встановити DVWA на Linux систему, можна також скористатися XAMPP. XAMPP - це серверна збірка для PHP розробки, що включає Apache, MariaDB, PHP, Perl. Також можливе використання MySQL. XAMPP є безкоштовним програмним забезпеченням. Ця збірка є доступною для операційних систем Windows, Linux, OS X [29].

4.3 Інструменти

Перша частина тестування полягає в тому, щоб зрозуміти, які найкращі інструменти для виконання оцінки вразливості та безпеки в межах доступних сканерів з відкритим кодом, включаючи деякі комерційні сканери. Вибрані інструменти були обрані на основі попередніх досліджень, в яких оцінюються (комерційні та з відкритим кодом) сканери вразливостей на основі п’яти критеріїв, серед яких: точність, відкликання, розрахунок індекса Юдена, веб-бенчмарк від OWASP та WASSEC [28]. Обраними інструментами стали Nmap, Metasploit, Burp Suite, OWASP ZAP, Wireshark та WebInspect. У PortSwigger було запропоновано безкоштовну ліцензію на Burp Suite Pro, але вона не була надана, в результаті чого була використана безкоштовна версія Burp Suite Community Edition.

4.3.1 Nmap

Nmap – це інструмент з відкритим кодом для сканування портів з ОС; його звичайне використання – виявлення мережі та аудит безпеки [30]. Nmap використовує необроблені IP-пакети для визначення доступних хостів, служб, операційних систем, можливих брандмауерів або фільтрів пакетів, серед багатьох інших специфікацій. Він був обраний завдяки популярності серед тестувальників безпеки та різноманітності утиліт, які дозволяють сканувати безліч різних компонентів та елементів, що належать до певної мережі, даючи фундаментальний результат під час сканування мережі.

Як описано організацією Nmap, він є гнучким, оскільки включає безліч механізмів сканування портів, потужним, оскільки він використовувався для сканування масивних мереж з тисячами комп'ютерів, портативним завдяки сумісності з усіма основними ОС, простим у використанні, безкоштовним, дозволяючи будь-якому користувачеві завантажувати його без фінансових витрат, добре задокументованим і переведений на кілька мов, підтримується великою спільнотою розробників і користувачів.

4.3.2 Metasploit

Metasploit – це дуже універсальний фреймворк, простий у налаштуванні та сумісний з більшістю операційних систем. Вбудований в Kali, цей фреймворк спростив для користувачів віддалене тестування безпеки систем. Цей інструмент дозволяє користувачам розробляти та виконувати експлойти за допомогою командного рядка або графічного інтерфейсу, використовуючи безліч експлойтів різного класу та широке середовище розробки експлойтів. Цей фреймворк розділений на модулі, що містять безліч сценаріїв, інструментів та плагінів з бібліотекою, яка дозволяє використовувати різні вразливості без необхідності писати додатковий код, визначаючи лише експлойт та його корисне навантаження [25].

4.3.3 Burp Suite Community Edition

Burp Suite – це платформа, яка використовується для тестування безпеки веб-додатків. Це інструмент, здатний відображати та аналізувати прикладне середовище. Community Edition включає деякі необхідні інструменти, такі як Repeater, Decoder, Sequencer, Comparer, Burp Intruder, HTTP та проксі WebSocket. Завдяки безлічі доступних плагінів і функціональних можливостей цей інструмент є одним з найбільш рекомендованих професійними тестувальниками [17].

Оскільки використання Burp Suite було обмежено випуском Community Edition, доступні такі функції:

Repeater – простий інструмент для ручного управління і перевидання окремих повідомлень HTTP і WebSocket з метою аналізу відповідей додатки.

Decoder – цей інструмент в основному використовується для перетворення закодованих або необроблених даних у їх канонічну, закодовану або хешовану форму шляхом розпізнавання різних форматів кодування.

Sequencer – цей інструмент дозволяє тестувальникам перевіряти непередбачувану інформацію, таку як маркери сеансу, маркери захисту від CSRF (підробки міжсайтових запитів), маркери скидання пароля та багато іншого.

Comparer – це простий інструмент, який використовується для порівняння будь-яких двох елементів даних.

Proxy – інструмент, за допомогою якого в Burp Suite можна керувати проксі-сервером між браузером і цільовим додатком, який може перехоплювати, перевіряти і змінювати необроблений трафік, що проходить в обох напрямках.

Intruder – це інструмент для автоматизованих індивідуальних атак. Він легко налаштовується для виконання різних завдань, від перебору веб-каталогів до активного використання складних вразливостей.

4.3.4 OWASP ZAP

OWASP ZAP – це сканер безпеки веб-додатків з відкритим кодом, який підтримується командою міжнародних волонтерів і є одним із найбільш часто використовуваних сканерів веб-додатків у світі [18]. Використовується як проксі-сервер, він дозволяє користувачам керувати спільним трафіком, який проходить через нього, що дозволяє зіставляти каталоги та ресурси веб-додатків через інструмент ZAP Spider. Цей інструмент простий у налаштуванні та використанні, він дозволяє знаходити та виявляти вразливості безпеки, пов'язані з впровадженням SQL, порушеною автентифікацією, розкриттям конфіденційних даних, порушеним контролем доступу, неправильною конфігурацією безпеки, міжсайтовими сценаріями, небезпечною десеріалізацією, компонентами з відомими вразливими місцями та відсутніми заголовками безпеки. Завдяки своїй сумісності, його можна налаштувати у всіх основних операційних системах.

4.3.5 Wireshark

Wireshark – програма-аналізатор трафіку для комп'ютерних мереж Ethernet і деяких інших. Має графічний користувальницький інтерфейс. Спочатку проект називався Ethereal, але через проблеми з торговою маркою в червні 2006 року проект був перейменований в Wireshark [31].

Функціональність, яку надає Wireshark, дуже схожа з можливостями програми tcpdump, проте Wireshark має графічний користувальницький інтерфейс і набагато більше можливостей по сортуванню і фільтрації інформації. Програма дозволяє користувачеві переглядати весь проходить по мережі трафік в режимі реального часу, переводячи мережеву карту в нерозбірливий режим (англ. promiscuous mode) [31].

4.3.6 WebInspect

WebInspect – це комерційний автоматизований інструмент тестування безпеки веб-додатків, який виявляє відомі та невідомі вразливості, включаючи вбудовування параметрів; міжсайтові сценарії; і обхід каталогів у веб-додатках [32].

WebInspect – це автоматизований і настроюваний інструмент тестування безпеки веб-додатків, який імітує реальні методи злому і атаки, дозволяючи ретельно аналізувати складні веб-додатки і служби на предмет вразливостей в системі безпеки. Надаючи можливість тестувати веб-додатки від розробки до виробництва, ефективно керувати результатами тестування та поширювати знання про безпеку по всій організації, WebInspect дає можливість захистити найбільш вразливі точки входу від атак.

Ключові переваги WebInspect

1) Тестування безпеки веб-додатків від розробки до виробництва;
2) Легко керуйте, переглядайте та діліться результатами тестів безпеки та історіями;
3) Забезпечити більш широке впровадження життєвого циклу за рахунок автоматизації безпеки;
4) Розширюйте знання з безпеки у всьому вашому бізнесі;
5) Доведіть відповідність правилам і політикам безпеки.

4.4 Використовувана методика

Розроблена методика, що використовується для цього прикладу, розділена на сім етапів, та є подібною до стандарту PTES. Весь процес тестування починається із зустрічі з замовником, на якій підписується угода про нерозголошення між тестувальником і клієнтом, що гарантує клієнту, що ніякі конфіденційні дані не будуть розкриті або передані в будь-якій формі. Після укладення угоди має бути визначена область дослідження тесту з точки зору обмежень інфраструктури, вимог, показників для оцінки безпеки та цілей

тесту (база даних, додаток, мережа, соціальна інженерія та інші). Додаткова інформація про ціль також повинна бути розкрита відповідно до сфери застосування та очікуваного підходу (якщо очікується, що тестер матиме підхід "чорної скриньки", жодна інформація не повинна розкриватися, якщо це підхід "білої скриньки", тестувальник також повинен мати доступ до всієї інформації в системі як його вихідний код, і якщо це підхід "сірої скриньки", слід розкривати лише відповідну інформацію), а також технічні деталі інфраструктури та інших відповідних процесів. За допомогою цієї інформації тестувальник повинен визначити оцінку часу для тесту та вирішити, які інструменти та методи використовувати. Наступним кроком цієї методики є етап початкового збору інформації, на якому тестувальник повинен вивчити більше інформації про програму, мережі і системи, в яких вона розміщена. Для цього за допомогою таких інструментів, як Nmap, Burp Suite або Metasploit, можна відобразити мережу, відкриті порти хоста та його запущені служби, зіставити деякі каталоги програм та виявити інформацію про використовувану базу даних, систему, в якій вона запущена, розкриті адреси, версії, екземпляр, ім'я сервера та порт TCP, на якому він запущений. Після збору базової інформації про цілі тестувальник повинен розпочати етап сканування, застосувавши більш активний підхід до програми. Використання автоматичних сканерів скорочує витрати часу, але деякі перевірки також слід проводити вручну. Для активного сканування тестер може використовувати OWASP ZAP для перевірки безпеки програми та пошуку активних вразливостей. Burp Suite – чудовий інструмент для ручної перевірки перехоплених повідомлень між браузером та додатком для перегляду та аналізу різних взаємодій. Етап аналізу вразливостей починається з розуміння наслідків раніше виявлених вразливостей, оцінки їх точок входу та класифікації їх відповідно до їх тяжкості. Після аналізу всієї інформації про знайдені вразливості починається експлуатація вразливостей, на якій тестувальник намагається використовувати точки входу, щоб підтвердити або спростувати наявність вразливості на цілі, а також протестувати можливі

точки входу або експлойти, які не були виявлені на етапі сканування, але які додаток не може використовувати. Під час цього процесу тестер може виявити нові вразливості. У цьому випадку їх слід проаналізувати ще раз, створюючи цикл між етапами аналізу вразливості та використання вразливостей, поки більше не будуть виявлені нові вразливості. Припустимо, що більше немає вразливостей для тестування. У цьому випадку тестувальник повинен перейти до етапу аналізу результатів, щоб оцінити результати експлуатації, зробивши висновок про можливий вплив, який експлуатація може мати на систему, пов'язані з цим ризики, способи їх запобігання та загальний результат тестування з точки зору визначених показників якості. Останнім етапом методики є етап звітності, на якому тестувальник повинен заповнити та надати замовнику звіт із детальною інформацією про обсяг, виявлені вразливості, кроки для їх пошуку та використання, можливий вплив та будь-які додаткові технічні деталі.

4.5 Тестовий приклад - DVWA

У цьому підрозділі представлений тестовий приклад застосування розробленої методики з використанням конкретних методів та інструментів для пошуку вразливостей застосунку DVWA. Кожен етап методики пояснений з точки зору процедур.

4.5.1 Планування

Основна мета тестування – зосередити зусилля на аналізі процесу та ручній перевірці програми та її функціональних можливостей, а також на перевірці коду та тестуванні безпеки. Першим кроком є планування тесту. Оскільки про додаток DVWA відомий доволі великий обсяг інформації та на сторінках із завданнями можна отримати доступ до вхідного коду будемо вважати, що з замовником було вирішено використовувати підхід "сірого ящика". Тобто, система, що стоїть за DVWA, була частково пояснена, доступ був наданий до певного порту установки. Далі необхідно підписати угоду про

нерозголошення, щоб запобігти витоку конфіденційної інформації (конкретні каталоги, інформація про деякі порти і локалізація деяких вразливостей), і обрати інструменти для проведених досліджень (Nmap, Metasploit, Burp Suite Community Edition, OWASP ZAP). Знання інфраструктури системи та деяких використовуваних протоколів може стати гарною відправною точкою для складання списку відомих вразливостей її компонентів.

4.5.2 Початковий збір інформації

На цьому етапі передбачається збір інформації щодо програми, щоб дізнатися більше про її структуру та компоненти програми. З огляду на це, другий крок методики починається зі сканування програми та мережі, яка її підтримує.

Використовуючи Burp Suite, можна встановити більшість каталогів, що використовуються DVWA, вручну перевіряючи програму та шукаючи інші порти, пов'язані з веб-додатком. Після складання карти програми настав час скласти карту мережі за допомогою Nmap. Команда Nmap, яка використовувалася для цього:

nmap -sC -sV -oN nmap/initial «адреса веб-застосунку»

Де

-sC = Script Scan

-sV = Service Version Detection

-oN = output scan in normal format

Потім Nmap повідомляє про всі доступні порти, які використовує програма, тип використовуваних портів та їх стани, запущені служби, автентифікацію та тип існуючої автентифікації та сценарії хосту.

Після встановлення програми і мережі необхідно шукати будь-яке підключення до бази даних, але не для того, щоб використовувати його, а для

того, щоб мати базові знання про структуру і всіх належних їй вузлах. Для того щоб знайти всі адреси бази даних, був використаний один з допоміжних модулів Metasploit, яким в даному випадку був MSSQL_Ping. Налаштовуючи облікові дані (ім'я користувача та пароль) та цільову адресу або діапазон хостів, цей модуль запитує певний порт для визначення портів прослуховування TCP будь-якого сервера MSSQL. На цьому збір інформації щодо програми завершено, і більша частина області дослідження була нанесена на карту і готова до сканування та аналізу.

4.5.3 Сканування

Цей етап спрямований на пошук нових вразливостей та нових точок входу за допомогою більш активного сканування, ніж попередній етап. Використовуючи правильні інструменти та правильну конфігурацію, можна знайти вразливості в системі. Для цього в основному використовуються автоматичні сканери, а також іноді ручна перевірка коду по HTTP-запитам.

Спочатку використовувався Burp Suite для аналізу запитів, зроблених між сервером і проксі-сервером установки. Він призначався для більш точного пошуку повідомлень про аутентифікацію, виконань і звернень до бази даних. На цьому етапі використовувались інструменти Burp Suite: проксі, декодер (використовується для декодування хешів), ретранслятор (для маніпулювання запитами, щоб спробувати знайти деякі вразливості у відповідях сервера) і, нарешті, секвенсор (який використовувався для перевірки захисних маркерів CSRF). Цей процес передбачав велику ручну перевірку програми та повідомлень, якими обмінювались обидві системи. Оскільки Burp Suite зберігає всі каталоги програм, є можливість створити список слів, що використовуються в контексті програми, який згодом можна використовувати для брутфорсу або вгадування каталогу. При виконанні ручної перевірки DVWA з'явилось достатньо помилок програми. Наприклад, коли у додатку виникає внутрішня помилка (під час виконання будь-якого сценарію, який містить помилку або обробляє невірні дані), з'являється стек помилок з

інформацією як з інтерфейсу, так і з бекенда. Проаналізувавши стек, вдалося знайти деяку конфіденційну інформацію, яка стосувалася помилки та інформації про внутрішні скрипти, каталоги і файли.

Потім OWASP ZAP був використаний для сканування програми за допомогою аутентифікації HTTP із заданими обліковими даними, також було проведено ручне сканування та відзначено наступні вразливості. На рисунку 4.1 відображені виявлені вразливості.

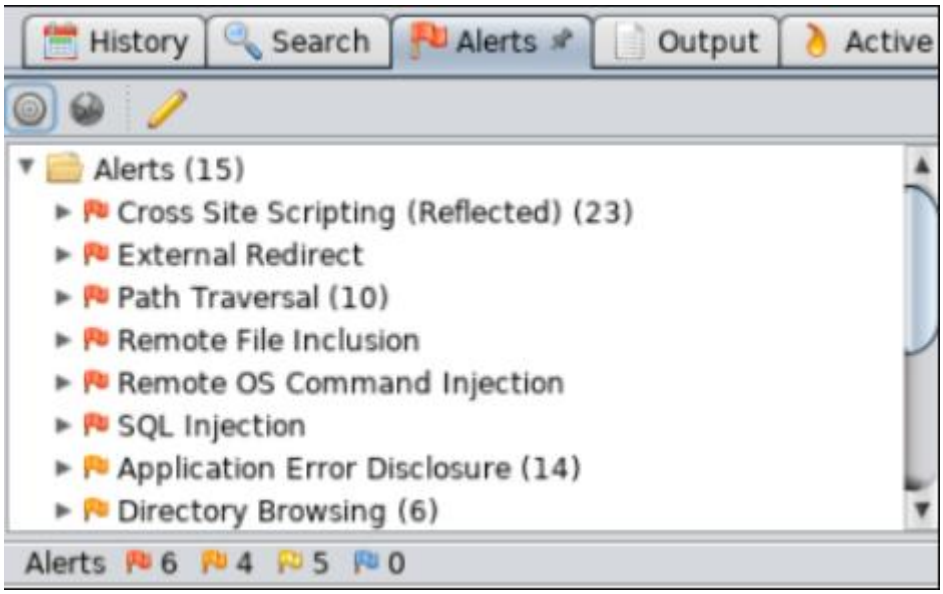


Рисунок 4.1 - Виявлені вразливості сканером OWASP ZAP

За допомогою OWASP ZAP Spider також можна було зіставити всі раніше виявлені каталоги, сценарії та файли, але також перевірялися форми введення та корисні навантаження, невідомі до цього моменту. Враховуючи результати, помітно, що було виявлено набагато більше вразливостей і, в даному випадку, з більш високим ступенем серйозності.

4.5.4 Аналіз вразливостей

Наступним кроком в розробленій методиці є аналіз вразливостей, виявлених до цього моменту. Для того щоб зробити це, в обов'язковому порядку необхідно перевірити виявлені вразливості, зрозуміти, як вони працюють і їх вплив на додаток в разі використання.

На етапі сканування ми отримали п’ятнадцять різних вразливостей, з яких шість мали високу ступінь серйозності. Тяжкість вразливості визначається загальною системою оцінки вразливості (CVSS); вона оцінює серйозність, а не ризик, безпосередньо пов'язаний з вразливістю, представляючи лише внутрішні характеристики вразливості, які є постійними з часом та в різних середовищах користувачів. Серйозність – це показник, який використовується для оцінки та інформування про характеристики та наслідки вразливостей безпеки.

Найбільш серйозні уразливості, виявлені в ході тестування, наступні:

- Cross Site Scripting – це тип ін’єкцій, під час яких шкідливі сценарії впроваджуються на безпечні та надійні веб-сайти. XSS-атаки відбуваються, коли зловмисник використовує веб-програму для надсилання шкідливого коду, як правило, у формі сценарію сторони браузера, іншому кінцевому користувачеві. Недоліки, які дозволяють цим атакам бути успішними, досить широко поширені та виникають у будь-якому місці, де веб-додаток використовує вхідні дані від користувача в межах вихідних даних, які він генерує, без їх перевірки чи кодування [33].
- External Redirect. Переспрямування URL-адрес представляють загальні функції, які використовуються веб-сайтами для пересилання вхідного запиту до альтернативного ресурсу. Це може бути зроблено з різних причин і часто робиться для того, щоб дозволити переміщувати ресурси всередині структури каталогів і уникнути порушення функціональності для користувачів, які запитують ресурс у його попередньому розташуванні [34].
- Path Traversal. Атака спрямована на доступ до файлів і каталогів, які зберігаються поза кореневою текою веб-сайту. Маніпулюючи змінними, які посилаються на файли з послідовністю «точка-крапка-слеш (../)» та її варіаціями або використовуючи абсолютні шляхи до файлів, можна

отримати доступ до довільних файлів і каталогів, що зберігаються у файловій системі, включаючи вихідний код програми або конфігурацію і критичні системні файли [35].

- Remote File Inclusion — це атака, спрямована на вразливі місця у веб-додатках, які динамічно посилаються на зовнішні сценарії. Метою зловмисника є використання функції посилань у програмі для завантаження зловмисного програмного забезпечення (наприклад, бекдор-оболонок) із віддаленої URL-адреси, розташованої в іншому домені [36].
- Remote OS Command Injection. Це вразливість веб-безпеки, яка дозволяє зловмиснику виконувати довільні команди операційної системи на сервері, на якому запущено програму, і зазвичай повністю скомпрометувати програму та всі її дані. Дуже часто зловмисник може використовувати вразливість ін’єкції команд ОС, щоб скомпрометувати інші частини інфраструктури хостингу, використовуючи довірчі відносини, щоб скерувати атаку на інші системи в організації [37].
- SQL Injection. Це вразливість веб-безпеки, яка дозволяє зловмиснику втручатися в запити, які програма робить до своєї бази даних. Зазвичай це дозволяє зловмиснику переглядати дані, які вони зазвичай не можуть отримати. Це може включати дані, що належать іншим користувачам, або будь-які інші дані, до яких має доступ сама програма. У багатьох випадках зловмисник може змінити або видалити ці дані, викликаючи постійні зміни у вмісті або поведінці програми [38].

Не всі з виявлених сканером вразливості мають прямий вплив на додаток, хоча деякі з них можуть призвести до втрати конфіденційних даних.

4.5.5 Використання вразливостей

На цьому етапі основна мета полягає в тому, щоб підтвердити, чи є виявлені вразливості реальними вразливостями або просто хибно-позитивними спрацьовуваннями, виявленими на попередніх етапах; також

передбачається вивчити нові точки входу або можливі експлойти, вразливості які не були відзначені, але все ще можуть бути присутніми.

Всі перераховані вище вразливості заявлені сканером як позитивні (підтверджені), але не експлуатувалися під час тестування, оскільки вони існували через конфігурації, зроблених в мережі і самим додатком навмисно. Однак були протестовані й інші вразливості, щоб забезпечити максимальну ефективність тестування. Параметри конфігурації для кожної вразливості перевірені, щоб підтвердити отримані результати. Протестовані такі вразливості, як сліпий міжсайтовий сценарій (BSQLi), заміна HTTP-запитів та атаки грубої сили. Після всіх тестів було виявлено що додаток має вразливість BSQLi. При порівнянні сканерів OWASP ZAP продемонстрував, що він майже не визначає цей тип вразливості [28], тому було вирішено перевірити наявність цієї вразливості вручну. Сліпа ін’єкція SQL – це тип атаки ін’єкції SQL, яка задає базі даних істинні чи хибні запитання та визначає відповідь на основі відповіді програми [39]. Ця атака часто використовується, коли веб-програма налаштована на показ загальних повідомлень про помилки, але не пом’якшила код, уразливий до впровадження SQL. Коли зловмисник використовує SQL-ін’єкцію, іноді веб-додаток відображає повідомлення про помилки з бази даних, які скаржаться на те, що синтаксис SQL-запиту є неправильним. Сліпа SQL-ін’єкція майже ідентична звичайній SQL-ін’єкції, єдина відмінність полягає в способі отримання даних із бази даних. Коли база даних не виводить дані на веб-сторінку, зловмисник змушений викрасти дані, задавши базі даних низку правдивих чи хибних запитань. Це робить використання вразливості SQL Injection більш складним, але не неможливим [39]. Тому ця вразливість вважається теж дуже серйозною та має бути виявлена.

Оскільки знайдені сканером вразливості є підтвердженими та була виявлена серйозна вразливість при перевірці застосунку DVWA вручну, можна стверджувати, що знайдені вразливості можуть бути використані в додатку та порушувати його безпеку.

4.5.6 Аналіз результатів

Враховуючи результати, і кількість виявлених та підтверджених вразливостей, використовуючи лише безкоштовні сканери, у застосунку DVWA, можна стверджувати, що цей додаток є абсолютно небезпечним.

4.5.7 Звітність

Повідомлення технічних деталей тесту і всіх аспектів, запланованих на першому етапі методики, здійснюється на етапі звітності. На цьому етапі тестувальник повинен розробити документ, що містить опис області дослідження, інформацію, розкриту в процесі, кроки, вжиті тестувальником для використання будь-якої вразливості, вплив виконаної експлуатації.

У цьому випадку звіт про весь процес тестування складається з цього розділу роботи. Приклад оформленого звіту розміщено у Додатку А.

Оскільки досліджуваний додаток є загальнодоступним та розроблений саме для таких перевірок, немає необхідності скривати результати тестування спираючись на угоду про нерозголошення, яка підписується з замовником на етапі планування.

4.6 Підсумок тестового прикладу

Результати тестування на проникнення показують, що додаток має велику кількість вразливостей, головним чином з точки зору конфігурації та розкриття конфіденційних даних (каталогів, сценаріїв та додаткової системної інформації) та доступних для перегляду каталогів. Більшість цих вразливостей можна використовувати, порушуючи безпеку веб-застосунку, тому цей додаток є дуже небезпечним. Однак, оскільки DVWA є навмисно вразливим та використовується саме для практики з пошуку вразливостей, ці вразливості не становлять критичної загрози для користувача.

ВИСНОВКИ

У роботі було розроблено нову методику для пошуку вразливостей у веб-застосунках, яка включає сім етапів, подібних до стандарту PTES, із застосуванням деяких стандартів у межах основних розділів, проведенням оглядової наради, використанням подібного підходу до процесу збору інформації та дотримуючись тих самих правил взаємодії, водночас активніше підходячи до етапу використання вразливостей, дотримуючись тих самих основ кожного етапу методики OWASP.

Основною відмінністю та цінністю запропонованої методики є її загальність. PTES представляє стандарт, якого повинні дотримуватися як клієнти, так і тестувальники, дозволяючи компаніям отримати більше інформації про конкретні етапи тестування, а тестувальникам отримати більше інформації про те, що слід взяти до уваги, і дії, необхідні для виконання тестування. OWASP представляє детальну методику тестування програм протягом усього життєвого циклу розробки програмного забезпечення та безпосередньо пов’язана не лише з програмою, але й із усією системою та її мережею. У розробленій методиці будь-який тестувальник може знайти загальні методи та завдання для виконання повного тесту на проникнення, зосередившись на самому додатку, а не на системі в цілому.

Для підтвердження концепції програмне забезпечення DVWA було використано як тестовий приклад для пошуку вразливостей за визначеною методикою. Застосовуючи цю методику до реального прикладного сценарію, який є загальнодоступним та існує саме для практики в області кібербезпеки, і отримавши результати, які відповідають стандартам двох найбільш застосовуваних методик тестування на проникнення, можна припустити, що вона може застосовуватись для пошуку вразливостей веб-застосунків. Було описано кожен крок, зроблений під час процесу використання методики,

перераховано і описано використовувані інструменти та наведено інформацію щодо знайдених вразливостей.

Ця робота також має прикладне значення, оскільки вона містить не лише порівняння між різними типами інструментів і аналіз існуючих методик та стандартів, але й запропоновує методику для тестування веб-додатків із використанням конкретних стандартів тестування та інструментів. Також зрозуміло, що для розробленого програмного забезпечення слід проводити постійний аналіз безпеки, оскільки ризики зростають із впровадженням технологій. Навіть якщо компанії вживають заходів для запобігання будь-яким недолікам безпеки, необхідно пам’ятати про наявність людської помилки.

ПЕРЕЛІК ПОСИЛАНЬ

1. Morgan S. Cybercrime To Cost The World $10.5 Trillion Annually By 2025. *Cybercrime Magazine*. 2020. URL: https://cybersecurityventures.com/cybercrime-damage-costs-10-trillion-by-2025/ (дата звернення: 20.09.2022).
2. Kaner C. Testing computer software. 2-ге вид. New York : Van Nostrand Reinhold, 1993. 480 с.
3. A Methodological Approach for Assessing Amplified Reflection Distributed Denial of Service on the Internet of Things / J. Costa Gondim та ін. *Sensors*. 2016. Т. 16, № 11. С. 1855. URL: https://doi.org/10.3390/s16111855 (дата звернення: 06.09.2022).
4. Bechtsoudis A., Sklavos N. Aiming at Higher Network Security through Extensive Penetration Tests. *IEEE Latin America Transactions*. 2012. Т. 10, № 3. С. 1752–1756. URL: https://doi.org/10.1109/tla.2012.6222581 (дата звернення: 06.09.2022).
5. João H. P. Web application penetration test: Proposal for a generic web application testing methodology : master Thesis. 2021. URL: http://hdl.handle.net/10071/24246 (дата звернення: 06.09.2022).
6. Cardwell K. Building Virtual Pentesting Labs for Advanced Penetration Testing - Second Edition. Packt Publishing - ebooks Account, 2016. 524 с.
7. Энциклопедический словарь Брокгауз и Ефрон. Биографии : в 12 т. Москва : Науч. изд-во "Большая Российская Энциклопедия", 1992. Т. 2 : Бейер - Вакер. 831 с.
8. OSSTMM - The Open Source Security Testing Methodology Manual. *ISECOM* : веб-сайт. URL: https://www.isecom.org/OSSTMM.3 (дата звернення: 11.09.2022).

9. Information Systems Security Assessment Framework (ISSAF) draft 0.2. *OISSG* : веб-сайт. URL: https://untrustednetwork.net/files/issaf0.2.1.pdf. (дата звернення: 11.09.2022).
10. WSTG - Latest | OWASP Foundation. *OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation* : веб-сайт. URL: https://owasp.org/www-project-web-security-testing-guide/latest/3-The_OWASP_Testing_Framework/1-Penetration_Testing_Methodologies (дата звернення: 11.09.2022).
11. PTES. Penetration Testing Execution Standard. Чинний від 2014-08-16. Вид. офіц. URL: http://www.pentest-standard.org/index.php/Main_Page. (дата звернення: 11.09.2022).
12. NIST SP 800-115. Technical Guide to Information Security Testing and Assessment. Чинний від 2008-09-16. Вид. офіц. Gaithersburg. 80 с.
13. BSI Study a penetration testing Model. *BSI - Bundesamt für Sicherheit in der Informationstechnik* : веб-сайт. URL: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/Penetration/penetration_pdf.pdf?__blob=publicationFile&v=1 (дата звернення: 11.09.2022).
14. Klíma T. PETA: Methodology of Information Systems Security Penetration Testing. *Acta Informatica Pragensia*. 2016. Т. 5, № 2. С. 98–117. URL: https://doi.org/10.18267/j.aip.88 (дата звернення: 28.11.2022).
15. Orrey K. Penetration Testing Framework 0.59. *VulnerabilityAssessment.co.uk*. : веб-сайт. URL: http://www.vulnerabilityassessment.co.uk/Penetration%20Test.html (дата звернення: 28.11.2022).
16. Макаренко С. И. Аудит безопасности критической инфраструктуры специальными информационными воздействиями [Security audit of critical infrastructure with special information impacts] : монографія. Санкт-Петербург : Наукоемкие технологии, 2018. 116 с.

17. Burp Suite - Application Security Testing Software. *Web Application Security, Testing, & Scanning - PortSwigger*. : веб-сайт. URL: https://portswigger.net/burp (дата звернення: 28.10.2022).
18. The ZAP Homepage. *OWASP ZAP*. : веб-сайт. URL: https://www.zaproxy.org/ (дата звернення: 28.10.2022).
19. Foca Free - Мощный инструмент для извлечения и анализа метаданных. *SPY-SOFT.NET* : веб-сайт. URL: https://spy-soft.net/foca-free/ (дата звернення: 28.10.2022).
20. Metacrawler. *Metacrawler 2.2.8.1247 – Download* : веб-сайт. URL: https://metacrawler.updatestar.com/ru (дата звернення: 28.10.2022).
21. SonarQube. *Code Quality and Code Security | SonarQube* : веб-сайт. URL: https://www.sonarqube.org/ (дата звернення: 28.10.2022).
22. HCL AppScan. *HCLSoftware* : веб-сайт. URL: https://www.hcltechsw.com/ru/products/appscan (дата звернення: 28.10.2022).
23. National Vulnerability Database. *NVD NIST* : веб-сайт. URL: https://nvd.nist.gov/ (дата звернення: 28.10.2022).
24. OSVDB. *Open Source Vulnerability Database* : веб-сайт. URL: http://osvdb.org/ (дата звернення: 28.10.2022).
25. Metasploit | Penetration Testing Software, Pen Testing Security | Metasploit. *Metasploit* : веб-сайт. URL: https://www.metasploit.com/ (дата звернення: 28.10.2022).
26. Offensive Security's Exploit Database Archive. *Exploit Database - Exploits for Penetration Testers, Researchers, and Ethical Hackers* : веб-сайт. URL: https://www.exploit-db.com/ (дата звернення: 28.10.2022).
27. Ncat, Netcat, nc - Инструменты Kali Linux. *Инструменты Kali Linux - Список инструментов для тестирования на проникновение и их описание* : веб-сайт. URL: https://kali.tools/?p=4578 (дата звернення: 28.10.2022).

28. Лахтін І. А. Порівняння комерційних сканерів вразливостей веб-додатків та сканерів з відкритим кодом. *CS&CS Journal*. 2022. № 2 (22).
29. DVWA – Damn Vulnerable Web Application. *Национальная библиотека им. Н. Э. Баумана* : веб-сайт. URL: https://ru.bmstu.wiki/DVWA (дата звернення: 07.11.2022).
30. Nmap: the Network Mapper - Free Security Scanner. *Nmap* : веб-сайт. URL: https://nmap.org/ (дата звернення: 28.10.2022).
31. Wireshark · Frequently Asked Questions. *Wireshark · Go Deep* : веб-сайт. URL: https://www.wireshark.org/faq.html#_what_is_wireshark (дата звернення: 07.11.2022).
32. HP WebInspect. *HP* : веб-сайт. URL: http://www.hp.com/hpinfo/newsroom/press_kits/2011/risk2011/HP_WebInspect_data_sheet.pdf (дата звернення: 27.10.2022).
33. Cross Site Scripting (XSS) | OWASP Foundation. *OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation* : веб-сайт. URL: https://owasp.org/www-community/attacks/xss/ (дата звернення: 10.11.2022).
34. OWASP ZAP – External Redirect. *OWASP ZAP* : веб-сайт. URL: https://www.zaproxy.org/docs/alerts/20019-1/ (дата звернення: 10.11.2022).
35. Path Traversal | OWASP Foundation. *OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation* : веб-сайт. URL: https://owasp.org/www-community/attacks/Path_Traversal (дата звернення: 10.11.2022).
36. What is RFI | Remote File Inclusion Example & Mitigation Methods | Imperva. *Learning Center* : веб-сайт. URL: https://www.imperva.com/learn/application-security/rfi-remote-file-inclusion/ (дата звернення: 10.11.2022).

37. What is OS command injection. *Web Application Security, Testing, & Scanning – PortSwigger* : веб-сайт. URL: https://portswigger.net/web-security/os-command-injection (дата звернення: 10.11.2022).
38. What is SQL Injection. *Web Application Security, Testing, & Scanning – PortSwigger* : веб-сайт. URL: https://portswigger.net/web-security/sql-injection (дата звернення: 10.11.2022).
39. Blind SQL Injection | OWASP Foundation. *OWASP Foundation, the Open Source Foundation for Application Security | OWASP Foundation* : веб-сайт. URL: https://owasp.org/www-community/attacks/Blind_SQL_Injection (дата звернення: 10.11.2022).

ДОДАТОК А

Приклад звіту розробленої методики

ЗВІТ З ПРОВЕДЕНОГО ПОШУКУ ВРАЗЛИВОСТЕЙ

DVWA

Аудитор: Лахтін І.А.

20.10.2022

ЗМІСТ

Цілі проекту 3

Обсяг проекту 3

Терміни 3

Список цілей 4

Обмеження 4

Висновки 5

Технічний звіт 6

Цілі проету

Це зразок звіту. Метою цього зразка звіту є лише показати уявлення про те, як може виглядати звіт про тест на проникнення. У цьому підрозділі звіту тестувальник повинен окреслити всі цілі тесту на проникнення, а також використовувані показники та їх значення відповідно до результатів та очікуваного результату тестів

Обсяг проекту

Завдання полягає в проведенні пошуку вразливостей в одному веб-застосунку DVWA. Інші системи в мережі не повинні тестуватися.

Всі вимоги щодо об’єму під час тестування були витримані.

Терміни

Тестування проводилось протягом періоду з 19.10.2022 року по 20.10.2022 (повний день). Доступ до мережі через VPN було надано на час проведення тесту на проникнення. Тестеру проникнення було присвоєно IP-адресу 172.17.0.1.

Усі знайдені вразливості зазвичай надаються з датою та міткою часу, коли вони були знайдені/використані.

Список цілей

- Ціль має IP-адресу 172.0.0.2.

Обмеження

Тест на проникнення класифікується як обережний тест сірого ящика в обмеженому діапазоні. Ми не вживаємо заходів, щоб бути непомітними під час тесту. Тест виконується з мережі компанії.

Інформаційна цінність тесту завжди обмежена.

Загалом безпеку слід розглядати як безперервний процес. Тому рекомендується регулярні тести безпеки, які можуть значно підвищити безпеку технічних систем, організаційної та кадрової інфраструктури.

Висновки

Було проведено пошук вразливостей у застосунку DVWA, за результатами якого, виявлено наступна кількість вразливостей:

Вразливість	Рівень ризику	Час виявлення
SQL Injection	Високий	13:30 19.10.2022
Blind SQL Injection	Високий	16:00 19.10.2022
Command Execution	Високий	16:15 19.10.2022
Insufcient Password Policy	Середній	11:10 20.10.2022
Deprecated Hash Function	Низький	10:30 19.10.2022

Технічний звіт

Було проведено пошук вразливостей у застосунку DVWA, за результатами якого, виявлено наступні вразливості:

Command Execution

Час виявлення	16:15 19.10.2022
Ризик для додатку	Високий
Інструмент виявлення	OWASP ZAP

Опис вразливості

Поле введення вказує на ціль команди ping, наприклад, IP-адресу. Однак введені користувачем дані можна об'єднати з іншою системною командою, яка виконуватиметься як дані користувача www. Згодом ця вразливість може призвести до виконання коду.

Скріншот підтверджуючий наявності вразливості

Тестер виконав дві некритичні команди id (перелік усіх користувачів) та ip a (перелік мережевих конфігурацій) через уразливість виконання коду. Результат команди не відображається у веб-застосунку

SQL Injection

Час виявлення	13:30 19.10.2022
Ризик для додатку	Високий
Інструмент виявлення	OWASP ZAP

Опис вразливості

Веб-програма схильна до SQL-ін'єкцій. Як наслідок, зловмисник може потенційно отримати доступ до всіх даних бази даних або маніпулювати ними

Скріншот підтверджуючий наявності вразливості

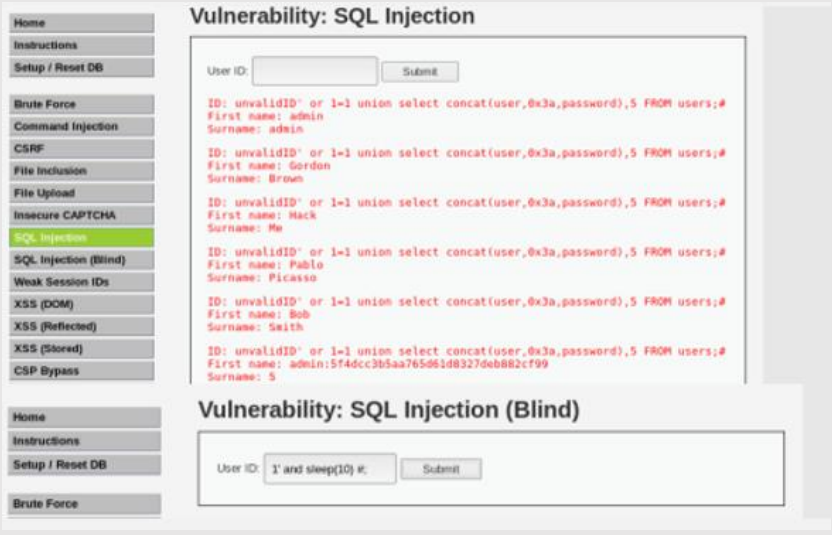
Введені дані та їх значення наведено нижче. Знімок екрана ілюструє доступ до бази даних, де друкуються всі хешовані паролі MD5, а також виконання команди SQL sleep().

```
Number of columns can
be          enumerated:
unvalidID'   or   1=1
order by 3 ;#
Note: order by 3 or higher provokes an SQL error: Unknown column '3' in 'order clause'

Table names can be enumerated:
unvalidID' or 1=1 union all select table_name,5 FROM information_schema.tables;#

Column names of specific a table can be enumerated:
unvalidID' or 1=1 union all select column_name,5 FROM information_schema.columns where table_name='users';#

A table can be enumerated (table users in this case):
unvalidID' or 1=1 union select concat(user,0x3a,password),5 FROM users;#
```



Blind SQL Injection

Час виявлення	16:00 19.10.2022
Ризик для додатку	Високий
Інструмент виявлення	Під час використання вразливостей вручну

Опис вразливості

Це тип атаки ін’єкції SQL, яка задає базі даних істинні чи хибні запитання та визначає відповідь на основі відповіді програми

Скріншот підтверджуючий наявності вразливості

Insufcient Password Policy

Час виявлення	11:10 20.10.2022
Ризик для додатку	Середній
Інструмент виявлення	Ручна перевірка

Опис вразливості

Користувач програми може встановити слабкий пароль. Ці паролі схильні до атак грубої сили. Конфіденційність не може бути забезпечена.

Скріншот підтверджуючий наявності вразливості

Deprecated Hash Function

Час виявлення	10:30 19.10.2022
Ризик для додатку	Низький
Інструмент виявлення	OWASP ZAP

Опис вразливості

Паролі користувачів у базі даних MySQL зберігаються як хеші MD5. Відомо, що MD5 застарів і його більше не слід використовувати. Якщо зловмисники зможуть отримати доступ до бази даних, хеш-значення можна легко перетворити на паролі користувачів у відкритому тексті.

Скріншот підтверджуючий наявності вразливості

ДОДАТОК Б

Копії публікації

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені В.Н.КАРАЗІНА
ХАРЬКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ имени В.Н. КАРАЗИНА
V.N. KARAZIN KHARKIV NATIONAL UNIVERSITY

КОМП'ЮТЕРНІ НАУКИ ТА КІБЕРБЕЗПЕКА
КОМПЬЮТЕРНЫЕ НАУКИ И КИБЕРБЕЗОПАСНОСТЬ
COMPUTER SCIENCE AND CYBERSECURITY
(CS&CS)

Issue 2(22) 2022

Заснований 2015 року

Міжнародний електронний науково-теоретичний журнал
Международный электронный научно-теоретический журнал
International electronic scientific journal

ISSN 2519-2310 CS&CS, Issue 2(22) 2022

УДК 681.04

ПОРІВНЯННЯ КОМЕРЦІЙНИХ СКАНЕРІВ ВРАЗЛИВОСТЕЙ ВЕБ-ДОДАТКІВ ТА СКАНЕРІВ З ВІДКРИТИМ КОДОМ

Лахтін Іван

Харківський національний університет імені В.Н. Каразіна, майдан Свободи 4, Харків, 61022, Україна
lakhtin.ivan@gmail.com

Рецензент: Роман Олійников, д.т.н., проф., Харківський національний університет імені В.Н. Каразіна, майдан Свободи, 4, Харків, 61022, Україна
roliynykov@gmail.com

Надійшла: Листопад 2022.

***Анотація**: У роботі порівнюється вісім сканерів вразливостей на основі двох, навмисно вразливих додатків. Порівняння виконується за допомогою п'яти критеріїв: точність, відкликання, розрахунок індекса Юдена, веб-бенчмарк від WASSEC та OWASP. У якості додатків, що тестувалися обрані: OWASP WebGoat та Damn Vulnerable Web Application (DVWA). Серед* досліджуваних сканерів є три комерційних сканера: Acunetix, HP WebInspect, AppScan, та п'ять сканерів з відкритим кодом такі, як: Arachni, IronWASP, Skipfish, OWASP ZAP, Vega. *За результатами тестування зроблено висновок, що комерційні сканери є більш ефективними по ряду показників (в т.ч. переліку загроз).*

***Ключові слова:** веб-додаток; вразливість, атака, сканер, безпека, тестування.*

1 Вступ

Економічна важливість веб-додатків у багатьох сферах, включаючи банківську діяльність, транспорт, промисловість, бізнес та освіту, збільшила потребу в механізмах контролю та підвищення їх якості. Широке використання веб-додатків в усіх галузях сучасної економіки, нажаль, призвело до настільки ж різкого збільшення числа атак. Ці атаки, як правило, спрямовані на «слабкі місця» програмного коду додатків, їх недоліки та помилки, що обумовлює наявність передумов виникнення вразливостей в механізмах забезпечення конфіденційності, цілісності та доступності програмних рішень, що пропонуються [1]. За оцінками спеціалістів з питань інформаційної безпеки (ІБ), щодня атакується більше мільйона веб-сайтів, причому 75% з цих веб-сайтів містять не усунені вразливості [2]. Коли атаки досягають своєї мети, вони можуть призвести до компрометації критичних даних та/або програмно-апаратних ресурсів жертви, та мати інші серйозні наслідки, які можуть поширювати далеко за межи корпоративної інфраструктури компанії (наприклад, створення бот-мережі).

Розробники програмного забезпечення (ПЗ) використовують сканери вразливостей для автоматизації процесу перевірки стану ІБ «своїх» веб-додатків та/або проведення масштабних тестувань поточного стану безпеки багатьох інших відомих веб-додатків [3]. Широке поширення сканерів вразливостей і існуючи відмінності в їх функціональності, щодо виявлення вразливостей, підвищують інтерес відносно оцінки (тестування) ефективності їх роботи. Основними цілями подібних тестувань, є оцінка та порівняння продуктивності комерційних сканерів та рішень з відкритим кодом, що дозволяє дослідити реальні показники їх можливостей, стосовно виявлення відомих загроз ІБ. У цій роботі представлені результати порівняльної оцінки деяких функцій безпеки та продуктивності для восьми існуючих інструментів виявлення вразливостей.

На даний час існує достатньо проектів, які спрямовані на вивчення можливостей сканерів вразливостей веб-додатків, це можуть бути інтернет-блоги в сфері IT, або офіційні сайти якогось з інструментів безпеки. Але в цих випадках порівняння сканерів зводиться до переліку переваг і їх недоліків, короткого опису сканерів та порівняння ціни.

УДК 004.492.2

АНАЛІЗ ВРАЗЛИВОСТЕЙ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА ОСНОВІ WEBGOAT

Лахтін І. А., магістрант гр. КБ-51
Науковий керівник: професор Олійников Р.В.
Харківський національний університет ім. В.Н. Каразіна

Тема роботи є актуальною, тому що веб-ресурси набувають все більшого значення у житті людини. Однак, не дивлячись на таку популярність, не так багато систем було присвячено аналізу вразливостей веб-додатків, особливо методам виявлення вразливостей програмного забезпечення та їх можливого використання.

Метою цієї роботи є дослідження вразливостей, що найчастіше зустрічається у веб-додатках, за допомогою системи WebGoat, яка створена навмисно небезпечною, щоб користувач навчився виявляти ці вразливості. Для досягнення поставленої мети необхідно ознайомитись з системою WebGoat та виявити чи достатньо матеріалів надається для вивчення обраних вразливостей.

На даний момент існує декілька проєктів, що допомагають навчитись виявляти вразливості у веб-додатках. Серед умовно безкоштовних можна виділити такі, як: Buggy web application (bWAPP), Damn Vulnerable Web Application (DVWA), Mutillidae, Metasploitable 2. Серед оплачуваних аналогів – Virtual Hacking Labs та PentesterLab Pro.

У ході аналізу були виявлені основні вразливості веб-додатків. В системі WebGoat були розглянуті найбільш популярні та небезпечні вразливості, за метрикою OWASP Top-10, такі, як: XSS, CSRF та SQL Injection. Також були вивчені загрози для інформації, до яких можуть привести ці вразливості.

WebGoat може використовуватись фахівцями з інформаційної безпеки, як система для отримання практичних знань в області безпеки програмного забезпечення. Найбільш небезпечними вразливостями на сьогоднішній день є XSS, CSRF та SQL Injection. Ці вразливості мають бути виявлені та виправлені в першу чергу.