Міністерство освіти і науки України
Державний вищий навчальний заклад
«Національний гірничий університет»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА
дипломної роботи

магістра
(назва освітньо-кваліфікаційного рівня)

галузь знань *12 Інформаційні технології*
(шифр і назва галузі знань)

напрям підготовки (спеціальність) *125 Кібербезпека*
(код і назва напряму підготовки)

спеціалізація (освітня програма) *Кібербезпека*
(код і назва спеціальності)

освітній рівень *магістр*
(назва освітнього рівня)

кваліфікація *професіонал із організації інформаційної безпеки*
(код і назва кваліфікації)

на тему: ***Методики тестування на проникнення веб - додатків***

Виконавець: студент ***6*** **курсу, групи** ***125м – 16 – 1***

(підпис) ***Кот Леонід Леонідович*** (прізвище ім'я по-батькові)

Керівники	Прізвище, ініціали	Оцінка	Підпис
роботи	к.ф-м.н., доц. Гусєв О.Ю.		
розділів:			
спеціальний	ст. викл. Кручінін О.В.		
економічний	к.е.н., доц. Волотковська Ю.О.		
Рецензент			
Нормоконтроль	к.ф-м.н.,доц. Гусєв О.Ю.		

Дніпро
2018

Міністерство освіти і науки України
Державний вищий навчальний заклад
«Національний гірничий університет»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
_____ Корнієнко В.І.

«_____»_____20___року

ЗАВДАННЯ
на виконання кваліфікаційної роботи магістра

спеціальності *125 Кібербезпека*
(код і назва спеціальності)

студенту **125м – 16 - 1** (група) **Кот Леоніду Леонідовичу** (прізвище ім'я по-батькові)

Тема дипломної роботи ***Методики тестування на проникнення веб - додатків***

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора Державного ВНЗ «НГУ» від _____№_____

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень ***Процес тестування на проникнення веб - додатків***

Предмет досліджень ***Методики тестування на проникнення***

Мета НДР ***Підвищення ефективності тестування на проникнення веб - додатків***

Вихідні дані для проведення роботи ***Матеріали науково – дослідної та преддипломної практик***

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Наукова новизна ***Адаптація методів тестування на проникнення веб - додатків***

Практична цінність ***Зниження тривалості та обгрунтування вибору засобів проведення тестування на проникнення***

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Відповідність методичним рекомендаціям до підготовки та захисту дипломної роботи

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	Строки виконання робіт (початок-кінець)
Аналіз вразливостей та методик тестування на проникнення веб – додатків	01.11.17 – 30.11.17
Розробка методики тестування на проникнення	01.12.17 – 11.12.17
Обгрунтування вибору засобів проведення тестування на проникнення	12.12.17 – 24.12.17
Розрахунок економічної ефективності методики	25.12.17 – 31.12.17
Оформлення результатів роботи	01.01.18 – 07.01.18

6 РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект ***Зниження вартості тестування на проникнення веб - додатка***

Соціальний ефект ***Підвищення рівня кібербезпеки веб - додатків***

7 ДОДАТКОВІ ВИМОГИ

Відповідність вимогам українського законодавства, відомчих нормативно – правових актів та міжнародних стандартів

Завдання видав _____ Гусєв О.Ю.
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ Кот Л.Л.
(підпис) (прізвище, ініціали)

Дата видачі завдання: 30.10.17

Термін подання дипломної роботи до ЕК _____

РЕФЕРАТ

Пояснювальна записка: с., рис., табл., додатків, джерел.

Об’єкт дослідження: процес тестування на проникнення веб – додатків.

Мета роботи: підвищення ефективності тестування на проникнення веб – додатків.

Методи дослідження: системний підхід, методи порівняння, індексний метод.

У спеціальнй частині розроблено методику тестування на проникнення, яка враховує міжнародні досягнення у тестуванні веб – додатків та містить перелік можливих засобів тестування.

У роботі проведено аналіз вразливостей та методик тестування на проникнення, структури системи реагування на компютерні надзвичайні події, баз даних та засобів пошуку вразливостей.

В економічному розділі наведено обгрунтування запропонованої методики.

Практичне значення роботи полягає у зниженні тривалості та обгрунтуванні вибору засобів тестування на проникнення. Результати здійснених у дипломній роботі досліджень можуть бути використані при тестуванні на проникнення веб – додатків та проведенні лабораторних робіт спеціалізованих курсів спеціальності «Кібербезпека».

Наукова новизна дослідження полягає у адаптації методів тестування на проникнення веб – додатків.

Напрямки подальших досліджень полягають у перевірці ефективності розробленої методики на практиці.

Ключові слова: ТЕСТУВАННЯ НА ПРОНИКНЕННЯ, ВЕБ – СЕРВЕР, ВРАЗЛИВІСТЬ, МЕТОДИКИ, КІБЕРБЕЗПЕКА, ЗАГРОЗА, ВЕБ – ДОДАТОК, АТАКА, ЗЛОВМИСНИК.

РЕФЕРАТ

Пояснительная записка с., рис., табл., приложений, источников.

Объект исследования: процесс тестирования на проникновение веб – приложений.

Цель работы: повышение эффективности тестирования на проникновение веб – приложений.

Методы исследования: системный подход, методы сравнения, индексный метод.

В специальной части разработана методика тестирования на проникновение, которая учитывает международные достижения в тестировании веб – приложений и содержит перечень возможных средств тестирования.

В работе проведен анализ уязвимостей и методик тестирования на проникновение, структуры системы реагирования на компьютерные чрезвычайные события, баз данных и средств поиска уязвимостей.

В экономическом разделе приведено обоснование предложенной методики.

Практическое значение работы заключается в снижении продолжительности и обосновании выбора средств тестирования на проникновение. Результаты проведенных в дипломной работе исследований могут быть использованы при тестировании на проникновение веб – приложений и проведении лабораторных работ специализированных курсов специальности «Кибербезопасность».

Научная новизна исследования заключается в адаптации методов тестирования на проникновение веб – приложений.

Направления дальнейших исследований заключаются в проверке эффективности разработанной методики на практике.

Ключевые слова: ТЕСТИРОВАНИЕ НА ПРОНИКНОВЕНИЕ, ВЕБ - СЕРВЕР, УЯЗВИМОСТЬ, МЕТОДИКИ, КИБЕРБЕЗОПАСНОСТЬ, УГРОЗА, ВЕБ - ПРИЛОЖЕНИЕ, АТАКА, ЗЛОУМЫШЛЕННИК.

ABSTRACT

Explanatory note p., pic., tables., applications, sources.

Object of research: the process of testing for the penetration of web applications.

Objective: increasing the effectiveness of testing for the penetration of web applications.

Research methods: system approach, comparison methods, index method.

In a special part, a technique for penetration testing has been developed that takes into account international achievements in testing web applications and contains a list of possible testing tools.

The work analyzes the vulnerabilities and methods of penetration testing, the structure of the system for responding to computer emergency events, databases and means of vulnerability search.

In the economic section, the proposed method is justified.

The practical importance of the work is to reduce the duration and justification of the choice of penetration testing tools. The results of the research conducted in the work can be used to penetration test of web applications and conduct laboratory work of specialized courses of the specialty Cybersecurity.

The scientific novelty of the study is to adapt the testing methods to the penetration of web applications.

The directions of further research are to test the effectiveness of the developed methodology in practice.

Key words: PENETRATION TEST, WEB SERVER, VULNERABILITY, METHODS, CYBERSECURITY, THREAT, WEB APPLICATION, ATTACK, WRECKER.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

CERT (Computer Emergency Response Team) – команда реагування на комп’ютерні надзвичайні події

CERT – UA (Computer Emergency Response Team – Ukraine) – команда реагування на комп’ютерні надзвичайні події в Україні

CNA (CVE Numbering Authorities) – адміністрація з нумерації CVE

CSS (Cascading Style Sheets) – каскадні таблиці стилей

CVE (Common Vulnerabilities and Exposures) – загальні вразливості та ризики

CVSS (Common Vulnerability Scoring System) – загальна система оцінки вразливостей

CWE (Common Weakness Enumeration) – загальний перелік слабких місць

FIRST (Forum of Incident Response and Security Teams) – форум команд реагування на надзвичайні події

FTP (File Transfer Protocol) – протокол передачі файлів

HTML (Hypertext Markup Language) – мова гіпертекстової розмітки

HTTP (Hypertext Transfer Protocol) – протокол передачі гіпертексту

IBM (International Business Machines) – міжнародні бізнес - машини

ID (Identifier) – ідентифікатор

IEC (International Electrotechnical Commission) – міжнародна комісія з електротехніки

IP (Internet Protocol) – міжмережевий протокол

ISSAF (Information Systems Security Assessment Framework) - система оцінки безпеки інформаційних систем

ISO (International Organization for Standardization) – міжнародна організація по стандартизації

LDAP (Lightweight Directory Access Protocol) – полегшений протокол доступу до директорій

NIST (National Institute of Standards and Technology) – національний інститут стандартів та технологій

NVD (National Vulnerability Database) – національна база даних вразливостей

OS (operating system) – операційна система

OSSIG (Open Information Systems Security Group) – група з безпеки відкритих інформаційних систем

OWASP (Open Web Application Security Project) – відкритий проект забезпечення безпеки веб-додатків

PHP (Hypertext Preprocessor) – препроцесор гіпертексту

PTES (Penetration Testing Execution Standard) – стандарт виконання тестування на проникнення

PTF (Penetration Testing Framework) – структура тестування на проникнення

RFI (Remote File Inclusion) – віддалене виконання файлів

SMTP (Simple Mail Transfer Protocol) – простий протокол передачі пошти

SQL (Structured Query Language) – мова структурованих запитів

SSL (Secure Sockets Layer) – рівень захищених сокетів

TF – CSIRT (Task Force – Collaboration Security Incident Response Teams) – робоча група з взаємодії команд реагування на надзвичайні події

TLS (Transport Layer Security) – протокол захисту транспортного рівня

URL (Uniform Resource Locator) – уніфікований локатор ресурсів

US – CERT (Unated States Computer Emergency Response Team) – команда реагування на комп’ютерні надзвичайні події Сполучених Штатів Америки

VND (Vulnerability Notes Database) – база даних записів про вразливості

WASC (Web Application Security Consortium) – Консорціум безпеки веб – додатків

XML (eXtensible Markup Language) – розширювана мова розмітки

XSS (Cross – site Scripting) – міжсайтове иконання сценаріїв

ДСТУ – державний стандарт України

НД ТЗІ – нормативний документ системи технічного захисту інформації

ОС – операційна система

ПЗ – програмне забезпечення

СКБД – система керування базами даних

ВСТУП

Роль веб – додатків має дуже велике значення у сучасному суспільстві та бізнесі. У побудові веб – додатків використовуються різні технології, які швидко розвиваються та вдосконалюються. Проте використання веб – додатків супроводжється ризиками, які можуть привести до порушення безпеки інформації. Це обумовлено тим, що методи, які використовують зловмисники, постійно вдосконалюються. Причиною цього є вразливості компонентів веб – додатків, які можуть існувати в них з початку їх проектування або з'явитися на інших етапах розробки та експлуатації.

Світові організації з кібербезпеки займаються дослідженнями вразливостей веб – додатків, створено стандарти обробки інформації про вразливості, існує широкий вибір засобів виявлення вразливостей та методик виявлення вразливостей. Проте методики пошуку вразливостей є загальними та охоплюють широке коло питань. Таким чином, зазначені методики можуть бути надлишковими при застосуванні для конкретних систем. Отже для пошуку конкретних вразливостей може бути витрачено необгунтовано багато часу.

Об'єкт дослідження: процес тестування на проникнення веб – додатків.

Мета роботи: підвищення ефективності тестування на проникнення веб – додатків.

У роботі проведено аналіз вразливостей та методик тестування на проникнення, структури системи реагування на компютерні надзвичайні події, баз даних та засобів пошуку вразливостей.

Практичне значення роботи полягає у зниженні тривалості та обгрунтуванні вибору засобів тестування на проникнення. Результати здійснених у дипломній роботі досліджень можуть бути використані при тестуванні на проникнення веб – додатків та проведенні лабораторних робіт спеціалізованих курсів спеціальності «Кібербезпека».

Наукова новизна дослідження полягає у адаптації методів тестування на проникнення веб – додатків.

РОЗДІЛ 1
АНАЛІЗ ВРАЗЛИВОСТЕЙ ТА МЕТОДИК ТЕСТУВАННЯ НА ПРОНИКНЕННЯ ВЕБ – ДОДАТКІВ

1.1 Аналіз принципів функціонування веб – додатків та технологій їх побудування

Як відомо, веб – додаток це – клієнт – серверний додаток, у якому клієнт взаємодіє з веб – сервером за допомогою веб – браузера, дані зберігаються на веб – сервері, а обмін даними здійснюється за допомогою комп'ютерних мереж. Клієнтська частина відповідає за формування запитів до веб – сервера та обробку відповідей. Серверна частина відповідає за обробку запитів від клієнта, виконує обчислення, формує відповідь та відправляє її клієнту за допомогою протоколу HTTP (Hypertext Transfer Protocol). Для формування відповідей на запит користувача веб – сервер може використовувати дані, що зберігаються в базах даних [1]. Принцип роботи веб – додатка представлено на функціональній схемі на рисунку 1.1.

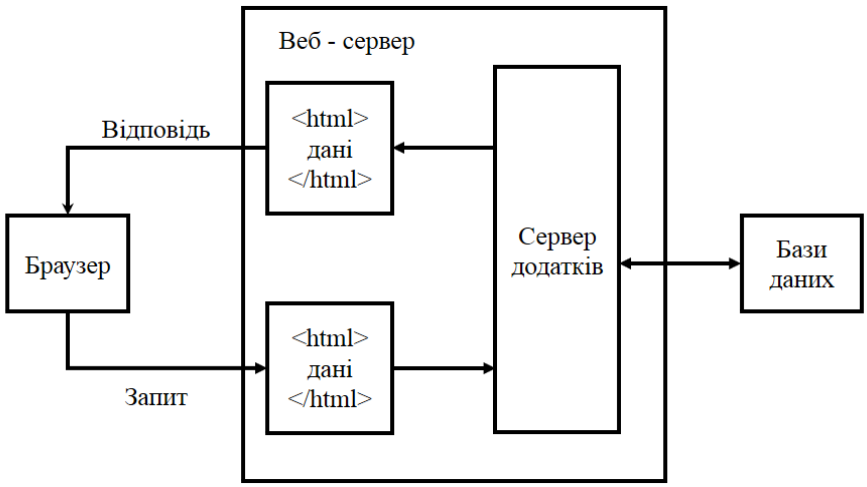

Рисунок 1.1 – Схема функціонування веб – додатка

Для побудови клієнтської та серверної частин додатка використовуються різні технології та мови програмування. Як правило, код серверної частини, що

обробляє запити клієнта, може бути реалізований за допомогою мов програмування [2]:

– Ruby on Rails;
– PHP (Hypertext Preprocessor);
– C#;
– Java;
– Python;
– JavaScript.

На стороні клієнта використовуються [2]:

– таблиці стилей CSS (Cascading Style Sheets);
– мова гіпертекстової розмітки HTML (Hypertext Markup Language);
– мова програмування JavaScript.

Найбільш популярними веб – серверами, що використовуються для функціонування веб – додатків, є [3]:

– Apache;
– Internet Information Services (IIS);
– Nginx;
– Google Web Server.

Таким чином, враховуючи різноманітність технологій, що використовуються у компонентах веб – додатків, виникає необхідність проведення аналізу існуючих вразливостей веб – додатків та способів їх використання потенційним зловмисником.

Класифікація веб – додатків за особливостями їх побудови та функцій може виглядати наступним чином [4]:

– статичні веб – додатки (веб – сторінки такого додатка передаються до клієнтської частині у такому вигляді, у якому зберігаються на сервері);
– динамічні веб – додатки (відповіді сервера формуються залежно від запитів з клієнтської частини);

– веб – додатки у сфері електронної комерції (веб – додатки цього типу включають елементи реєстрації користувачів, а також механізми, що дозволяють проводити платежі через Інтернет);

– веб – портали (цей тип веб – додатків включає елементи, що потребують реєстрації користувача: форуми, онлайн – чати, електрона пошта та ін.);

– веб – додатки з елементами анімації, де використовується технологія Flash;

– веб – додатки з системами управління контентом (WordPress, Joomla, Drupal), які використовуються для постійного оновлення вмісту, призначеного для користувачів веб – додатка.

1.2 Аналіз існуючих вразливостей веб – додатків

Поняття вразливості описане в таких стандартах, як ISO/IEC 29147:2014 Information technology. Security techniques. Vulnerability disclosure (в Україні діє стандарт ДСТУ ISO/IEC 29147:2016 Інформаційні технології. Методи захисту. Розкриття вразливостей) та ISO/IEC 27000:2016. В стандарті ISO/IEC 29147 сказано, що вразливість – це слабке місце в програмному забезпеченні, апаратному забезпеченні або онлайн – сервісі. Слабке місце в системі може бути викликане недоліками в проектуванні програмного та апаратного забезпечення, недоліками в керуванні процесом розробки та ін. [5, с.8]. В стандарті ISO/IEC 27000:2016 вразливість визначається як слабке місце активу або засоба контролю та управління, яке може бути використане однією або кількома загрозами [6, с.20]. В документі НД ТЗІ 1.1 – 003 – 99 сказано, що вразливість системи – нездатність системи протистояти реалізації певної загрози або сукупності загроз [7, с.10].

Класифікацією вразливостей веб – додатків займається Консорціум з безпеки веб – додатків WASC (Web Application Security Consortium). Згідно даних офіційного веб – сайту Консорціуму WASC, він представляє собою неприбуткову організацію, що складається з міжнародних експертів, фахівців з інформаційної безпеки та безпеки мережі Інтернет [8]. Консорціумом WASC

розроблено спеціальний документ, в якому описана класифікація вразливостей веб – додатків – WASC Threat Classification. Згідно документа WASC Threat Classification, вразливості веб – додатків поділяються за наступними етами життєвого циклу програмного забезпечення [9, с.14]:

– етап проектування – охоплює вразливості, які можуть з’явитися внаслідок помилок у проектуванні веб – додатка;

– етап реалізації – охоплює вразливості, які можуть з’явитися через помилки під час реалізації компонентів веб – додатка, наприклад – написання програмного коду;

– етап розгортання – охоплює вразливості, які можуть виникнути під час налаштування веб – додатка до роботи, наприклад – неправильна конфігурація веб – сервера;

Класифікація вразливостей за етапами життєвого циклу веб – додатка вказана в таблиці 1.1 [9, с. 15].

Таблиця 1.1

Фрагмент класифікації вразливостей згідно документа WASC Threat Classification за етапами життєвого циклу веб – додатка

Вразливість	Етапи життєвого циклу веб – додатка		
	Проекту-вання	Реаліза-ція	Розгор-тання
1	2	3	4
Зловживання функціональними можливостями (Abuse of Functionality)	+		
Неправильна конфігурація додатка (Application Misconfiguration)		+	+
Підбір (Brute Force)	+	+	
Переповнення буфера (Buffer Overflow)		+	
Підміна контенту (Content Spoofing)		+	

Продовження табл.1.1

1	2	3	4
Передбачуване значення ідентифікатора сесії (Credential/Session Prediction)		+	
Міжсайтове виконання сценаріїв (Cross-Site Scripting)		+	
Підробка міжсайтових запитів (Cross-Site Request Forgery)	+	+	
Відмова в обслуговуванні (Denial of Service)	+	+	
Індексування директорій (Directory Indexing)			+
Атака на функції форматування строк (Format String)		+	
Підміна HTTP – відповідей (HTTP Response Smuggling)		+	
Розщеплення HTTP – відповідей (HTTP Response Splitting)		+	
Підміна HTTP – запитів (HTTP Request Smuggling)		+	
Розщеплення HTTP – відповідей (HTTP Request Splitting)		+	
Переповнення цілого значення (Integer Overflows)		+	
Неправильне розділення доступу до файлової системи (Improper Filesystem Permissions)		+	+
Неправильна обробка вхідних даних (Improper Input Handling)		+	

Продовження табл.1.1

1	2	3	4
Неправильна обробка вихідних даних (Improper Output Handling)		+	
Витік інформації (Information Leakage)	+	+	+
Незахищене індексування (Insecure Indexing)		+	+
Недостатня протидія автоматизації (Insufficient Anti-automation)	+	+	
Недостатня аутентифікація (Insufficient Authentication)	+	+	
Недостатня авторизація (Insufficient Authorization)	+	+	
Недостатня реалізація механізму відновлення пароля (Insufficient Password Recovery)	+	+	
Недостатня перевірка процесу (Insufficient Process Validation)	+	+	
Недостатня тривалість сеансу (Insufficient Session Expiration)	+	+	+
Недостатній захист транспортного рівня (Insufficient Transport Layer Protection)	+	+	+
Впровадження операторів LDAP (LDAP Injection)		+	
Інєкції через поштові команди (Mail Command Injection)		+	
Додавання нульових байтів до даних (Null Byte Injection)		+	
Виконання команд ОС (OS Commanding)		+	

Продовження табл.1.1

1	2	3	4
Зворотній шлях у директоріях (Path Traversal)		+	
Передбачуване розташування ресурсів (Predictable Resource Location)		+	+
Додавання віддалених файлів (Remote File Inclusion, RFI)		+	+
Обхідний маршрут (Routing Detour)			+
Неправильна конфігурація сервера (Server Misconfiguration)			+
Фіксація сесії (Session Fixation)		+	+
SQL ін’єкції (SQL Injection)		+	
Зловживання перенаправленням URL (URL Redirector Abuse)	+	+	
XPath ін’єкції (XPath Injection)		+	
Неефективна обробка даних XML – аналізаторами (XML Attribute Blowup)		+	
Аналіз XML – вводу (XML External Entities)		+	
Розширення XML – об’єктів (XML Entity Expansion)		+	
XML ін’єкції (XML Injection)		+	
Ін’єкція запитів XQuery (XQuery Injection)		+	

Помилки, що можуть бути допущені під час різних етапів життєвого циклу програмного забезпечення, описані в списку CWE (Common weakness enumeration), який сформовано корпорацією MITRE. Згідно даним веб – сайту корпорації MITRE, CWE є базовим стандартом при ідентифікації слабких місць

в програмному забезпеченні, та запобіганню їх появи [10]. Кожному недоліку у списку присвоєно власний ідентифікатор (ID), список містить близько тисячі CWE ID. При описі вразливості в базах даних вразливостей може бути вказано CWE ID для додаткової інформації. Кожен CWE ID у списку CWE має наступний опис [11]:

– короткий та/або розширений опис про конкретний CWE ID;

– етап, під час якого може бути допущена розробником помилка, яка згодом може привести до виникнення вразливості (етап проектування, експлуатації);

– мова програмування/платформа, які є основою функціонування програмного забезпечення (Java, C++, або будь – яка);

– також в якості прикладу можуть бути наведені фрагменти програмного коду, в якому допущено помилку, яка призводить до неправильного функціонування або виникнення вразливості в програмному забезпеченні;

– посилання на записи в міжнародних базах даних вразливостей, які містять дані про існуючі вразливості в компонентах програм, до яких призвела помилка, що має CWE ID;

– шляхи до виправлення помилки, або правильний варіант фрагмента програмного коду, що гарантує безпеку обробки даних;

– зв’язки з іншими CWE ID, внаслідок яких виникла помилка в конкретному місці програмного забезпечення на конкретному етапі.

Іншим варіантом класифікації вразливостей та ризиків веб – додатків є рейтинг 10 найпоширеніших вразливостей та ризиків – OWASP Top – 10. OWASP – це некомерційна організація, діяльність якої орієнтована на підвищення безпеки програмного забезпечення [12]. Остання редакція документа OWASP Top – 10 випущена організацією OWASP за 2017 рік і виглядає наступним чином [13, с.7]:

– вставка інструкцій (Injection) – відбувається вставка SQL – інструкцій, які передаються на обробку веб – додатку, який після їх отримання може почати виконувати довільні команди;

– некоректна аутентифікація (Broken Authentication) – функції аутентифікації можуть бути реалізовані таким чином, що дозволяють обходити паролі, або отримувати ідентифікатори користувачів;

– витік критичних даних (Sensitive Data Exposure) – недостатня захищеність персональних даних;

– атаки на засоби аналізу XML – вводу (XML Eternal Entities) – зловмисники можуть завантажувати XML – файли на сервер або включати шкідливий код у документ XML;

– неправильний контроль доступу (Broken Access Control) – контроль доступу реалізовано таким чином, що авторизовані користувачі можуть мати в системі такі повноваження, які не повинні мати;

– небезпечна конфігурація оточення (Security Misconfiguration) – відсутність оновлень та неправильна конфігурація окремих компонентів веб – додатків може нести в собі додаткову загрозу безпеці;

– міжсайтове виконання сценаріїв (XSS) – зловмисник отримує можливість виконувати сценарії у браузері жертви, перехоплювати сценарії користувача, перенаправляти користувачів на інші веб – сайти;

– незахищений процес десеріалізації (Insecure Deserialization) – зловмисник може порушати логіку роботи веб – додатка, підробляючи об’єкти додатка, що призводить до віддаленого виконання коду зловмисника;

– використання компонентів з відомими вразливостями (Using Components with Known Vulnerabilities) – програмне забезпечення, у якого термін підтримки вже вичерпаний, або яке є неоновленим, може залучити більше зловмисників до його зламу;

– недостатній моніторинг та ведення журналів подій (Insufficient Logging and Monitoring) – погано організований механізм ведення журналів подій та моніторингу може призвести до того, що зловмисники можуть неодноразово робити атаки на веб – додатки, залишаючись непоміченими.

Згідно досліджень компанії Vercode, яка займається тестуванням захищеності програмного забезпечення, при проведенні тестування на

проникнення у 74 % програмних додатків знаходиться як мінімум одна вразливість зі списку OWASP Top – 10 [14]. Враховуючи ці результати досліджень, виникає необхідність проаналізувати існуючу інформацію про вразливості зі списку OWASP Top – 10 для підвищення ефективності їх виявлення при проведенні тестування на проникнення. Далі наведно більш докладний аналіз вразливостей зі списку OWASP Top – 10.

1.2.1 Вставка інструкцій

Прикладом вставки інструкцій може бути вставка SQL – інструкцій. Вставка SQL – інструкцій, залежно від типу СКБД (системи керування базами даних), що використовується, може дати можливість зловмиснику виконувати будь-який запит до бази даних (читати вміст таблиць, видаляти, змінювати або додавати дані), отримувати можливість читання та/або запису локальних файлів і виконання будь-яких команд на сервері, на який націлений зловмисник. Атака типу «вставка SQL – інструкцій» може бути реалізована через некоректну обробку вхідних даних, що використовуються в SQL – запитах. Далі виконано аналіз прикладу здійснення атаки типу вставка SQL – інструкцій [15]. Якщо існує серверне програмне забезпечення, яке отримує вхідний параметр id, використовує його для створення SQL-запиту, то PHP – скрипт обробки запиту може мати наступний вигляд:

```
$id = $_REQUEST['id'];
$res = mysqli_query("SELECT * FROM news WHERE id_news = " . $id);
```

Якщо на сервер буде передано параметр id, що дорівнює 5 (http://example.org/script.php?id=5), то виконається наступний SQL-запит:

```
SELECT * FROM news WHERE id_news = 5
```

Але якщо зловмисник передасть в якості параметра id рядок -1 OR 1 = 1 (http://example.org/script.php?id=-1+OR+1=1), то виконається запит:

```
SELECT * FROM news WHERE id_news = -1 OR 1 = 1
```

Отже, зміна вхідних параметрів шляхом додавання в них конструкцій мови SQL викликає зміну в логіці виконання SQL-запиту (у прикладі вище замість запису з заданим ідентифікатором будуть обрані всі наявні в базі записи) [15]. На рисунку 1.2 зображено інтерфейс тестового веб – додатка, який дозволяє користувачам виводити списки транзакцій, здійснених за певний період часу. Значення, що вводиться в поле введення дати, не обробляється фільтрами, тому додаток вразливий для вставки SQL-коду [16]. Введення наступного значення дозволяє виконати команду, яка повертає вміст бази даних, включаючи імена і паролі користувачів: 1/1/2010 union select userid, null, username + '' + password, null from users-- .

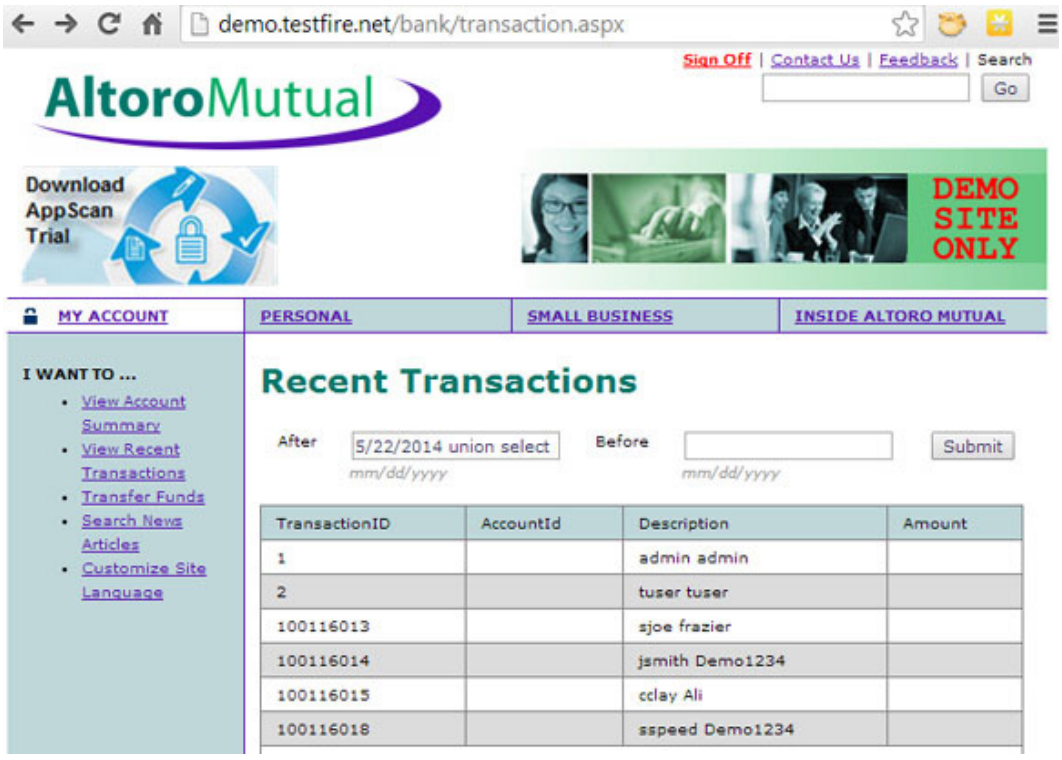

Рисунок 1.2 – Інтерфейс тестового веб – додатка

Згідно рисунку 1.2, перша частина значення, введеного в поле дати, є датою – 1/1/2010. Ці дані обробляються веб – додатком. За датою слідує оператор union, що означає початок SQL – команди. Методом проб і помилок зловмисник може обчислити назву таблиці, яка містить дані про облікові записи користувачів (вона називається users), і кількість полів в ній. Після

цього зловмисник використовує цю інформацію для складання SQL – команди select, що повертає дані з таблиці users. Отримана інформація, що містить імена і паролі облікових записів користувачів, відображається на веб – сторінці в формі для виведення списку транзакцій [16].

1.2.2 Некоректна аутентифікація

Згідно досліджень, зловмисники можуть мати доступ до сотень мільйонів допустимих комбінацій імені користувача та паролів для заповнення форм у веб – додатках, адміністративних списках облікових записів, інструментів перебору значень за словником [17]. Зловмисникам достатньо отримати доступ лише до кількох облікових записів або лише до одного облікового запису адміністратора, щоб зламати систему, вкрасти гроші, використовувати методи шахрайства, викрасти конфіденційну інформацію та ін.

Атаки на засоби аутентифікації можливі, якщо веб – додаток має наступні слабкі місця [17]:

– місця для заповнення облікових даних, коли у зловмисника є список припустимих імен користувачів та паролів;
– відсутній захист від реалізації атак методом перебору;
– дозволено прості або відомі паролі, такі як «password» або «admin/ admin»;
– використовуються неефективні процеси відновлення облікових даних та забутих паролів, які не можуть бути безпечними;
– відсутня багатофакторна аутентифікація;
– надання ідентифікаторів сеансу у URL – адресі.

Прикладом атаки на засоби аутентифікації може бути випадок, коли зловмисник знає секретний ідентифікатор сеансу та може представитись веб-серверу аутентифікованим користувачем і скомпрометувати його обліковий запис. Якщо у веб – додатку недостатньо реалізований захист ідентифікаторів сеансів (відображення ідентифікаторів всередині URL – адреси замість використання cookie – файлів), то зловмиснику дуже просто отримати

ідентифікатор сеансу. Навіть якщо у веб – додатку зберігаються ідентифікатори сеансів у cookie – файлах, зловмисник все одно може отримати потрібну інформацію з локальних файлів користувача, змусивши його виконати замаскований скрипт. За допомогою простого скрипта можна дізнатись ідентифікатор сеансу, що зберігається в cookie – файлі на комп'ютері користувача. Прикладом може слугувати отримання ідентифікатора сеансу за допомогою скрипта, введеного в поле пошуку вразливого веб – додатка: <script> alert (document.cookie) </script>. На рисунку 1.3 зображено приклад тестового веб додатка, де показано результат виконання скрипта для отримання ідентифікатора сеансу [16].

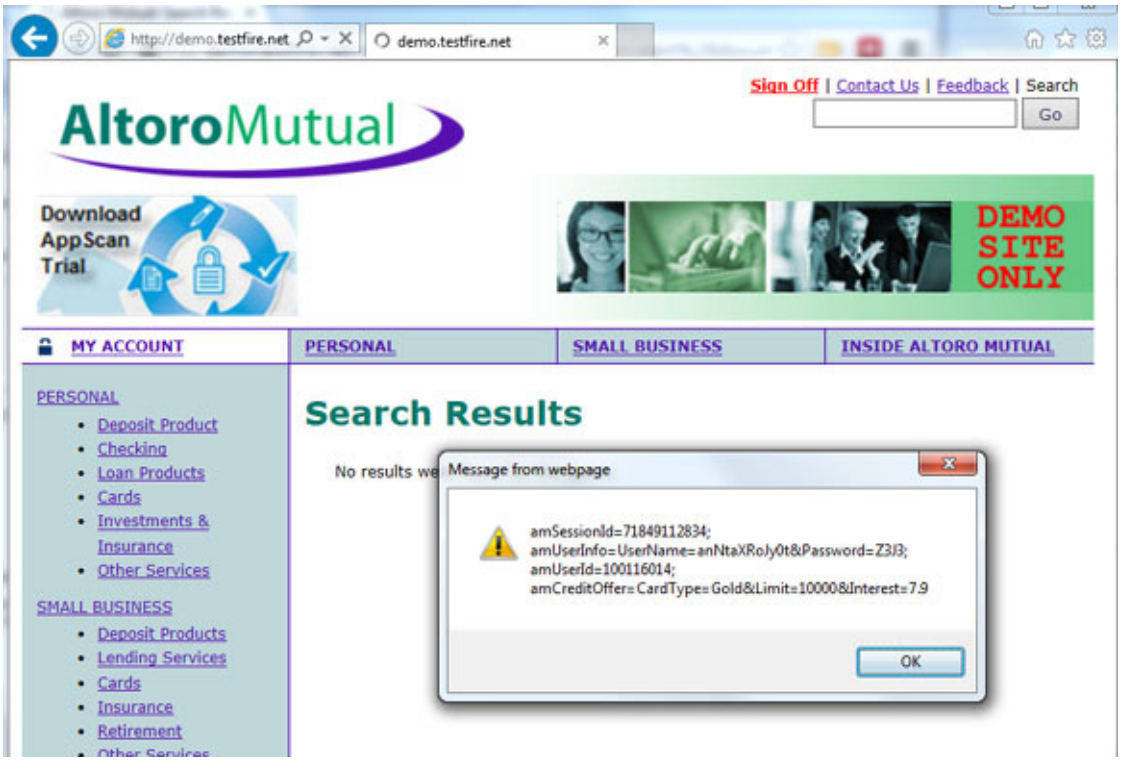

Рисунок 1.3 – Результат роботи скрипта

Для підвищення захищеності системи аутентифікації та управління сеансами у веб – додатках необхідно використовувати сучасні платформи для розробки ПЗ (програмного забезпечення), а також прийоми написання безпечного коду. Зокрема, можна застосовувати прив'язку вихідної IP – адреси до ідентифікатора користувача сеансу і обов'язкову повторну аутентифікацію

при зміні важливих параметрів облікового запису (наприклад, пароля або адреси електронної пошти). Саме ці дії намагаються виконувати зловмисники відразу ж після злому веб – сеансу. У цих випадках такий простий прийом програмування, як повторна аутентифікація, знищує несанкціонований сеанс і знижує ризик зламу [16].

Іншим простим прийомом програмування, який не можна залишати без уваги, є знищення сеансу після закінчення заданого часу простою. Наприклад, якщо користувач не працює з веб – сайтом протягом п'яти хвилин, додаток розриває сеанс, змушуючи користувача (або будь-якого зловмисника) виконувати повторну аутентифікацію для продовження роботи. Згідно досліджень, цей прийом також знижує ризик несанкціонованого використання сеансу та його використовують в більшості банківських веб – додатків [16].

1.2.3 Витік критичних даних

Згідно досліжень, протягом останніх кількох років викрадення критичних даних було найпоширенішою атакою. Головним недоліком при цьому залишається відсутність шифрування конфіденційних даних або коли використовуються нестійткі до зламу криптографічні алгоритми, засоби генерації ключів і управління [18].

Для визначення можливості витоку критичних даних треба визначити, які дані потребують захисту у кожному конкретному випадку під час їх передачі та зберігання. Такими даними можуть бути паролі, номери кредитних карток, медичні записи, особиста інформація та ділова документація або нша конфіденційна інформація. Для всіх цих типів даних необхідно перевірити [18]:

– чи є такі дані, що передаються у вигляді відкритого тексту (це стосується таких протоколів, як HTTP, SMTP та FTP);

– чи використовуються старі або нестійкі до зламу криптографічні алгоритми у системі або у програмному коді;

– як реалізовано механізм розподілення та керування криптографічними ключами;

– чи використовується поштовим клієнтом недійсний сертифікат отриманий від сервера.

Одним з сценаріїв витоку критичних даних може бути процес шифрації програмою номерів кредитних карток у базі даних за допомогою автоматичного шифрування бази даних. Проте ці дані автоматично розшифровуються при завантаженні, дозволяючи застосовувати вставки SQL – інструкцій для отримання номерів кредитних карт у вигляді відкритого тексту. Іншим сценарієм витоку критичних даних може бути веб – сайт, у якому не використовується протокол захисту транспортного рівня TLS (Transport Layer Security) для всіх сторінок або використовуються нестійкі до зламу криптографічні алгоритми шифрування. Зловмисник може здійснити моніторинг мережевого трафіку (наприклад, в незахищеній бездротовій мережі), знизити рівень зв'язків з HTTPS до рівня HTTP, перехопити запити та викрасти дані з cookie користувача. Нападник потім відтворює цей файл cookie і перехоплює сеанс (аутентифікований) користувача, отримуючи доступ або змінюючи особисті дані користувача. Крім того, у такий спосіб може змінюватися одержувач грошового переказу. Також до витоку критичних даних може призводити використання простих алгоритмів обчислення хешу пароля. В такому випадку всі хеші можуть бути обчислені за допомогою райдужних таблиць попередніх хешів за допомогою графічних процесорів [18].

1.2.4 Атаки на засоби аналізу XML – вводу

XML (eXtensible Markup Language) – це стандарт для обміну структурованими даними у текстовому форматі.

Вміст XML – файлу може мати наступний вигляд [19]:

```
version="1.0" encoding="UTF-8"?>
<order>
```

```
    <product>1234</product>
    <count>1</count>
    <orderer>
        <contact>Jan P. Monsch</contact>
        <account>789</account>
    </orderer>
</order>
```

Можливі цілі при здійсненні атаки:

– мережевий сервіс;
– генератор XML;
– XML-аналізатор;
– код програми.

На рисунку 1.4 зображено можливі місця здійснення атаки на засоби аналізу XML – вводу [19].

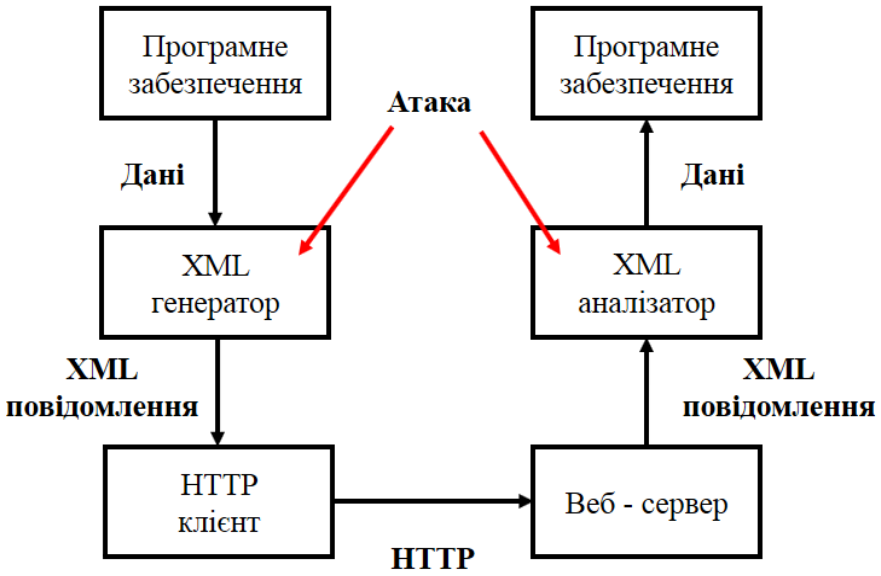

Рисунок 1.4 – Можливі місця здійснення атаки
на засоби аналізу XML – вводу

Нижче наведено можливі приклади організації атак на засоби аналізу XML – вводу [20]. На рисунку 1.5 представлено приклад визначення типу

даних, що називається foo, з елементом bar, з яким зв'язане слово «World». Отже, в будь – який час синтаксичний аналізатор XML замінить цей об'єкт словом «World».

Рисунок 1.5 – Приклад передачі XML – елементів до аналізатора

Згідно досліджень, XML – об'єкти можуть використовуватися зловмисниками для атак «Відмова в обслуговуванні» шляхом вставки об'єктів з елементами, всередині яких знаходяться інші елементи і т.д [20]. На рисунку 1.6 представлено приклад запиту до XML – аналізатора та результат обробки запиту.

Рисунок 1.6 – Приклад передачі до XML – аналізатора обєкта з вкладеними елементами та результат обробки запиту

Посилаючи необхідні запити до XML – аналізатора, зловмисник, який володіє інформацією про структуру веб – додатка, може дізнатися інформацію про програмне забезпечення, що використовується [20]. Приклад можливих

інструкцій у XML – файлі та результат роботи XML – аналізатора наведено на рисунку 1.7.

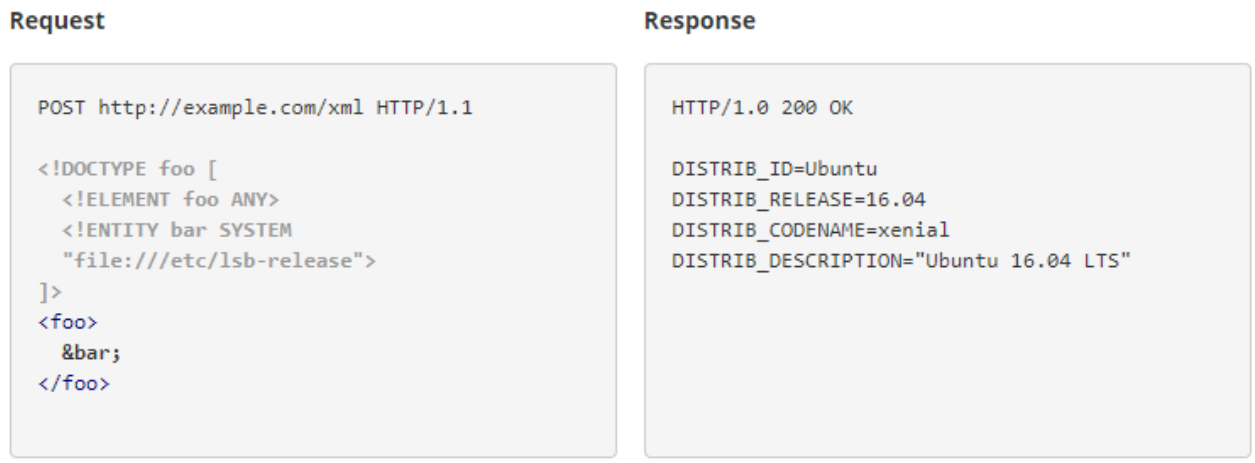

Рисунок 1.7 – Приклад передачі до XML – аналізатора об’єкта з інструкціями на визначення операційної системи

1.2.5 Неправильний контроль доступу

Призначення контролю доступу – забезпечити виконання вимог політики безпеки таким чином, щоб користувачі системи не змогли виконувати у системі дії, які виходять за межі повноважень користувачів. Прикладом контролю доступу може бути обження доступу до ресурсів неавторизованих користувачів [21]. Проте, як відомо, обмеження доступу до ресурсів може стосуватися не лише файлів та баз даних, а також [22]:

– програм;
– спеціальних функцій додатків;
– спеціальних полів даних;
– пам'яті;
– приватних або захищених змінних;
– носіїв інформації;
– засобів передачі інформації.

Недоліки у засобах контролю доступу, як правило, призводять до несанкціонованого розкриття інформації, модифікації або знищення всіх даних

або виконання бізнес-функцій за межами користувача. Загальні вразливості засобів контролю доступу включають [21]:

– обхід перевірок контролю доступу шляхом зміни URL – адреси, внутрішнього стану програми, HTML – сторінки або просто використання спеціального інструмента атаки на програмний інтерфейс;

– можливість за допомогою первинного ключа змінювати всі записи інших користувачів, що дозволяє переглядати чи редагувати чужі облікові записи;

– підвищення привілеїв (повноваження користувача в системі без проходження етапів авторизації, або повноваження адміністратора після входу у систему в якості користувача;

– маніпулювання метаданими, такими як повторне використання або підміна токенів керування доступом або файлів cookie, за допомогою яких можна підвищувати привілеї у системі.

1.2.6 Небезпечна конфігурація оточення

Згідно даних досліджень організації OWASP, можливість неправильної конфігурації оточення, тобто середовища функціонування веб – додатка, може статися на будь-якому рівні компонентів додатка, включаючи мережеві сервіси, платформу, веб-сервер, сервер додатків, базу даних попередньо встановлені віртуальні машини та ін. Автоматизовані сканери вразливостей здатні виявляти неправильні конфігурації, використовувати облікові записи або стандартні конфігурації, тощо. Доступність автоматизованих засобів пошуку вразливостей допомагають зловмисникам здійснити несанкціонований доступ до деяких системних даних або функціональних можливостей. Інколи такі недоліки приводять до повного розкриття структури системи та викрадення конфіденційних дани та ін. [23]. Додаткову загрозу несе наявність програмного забезпечення, яке не оновлюється. Можливі сценарії атак через небезпечну конфігурацію:

– на сервері додатків встановлено програмне забезпечення, яке не вилучене після закінчення проектування або тестування системи (це ПЗ містить відомі вразливості, які можуть використовувати нападники) та тестові облікові записи не були змінені;

– конфігурації сервера додатків дозволяють отримувати докладні повідомлення про помилки (може бути відображена інформація про ПЗ, що використовується, версія та зловмисник може виконати пошук вразливостей до вказаного ПЗ).

1.2.7 Міжсайтове виконання сценаріїв

Атаки на веб – системи, в яких є вразливість до міжсайтового виконання сценаріїв полягають у тому, що виконується вставка коду зловмисника у сторінку веб – додатка, яка буде виконана на комп'ютері користувача під час відкриття ним цієї сторінки) та відбуваєтья взаємодія цього коду з веб – сервером зловмисника. Даний тип атаки відомий як XSS – атаки [24].

На веб – сайті компанії Acunetix, яка займається розробкою рішень для підвищення захищеності веб – додатків визначено схему здійснення XSS – атаки, предствлену на рисунку 1.8 [24].

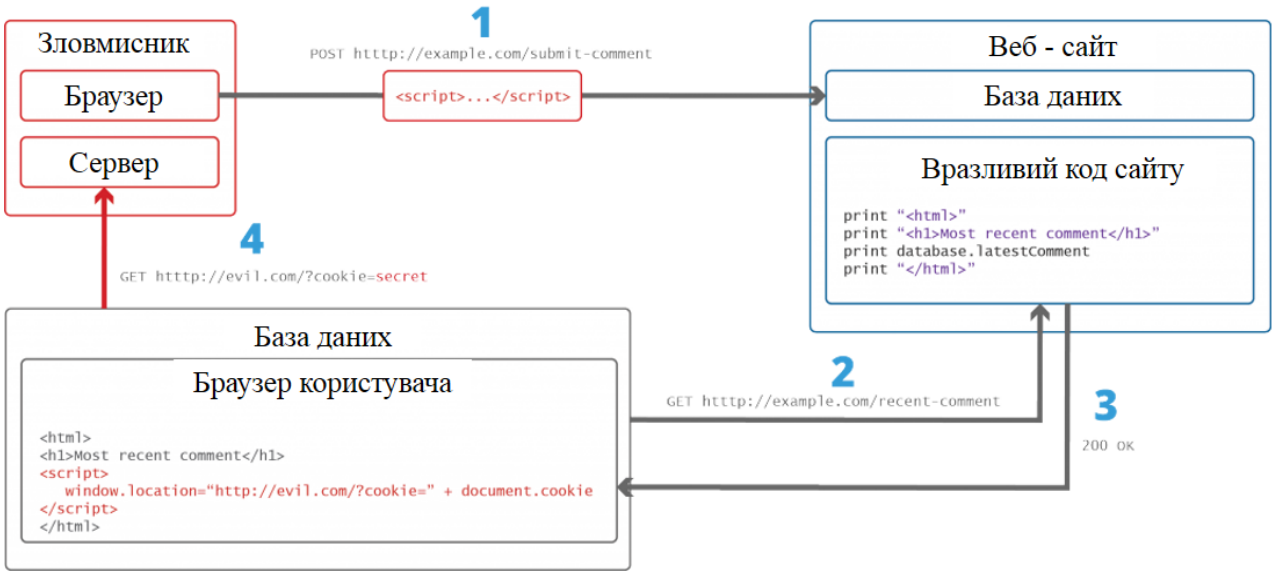

Рисунок 1.8 – Схема здійснення XSS – атаки

На рисунку 1.8 цифрами позначено:

1 – зловмисник знаходить вразливий веб-сайт, вставляє у форму вводу даних скрипт;

2 – користувач формує запит веб-сторінки з веб-сайту;

веб-сервер надсилає користувачу сторінку з кодом зловмисника як частину HTML – документу;

3 – браузер користувача виконує код зловмисника всередині HTML документу та відправляе cookie користувача на сервер зловмисника, після чого зловмисник

4 – може використати вкрадений cookie – файл жертви для висування себе за іншу особу;

Для того, щоб код зловмисника виконався на сторінці користувача, зловмисник може використовувати техніки соціальної інженерії з метою переконати користувача відкрити сторінку з завантаженим скриптом.

Враховуючи можливі збитки від здійснення атаки такого типу, XSS – атаки увійшли до списку OWASP Top – 10 та є одними з найпоширеніших атак.

1.2.8 Незахищена десеріалізація

Процес серіалізації використовується для передачі об’єктів по мережі та збереження їх у файлах. Для цього об’єкт заповнюється потрібними данними, потім викликається код серілізації, і результатом є або XML – документ, або документ іншого формату. Результат серіалізації передається до приймаючої сторони електроною поштою або HTTP. Одержувач створює об’єкт того ж типу, який був до серіалізації, цим самим проводячи процес десеріалізації. Згідно досліджень компанії Acunetix, більшість загроз безпеки інформації виникає у процесі десеріалізації, коли замість відправленого серіалізованого обєкта буде отримано схожий об’єкт, але з кодом зловмисника.

Успішна десеріалізація обєкта зловмисника може призвести до таких атак, як «відмова в обслуговуванні», підміна даних користувача у процесі аутентифікації, віддалене виконання коду [25].

За даними організації OWASP, серіалізація може використовуватися у наступних елементах веб – додатків [26]:

– елементи розподілених систем;
– мережеві протоколи, веб – сервіси;
– засоби кешування;
– бази даних, файлові системи;
– HTTP – файли cookie, параметри HTML – форм, токени аутентифікації.

1.2.9 Використання компонентів з відомими вразливостями

В структурі веб – додатка використовується багато компонентів, виробники яких можуть відрізнятися. По перше, при використанні різних компонентів при побудові веб – додатка слід враховувати, що при створенні компонентів на кожному з можливих етапів виробником можуть бути допущені помилки, що можуть призводити до виникнення вразливостей у цих компонентах. Компонентами веб – додатків можуть бути реалізації різних сервісів, сторонні бібліотеки, системи керування контентом, реалізації веб – серверів, операційні системи тощо.

По друге, якщо вразливість у компонента існує, вона може бути знайдена виробником компонента, користувачем, або зловмисником. Завдання виробника полягає у виправленні вразливості. Але виправлення вразливості може зайняти деякий час, який можуть використати зловмисники для проведення атак на користувачів програм, які містять компоненти з вразливостями. Слід зауважити, що у випадку виробника або користувача інформація про вразливість може бути розкрита для суспільства після випуску оновлення для компонента (користувач знайшов вразливість та повідомив виробнику). Проте, якщо користувач знайшов у компоненті вразливість та

використовує цю інформацію для експлуатації цієї вразливості, його можна вважати зловмисником. У випадку зловмисника інформація про вразливість може бути доступна на спеціальних форумах для зловмисників. Також слід додати те, що багато користувачів не встановлюють оновлення для компонентів програм, що використовують, піддаючи ризику систему та власні дані. Враховуючи доступність інформації про вразливості для суспільства у вигляді відкритих баз даних вразливостей, доступність методик пошуку вразливостей та їх експлуатації, а також програмних засобів використання вразливостей, перевірка наявності компонентів, що можуть містити відомі вразливості є обов'язковою.

1.2.10 Недостатній моніторинг та ведення журналів подій

Як відомо, використання зловмисниками можливої недостатньої реалізації процесу моніторину та ведення журналів подій є основою майже всіх основних інцидентів. Одною з стратегій по визначенню того, чи достатній рівень моніторингу реалізовано – це перевіряти журнали реєстрації подій після проведення тестування на проникнення. Дії тестувальників повинні реєструватися таким чином, щоб зрозуміти, які збитки можуть нанести дії потенційних зловмисників. Згідно досліджень, у 2016 році час на виявлення вторгнень складав 191 день – достатній час для нанесення збитків [27].

Прояви реалізації недостатнього моніторингу та ведення журналів подій відбуваються, коли [27]:

– невдалі спроби введення логінів, паролів та транзакції не реєструються;
– засобами реєстрації подій не передбачена фіксація підозрілої активності;
– журнали реєстрації подій зберігаються лише локально;
– під час проведення тестування на проникнення не виникає повідомлень про спроби вторгнення у систему;
– повідомлення про помилки реєструються у журналах подій без чітких формулювань особливостей подій.

1.3 Аналіз структури системи реагування на комп'ютерні надзвичайні події

Експлуатація зловмисниками вразливостей, які можуть містити веб – додатки, може призвести до втрати даних, нанесення збитків як звичайним користувачам, так і великим компаніям, отже необхідно проаналізувати світові та вітчизняні організації, що займаються питаннями кібербезпеки з метою визначення найкращих практик по захисту веб – додатків.

Питаннями, що відносяться до кібербезпеки, дослідженням вразливостей в програмних продуктах, розробкою стандартів для мов програмування, які підвищують безпеку використання цих мов займаються CERT (Computer Emergency Response Team) – команди реагування на комп'ютерні надзвичайні події [28].

В Сполучених Штатах Америки діє команда US – CERT (Unated States Computer Emergency Response Team), на території Європейського Союзу – TF-CSIRT (Task Force – Collaboration Security Incident Response Teams) [29].

В Україні існує команда реагування на комп'ютерні надзвичайні події, відома як CERT – UA. CERT – UA є структурним підрозділом Державної служби спеціального зв’язку та захисту інформації України [30]. CERT – UA займається [31]:

– накопиченням та аналізом даних про кіберзагрози;
– оцінкою захищеності державних інформаційних ресурсів;
– участю у форумі команд реагування на інциденти інформаційної безпеки;
– наданням допомоги з питань протидії кіберзагрозам.

Також слід зазначити, що за даними веб – сайту Національного банку України, ведуться роботи зі створення центру реагування на інциденти кібербезпеки у банківській системі та платіжному просторі України (CERT – NBU) [32]. Крім того, за даними веб – сайту Дніпровської місцевої ради, у місті Дніпро також відкрито центр реагування на кібератаки, в якому працюють спеціалісти управління інформаційних технологій [33]. На рисунку 1.9 зображено схему обміну інформацією про кіберінциденти в Україні.

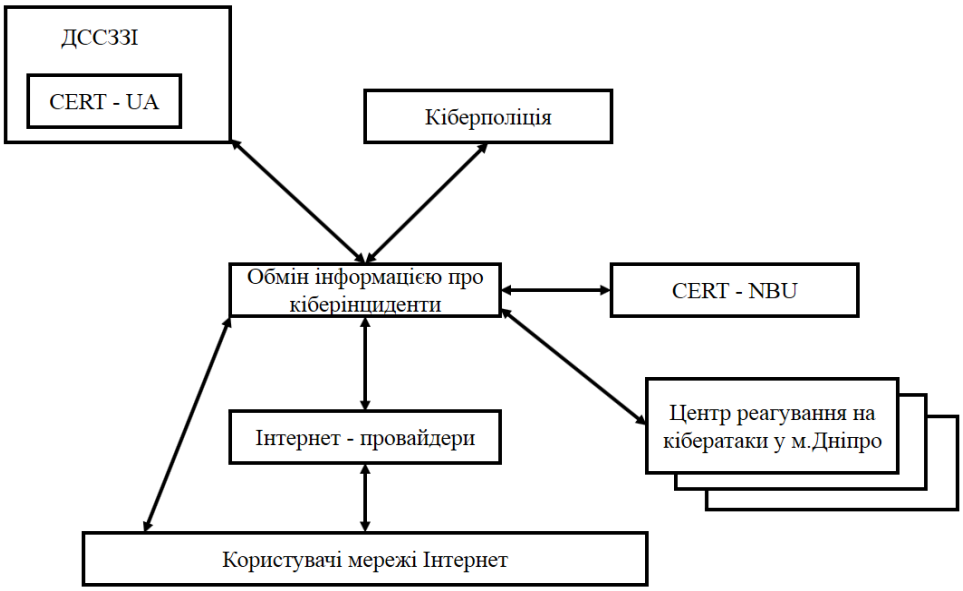

Рисунок 1.9 – Схема обміну інформацією про кіберінциденти в Україні

Слід додати, що 15 березня 2016 року було затверджено Стратегію кібербезпеки України [34]. Отже, в Україні на даний час існує система реагування на кіберінциденти, вдосконалення якої полягає у розробці нормативних документів, що охоплюють питання кібербезпеки та розвитку технологій захисту від кібератак.

Для обміну інформацією кіберінциденти було створено форум FIRST (Forum of Incident Response and Security Teams), що складається з команд реагування на комп'ютерні надзвичайні події [35].

Форумом FIRST розроблено загальну систему оцінки вразливостей CVSS (Common Vulnerability Scoring System) [36].

1.4 Аналіз загальної системи оцінки вразливостей CVSS

Якщо компанія, яка займається розробкою програмного забезпечення, володіє інформацією про наявність існуючих вразливостей в своєму продукті, виникає необхідність правильно розставити пріоритети при виправленні критичних вразливостей в першу чергу. Система CVSS розроблена з метою обчислення числового значення кожної знайденої вразливості, яке

використовується при визначенні пріоритетів при виправленні знайдених вразливостей [37, с.5].

Система оцінки CVSS складається з 3 типів метрик: базових, часових та контекстних метрик. Значення кожної метрики оцінюється по шкалі від 0,0 до 10,0 балів. Базові метрики відображають характеристики вразливості які не змінюються з часом: вектор атаки, складність атаки, необхідні повноваження, взаємодія з користувачем, вплив на конфіденційність, вплив на цілісність, вплив на доступність. Часові метрики відображають характеристики вразливості, які змінюються з часом, контекстні метрики представляють характеристики вразливості, які є унікальними для певного середовища, в якому знайдено вразливість [37, с.6]. Кожна з характеристик метрики має декілька значень, які описують можливі рівні її критичності. Так, згідно з специфікації системи CVSS версії 3, характеристика, що описує вплив на конфіденційність, відображає вплив успішної експлуатації вразливості на збереження конфіденційності в системі. Варіанти значень цієї характиристики наведені в таблиці 1.2 згідно з специфікації системи CVSS версії 3. Значення зростає із ступенем втрат від впливу на вразливий компонент [37, с.11].

Таблиця 1.2 – Значення впливу успішної експлуатації вразливості на збереження конфіденційності

Рівень впливу на збереження конфіденційності	Опис
Високий	Повна втрата конфіденційності, всі ресурси стають відкритими для зловмисника.
Низький	Розкриття інформації не призводить до серйозного збитку від впливу на вразливий компонент.
Вплив відсутній	Втрати конфіденційності відсутні

Кожне значення впливу на збереження конфіденційності, вказане в таблиці 1.2, має свій числовий коефіцієнт. Дані про числові коефіцієнти наведено в таблиці 1.3 [37, с.20].

Таблиця 1.3 – Значення числових коефіцієнтів для кожного випадку впливу на збереження конфіденційності

Назва значення	Числовий коефіцієнт
Високий (High, H)	0,56
Низький (Low, L)	0,22
Вплив відсутній (None, N)	0

Згідно специфікації системи CVSS версії 3, при описі вразливості таким чином обчислюються всі метрики та їх числові коефіцієнти, які підставляються у результуючий вираз, результатом обчислення якого є оцінка вразливості по шкалі від 0,0 до 10,0. Залежно від значення оцінки для вразливості виставляється рейтинг, значення якого вказані в таблиці 1.3 [37, с.16]:

Таблиця 1.4.3 Значення рейтингу вразливості

Рейтинг вразливості	Оцінка CVSS
Немає	0,0
Низький	0,1 – 3,9
Середній	4,0 – 6,9
Високий	7,0 – 8,9
Критичний	9,0 – 10,0

Таким чином, представлена система дозволяє оцінити числове значення критичності кожної знайденої вразливості, яке допомагає правильно розставити пріоритети у виправленні вразливостей та у організації процесів пошуку

вразливостей й оцінюванні ризиків від їх експлуатації. Числові оцінки критичності вразливостей також описані у базах даних вразливостей.

1.5 Аналіз існуючих баз даних вразливостей

Однією з найпоширеніших баз даних вразливостей є база даних CVE (Common Vulnerabilities and Exposures), розроблених корпорацією MITRE. Згідно інформації офіційного веб сайту бази даних CVE, цей проект було запущено у 1999 році, коли одна й та ж вразливість, знайдена різними дослідниками, мала різний опис і виникла необхідність у створенні єдиного стандарту назв та опису вразливостей для більш ефективної обробки інформації та прискорення їх виправлення [38]. Проект CVE спонсується організацією з реагування на комп'ютерні надзвичайні події US – CERT.

Згідно стандарту опису вразливостей, кожній вразливості присвоюється свій ідентифікатор, який має такий вигляд: CVE – YYYY – NNNN, де [39]:

– CVE – префікс;
– YYYY – рік виявлення вразливості;
– NNNN – порядковий номер (послідовність з 4 і більше цифр).

Кожен запис про вразливість включає [39]:

– CVE – ідентифікатор;
– короткий опис вразливості;
– посилання на інші ресурси, що мають відношення до виявлення вразливості.

Процес додавання вразливості в базу містить три етапи:

– обробка – дослідження і процес приведення вразливості до формату CVE;
– присвоєння – призначення вразливості ідентифікатора CVE;
– публікацію – створення запису і його публікація на інтернет-ресурсі CVE.

Короткий опис вразливості складається співробітниками спеціального відділу компанії MITRE (MITRE's CVE Content Team), які аналізують звіти про знайдені вразливості, досліджують будь-яку суперечливу інформацію або

несумісне використання термінології, а потім складають опис вразливості, що включає всю необхідну інформацію, щоб користувачам легше було знайти вразливість по ідентифікатором або розрізнити схожі вразливості [40]. Право на присвоєння CVE – ідентифікаторів вразливостям також мають організації, що мають на це повноваження та входять до списку CNA (CVE Numbering Authorities). Станом на листопад 2017 року до списку CNA, розташованого на офіційному веб – сайті CVE MITRE, входить 81 організація, така як Adobe Systems Incorporated, Apple Inc., Cisco Systems, Inc., IBM Corporation, Microsoft Corporation, Oracle та ін. [41]. Окремо на сайті проекту CVE MITRE розміщена контактна інформація та електронна пошта організацій, які входять до списку CNA, призначена для дослідників, які можуть через вказану поштову адресу повідомити про найдену вразливість. Фрагмент контактної інформації організацій CNA розміщено на рисунку 1.10 [42].

Microsoft	Microsoft Corporation	secure@microsoft.com
Mozilla	Mozilla Corporation	security@mozilla.org
Novell	Micro Focus International	security@suse.com
Nvidia	N/A	psirt@nvidia.com
Objective Development Software	https://obdev.at/go/cna	N/A
OpenSSL	OpenSSL Software Foundation	openssl-security@openssl.org
Oracle	Oracle	secalert_us@oracle.com

Рисунок 1.10 – Фрагмент контактної інформації організацій CNA

Іншим прикладом бази даних вразливостей є база даних CVE Details, яка відображає критичність вразливостей за допомого числових оцінок, розрахований системою CVSS. За даними офіційного веб – сайту бази даних CVE Details, статистична інформація про кількість записів в базі даних та їх оцінки, виглядає наступним чином, як вказано на рисунку 1.11 [43].

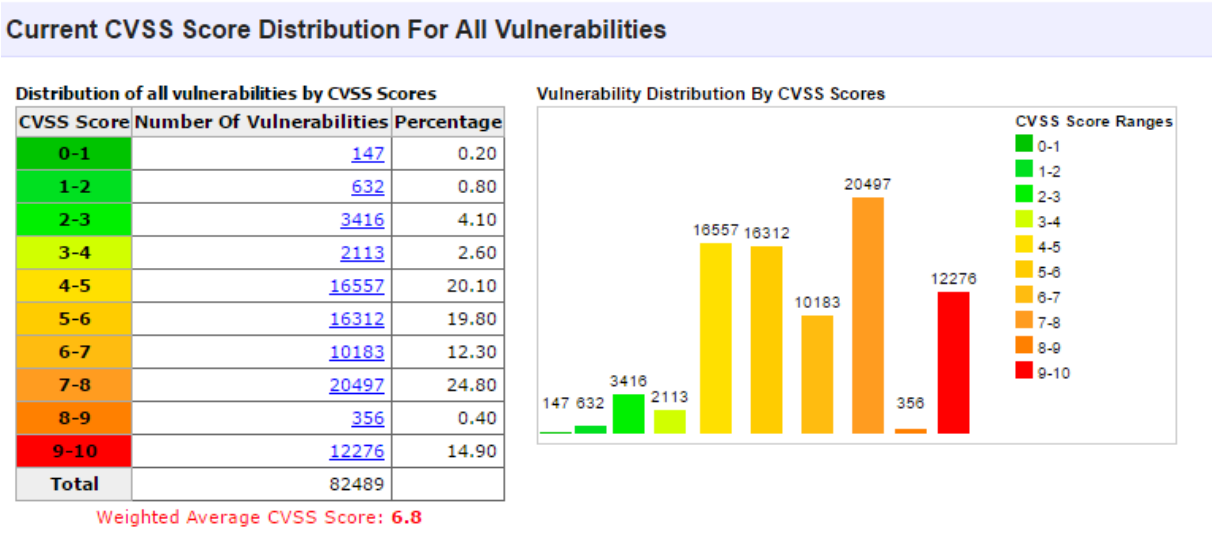

Рисунок 1.11 – Фрагмент головної сторінки бази CVE Details

До інших великих центрів детектування вразливостей відносяться:

– Національний інститут стандартів та технологій ((National Institute of Standards and Technology — NIST) та Національна база данных вразливостей (National Vulnerabilities Database — NVD) [44];

– база записів про вразливості (Vulnerability Notes Database — VND) організації US – CERT [45];

– база даних SecurityFocus [46].

Бази даних вразливостей дозволяють структурувати інформацію про вразливості та надають корисні рекомендації щодо їх виправлення у доступній формі для користувачів, експертів з кібербезпеки. Завдяки цьому можна знизити ризик використання вразливостей зловмисниками.

1.6 Огляд особливостей процесу розкриття вразливостей

Процес розкриття інформації про знайдені вразливості описано у стандарті ISO/IEC 29147:2014 Інформаційні технології. Методи захисту. Розкриття вразливостей (ISO/IEC 29147:2014 Information technology. Security techniques. Vulnerability disclosure) [47].

Згідно стандарту ISO/IEC 29147:2014, в процесі розкриття вразливостей приймають участь сторони [47, с. 6 – 7]:

– особа, що знайшла вразливість (дослідник);
– координатор (особа, яка приймає участь у процесі розкриття вразливостей, координує дії виробників програмного забезпечення та осіб, що знайшли вразливість);
– виробник програмного забезпечення;
– інші виробники програмного забезпечення (виробники, що використовують програмне забезпечення іншого виробника, яке має вразливість);
– користувач.

Згідно стандарту ISO/IEC 29147, обмін інформацією про знайдену вразливість здійснюється за схемою, позначеною на рисунку 1.12 [47, с. 9]:

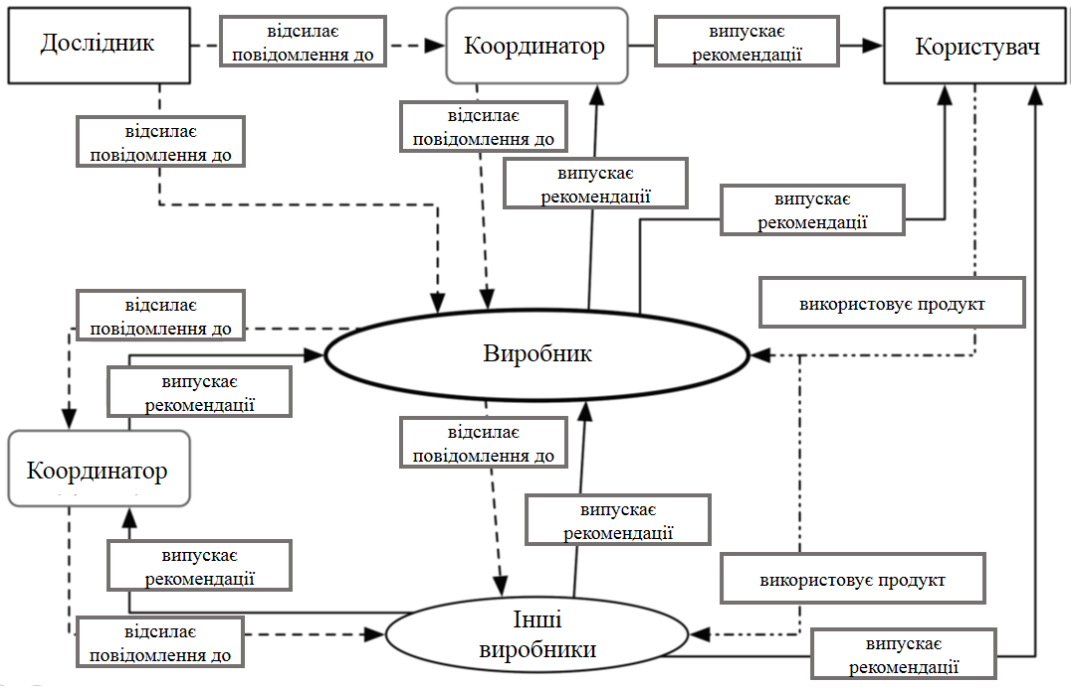

Рисунок 1.12 – Схема обміну інформацією про вразливість

Оскільки пошуком вразливостей можуть займатися не тільки співробітники компаній, що займаються питаннями підвищення безпеки програмного забезпечення, а й зловмисники, у стандарті ISO/IEC 29147:2014

зазначено, що кожна компанія – виробник програмного забезпечення повинна мати свою політику розкриття вразливостей, тобто політику розголошення інформації про вразливість користувачам. Також, оскільки при створенні програмного забезпечення може використовуватися інше програмне забезпечення, сторонні програмні бібліотеки, необхідна взаємодія з усіма виробниками компонентів програмного забезпечення [47].

Прикладом керування процесом розкриття вразливостей може бути документ «Процес керування вразливостями в захисті програмного забезпечення», розроблений компанією Symantec. Згідно цього документа, існує декілька видів розкриття інформації про вразливість[48] :

– повне розкриття (вся інформація про вразливість, назва програмного продукту, фрагмент програмного коду, або дії, які призвели до знаходження вразливості, публікуються одразу після знаходження вразливості);
– відповідальне розкриття (коли фахівцям компанії Symantec стає відома інформація про існування вразливості в конкретному програмному забезпеченні, перш за все інформується виробник цього програмного забезпечення і йому надається від 40 до 180 днів на виправлення вразливості та випуску оновлення, після цього інформація про вразливість публікується для користувачів).

Іншим джерелом інформації про вразливості є програми Bug Bounty, які пропонують виробники програмного забезпечення. Участь в програмі Bug Bounty надає можливість отримати винагородження за знайдені вразливості у програмних продуктах [49]. Програми Bug Bounty реалізовані Apple, Amazon, Google, Microsoft та ін.

Реалізація схеми обміну інформацією про знайдені вразливості на рисунку 1.12 може бути застосована і в Україні, прикладом дослідників вразливостей можуть бути вітчизняні експерти з кібербезпеки, прикладом координатора – центр реагування CERT – UA, який може випускати та доповнювати рекомендації виробників користувачам для виправлення вразливостей та зменшення загрози їх використання. Оскільки вразливості

компонентів систем несуть загрозу їх використання зловмисниками, необхідно розглянути існуючі засоби, які дозволяють шукати можливі вразливості.

1.7 Аналіз існуючих засобів пошуку вразливостей

Для пошуку вразливостей використовуються спеціальні сканери вразливостей. До основних завдань сканерів вразливостей відносяться [50]:

– ідентифікація та аналіз загроз;
– інвентаризація ресурсів (операційна система, програмне забезпечення, мережеві пристрої);
– формування звітів з описом вразливостей та варіанти виправлення вразливих місць системи.

Існує два основних механізми роботи сканерів вразливостей [50]:

– сканування – пасивний аналіз, що представляє собою збір інформації про порти, операційні системи, версії програмного забезпечення, отримані дані порівнюються з правилами визначення інформації про порти, програмне забезпечення та ін. компоненти системи та формується висновок про наявність або відсутність вразливостей;
– зондування – активний аналіз, що представляє собою імітацію атаки на вразливість, яка перевіряється.

Існують різні засоби пошуку вразливостей для різних етапів перевірки систем на наявність вразливостей, від сканерів портів до комплексних систем, які складаються зі сканерів портів, засобів пошуку експлойтів для вразливостей. Прикладами засобів пошуку вразливостей можуть бути:

– програма для дослідження мережі Nmap – дозволяє визначати хости в мережі, встановлені служби та їх версію, які операційні системи та версії вони використовують, пакетні фільтри/міжмережеві екрани [51];
– комплекс засобів перевірки веб – додатків Burp Suite – має власний проксі сервер, сканер вразливостей, засіб автоматизації атак [52];

– Metasploit Framework – інструмент для перевірки та експлуатації вразливостей [53].

Засоби пошуку та експлуатації вразливостей можуть бути також зібрані у спеціальних операційних системах, призначених для аналізу захищеності та проведення тестувань комп'ютерних систем на проникнення. Існують наступні операційні системи для проведень тестування комп'ютерних систем на проникнення:

– ОС Kali Linux [54];
– ОС BlackArch Linux [55];
– ОС Parrot Security OS [56];
– ОС BlackBox [57];

Приклад звіту про знайдену вразливість сканером Nessus вказано на рисунку 1.13 [58].

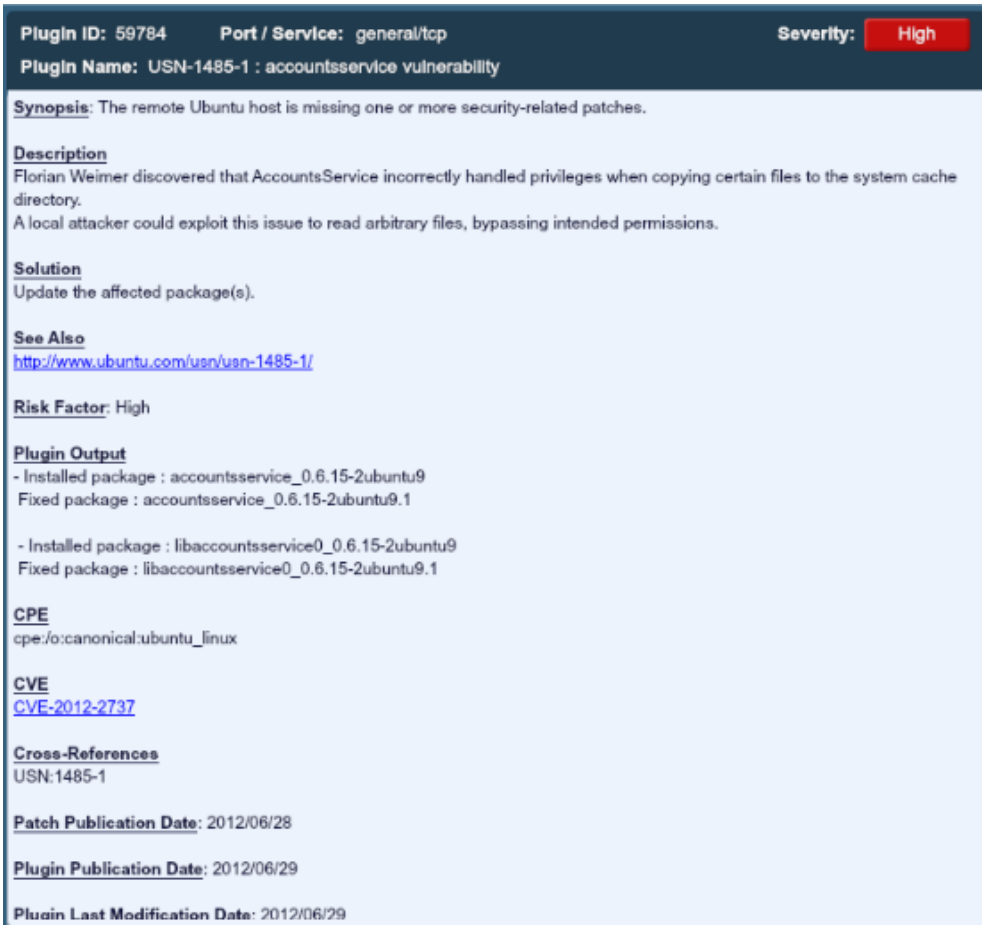

Рисунок 1.13 – Приклад звіту про знайдену вразливість знайдену сканером Nessus

Таким чином, існують наступні варіанти пошуку вразливостей – використання комплексних комерційних рішень, таких як Nessus, або використання засобів, що входять до спеціальних ОС, таких як Kali Linux, та ін. Проте, використання спеціальних ОС для пошуку вразливостей та проведення тестувань на проникнення має на увазі дотримання певного алгоритму застосування вбудованих у ОС засобів з метою підвищення ефективності пошуку вразливостей. Враховуючи це, необхідно проаналізувати існуючі методики пошуку вразливостей та тестування на проникнення.

1.8 Аналіз існуючих методик тестування на проникнення веб – додатків

Згідно термінології НД ТЗІ 1.1 – 003 – 99, тестування на проникання – випробування, метою яких є здійснення спроби обминути або відключити механізми захисту [7, с.13]. Як правило, сценарій тестування на проникнення виглядає наступним чином [59]:

– планування тесту на проникнення;
– збір інформації про цільові системи;
– пошук вразливостей;
– проникнення в систему;
– складання звітів;
– очищення систем від наслідків тесту.

Враховуючи це, існує декілька підходів до проведення тестування на проникнення [59]:

– white box – виконавець проведення тестування на проникнення має доступ до системи та має в своєму розпорядженні повну інформацію про її побудову;
– grey box – імітація дій зловмисників по зламу системи, які мають часткову інформацію про систему (діапазони IP – адрес, ідентифікатори бездротових мереж, доступ до системи з низьким рівнем привілеїв та ін.);

– black box – імітація дій зловмисників, у яких є тільки назва компанії та практично нульові відомості про систему.

Для проведення тестування на проникнення веб – додатків існують наступні методики:

– OWASP Testing Guide (Інструкція тестування OWASP);
– PTES – Penetration Test Execution Standard (Стандарт виконання тесту на проникнення);
– ISSAF - Information System Security Assessment Framework – PTF (Penetration Testing Framework) – Структура оцінки безпеки інформаційної системи – Структура тестування на проникнення.

Автори методики OWASP Testing Guide наголошують на тому, що необхідно впроваджувати тестування захищеності веб – додатків на кожному з етапів розробки програмного забезпечення [60, с. 12]. Згідно методики OWASP Testing Guide, тестування проводиться у наступній послідовності [60, с. 30]:

– збір інформації;
– тестування конфігурації;
– тестування механізмів керування ідентифікацією;
– тестування процесу аутентифікації;
– тестування процесу авторизації;
– тестування механізмів керування сесіями;
– тестування обробки вхідних даних від користувача;
– обробка помилок;
– тестування механізмів, що реалізують криптографічні функції;
– тестування бізнес – логіки додатка;
– тестування клієнтської частини.

У кожному з етапів докладно описується інформація, яку необхідно зібрати під час виконання тестування, як обробляти отриману інформацію, які компоненти додатка необхідно перевірити, та програмні засоби, за допомогою яких можна провести тестування додатка на кожному з етапів з прикладами їх

використання. В кінці кожного з етапів наведено посилання, які містять додаткову корисну інформацію про особливості проведення тестування.

Стандарт виконання тесту на проникнення PTES описує 7 основних етапів проведення тестування на проникнення [61]:

– попередня взаємодія між сторонами;
– збір інформації;
– моделювання загроз;
– аналіз вразливостей;
– експлуатація вразливостей;
– пост – експлуатаційний період та оцінка можливих збитків від атак;
– формування звітів.

Окремою частиною стандарту PTES є розділ технічних рекомендацій (Technical Guidelines) у якому описане необхідне програмне забезпечення і додаткова інформація для практичної реалізації тестування на проникнення.

Методика ISAAF розроблена групою безпеки відкритих інформаційних систем (OSSIG – Open Information Systems Security Group). Згідно методики ISAAF, тестування на проникнення склається з наступних 3 фаз [62, с. 13]:

– планування та підготовка до тестування (підписання угоди між замовником та виконавцем тестування на проникнення, узгодження методики тестування та набору програм для проведення тестування на проникнення);
– проведення тестування на проникнення (збір інформації, складання схеми мережі, що тестується, ідентифікація вразливостей, заходи з проникнення у систему, отримання доступу до ресурсів, компрометування віддалених користувачів / сайтів, приховування слідів);
– формування звіту про проведене тестування на проникнення (перелік програм та методик, що були використані під час тестування, дата та час проведення тестування, список знайдених вразливостей, рекомендації для підвищення безпеки).

Методика ISAAF дозволяє проводити:

– оцінку захищеності паролів;
– оцінку захищеності мережевих пристроїв;
– оцінку захищеності міжмережевих екранів;
– оцінку захищеності систем виявлення вторгнень;
– оцінку захищеності веб – додатків;
– оцінку захищеності операційних систем;
– аудит програмного коду;
– аналіз захищеності баз даних.

1.9 Висновки до першого розділу

Захищеність веб – додатків від атак зловмисників залежить від технологій та компонентів, які використовуються при побудові веб – додатків, а також від можливих вразливостей у цих компонентах. Існують різні класифікації вразливостей, кожна атака через вразливість має свої особливості, але причина виникнення вразливостей – помилки при проектуванні, реалізації та застосуванні компонентів веб – додатків, отже виникає необхідність пошуку вразливостей та реагування на інформацію про випадки їх знайдення. Як в Україні, так і в інших країнах світу організовуються команди реагування на надзвичайні події у кібербезпеці, що складаються з експертів та дослідників, існує багато програм по отриманню винагороди за знайдені вразливості, такі як Bug Bounty. Статистичну інформацію про знайдені вразливості та їх особливості можна знайти у спеціально створених базах даних вразливостей, окремі міжнародні стандарти регулюють процес розкриття вразливостей.

Широкий вибір засобів дозволяє проводити пошук вразливостей, проте ефективність їх використання залежить від алгоритму дій, за яким необхідно проводити цей пошук. Алгоритми дій представлені у вигляді спеціальних методик, які охоплюють широке коло питань кібербезпеки, таких як тестування захищеності фізичного середовища, бездротових мереж, операційних систем,

мережевого обладнання та ін. Таким чином, необхідні додаткові витрати часу на аналіз існуючих методик та обрання тих складових, які підходять для тестування веб – додатків. Тому виникає необхідність у розробці такої методики тестування на проникнення, яка враховувала б міжнародні досягнення у тестуванні веб – додатків та містила перелік можливих засобів тестування.

РОЗДІЛ 2
РОЗРОБКА МЕТОДИКИ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ

Існують різні вразливості, атаки через вразливості та різний розмір збитків від кожного з типу атак. Список OWASP Top – 10 охоплює найбільш поширені вразливості, що свідчить про високу вірогідність атак зловмисників різного рівня підготовки.

Результат пошуку можливих вразливостей веб – додатків зі списку OWASP Top – 10 дозволить оцінити базовий рівень захищеності веб – додатка від атак зловмисників.

2.1 Формулювання завдання

Для створення ефективної методики, яка б дозволяла тестувати на проникнення веб – додатки за списком вразливостей та загроз, вказаних у рейтингу ризиків OWASP Top – 10 необхідно:

– сформулювати вимоги до методики, що розробляється з метою тестування на проникнення веб – додатка на наявність кожного з 10 ризиків списку OWASP Top – 10;
– вибрати необхідні нтапи з існуючих методик, які дозволяли б провести тестування на проникнення веб – додатка;
– розробити методику тестування на проникнення веб – додатка із визначенням етапів тестування та програмними засобами, які дозволяють перевіряти наявність кожного з 10 ризиків списку OWASP Top – 10.

2.2 Основні вимоги до методики тестування на проникнення веб - додатків

Методика тестування на проникнення веб - додатків за рейтингом вразливостей та ризиків OWASP Top – 10 повина відповідати наступним вимогам:

– враховувати необхідність початкового збору інформації про відкриті порти на сервері, запущені на сервері служби, тип та версію веб – сервера, тип міжмережевого екрану, тип операційної системи;

– методика повинна враховувати необхідність тестування веб – додатка на наявність компонентів, що можуть містити відомі вразливості;

– методика повинна враховувати необхідність тестування засобів перевірки вхідних даних до веб – додатка на наявність вразливостей, які можуть призвести до вставки інструкцій (вставки SQL, XML – інструкцій, міжсайтове виконання сценаріїв);

– методика повинна враховувати перевірку засобів аутентифікації на можливість реалізації загроз некоректної аутентифікації;

– методика повинна враховувати перевірку веб – додатка на можливий витік конфіденційних даних;

– методика повинна дозволяти проводити перевірку правильності контролю доступу;

– методика повинна враховувати необхідність перевірки веб – додатка на наявність вразливостей у компонентах, що відповідають за десеріалізацію об’єктів;

– методика повинна враховувати необхідність перевірки компонентів додатка, що відповідають за моніторинг та ведення журналів;

– після проведення необхідних етапів тестування повинен бути сформований звіт про результати тестування;

2.3 Розробка методики тестування на проникнення веб - додатків за рейтингом ризиків OWASP Top – 10

Враховуючи вимоги, які повинна містити методика, та рекомендації в існуючих методиках, тестування на проникнення доцільно розділити на наступні етапи:

– збір інформації про систему;

– тестування компонентів веб - додатка, що можуть містити відомі вразливості;
– тестування засобів перевірки вхідних даних до веб – додатка;
– тестування засобів автентифікації;
– тестування можливості витоку конфіденційних даних;
– тестування засобів контролю доступу;
– тестування веб – додатка на наявність компонентів, що відповідають за десеріалізацію об’єктів;
– тестування засобів моніторингу та ведення журналів подій;
– підготовка звіту.

2.3.1 Збір інформації про систему

Цей етап призначений для визначення інформації про веб – додаток, версії веб – сервера, відкритих портів, запущених служб. Зібрана інформація може свідчити про об’єм інформації, яка може бути доступна зловмиснику, який може використати її для пошуку записів про існуючі вразливості в базах даних вразливостей та здійснити атаку на веб – додаток.

Для проведення етапу збору інформації було обрано сканер Nmap, що входить до ОС Kali Linux. В залежності від заданих опцій програма формує відповідні вихідні дані. Сканування за допомогою Nmap виконується після задання спеціальних команд, загальний синтаксис яких виглядає наступним чином: `nmap [ <Тип сканування> ...] [ <Опції> ] { <веб – адреса цілі сканування> }`

Nmap використовується для:

– перевірки відкритих портів на сервері;
– сканування запущених на сервері служб;
– визначення типу та версії веб – сервера;
– визначення типу міжмережевого екрану;
– визначення типу операційної системи;

Приклад роботи Nmap вказано на рисунку 2.1:

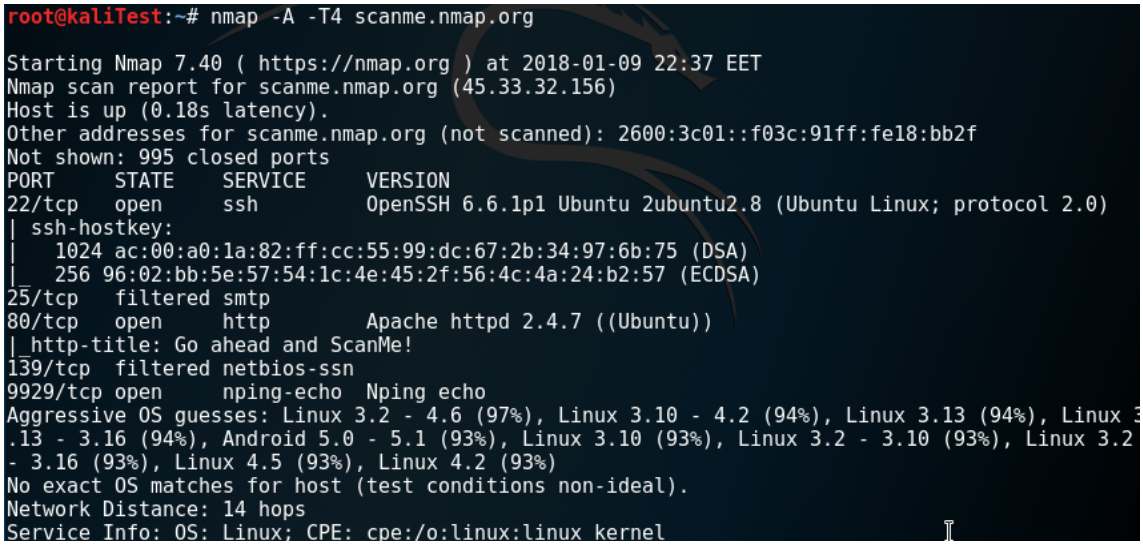

Рисунок 2.1 – Результат роботи сканера Nmap

2.3.2 Тестування компонентів веб - додатка, що можуть містити відомі вразливості

Після збору інформації необхідно перевірити наявність експлойтів для веб – серверів та операційних систем згідно версій. Для цього пропонується використовувати утиліту searchsploit, що входить до ОС Kali Linux. Утиліта використовує базу даних експлойтів Exploit Database [63], яка може бути використана для перевірки можливості атаки за допомогою існуючих вразливостей.

Знайдені експлойти виконуються за допомогою програми Metasploit Framework, яка входить до Kali Linux та дозволяє:

- імітувати мережеві атаки;
- виявляти вразливості системи;
- перевіряти ефективність роботи систем виявлення вторгнень;
- розробляти нові експлойти;
- створювати звіти за результатами роботи.

Metasploit Framework запускається за допомогою команди msfconsole. Основними командами Metasploit Framework є:

– use – вибір певного експлойта для роботи;
– back – команда для припинення роботи з обраним модулем;
– show – вивести список модулів певного типу;
– set – встановити значення певного об’єкту;
– run – запустити допоміжний модуль після того, як були встановлені необхідні опції;
– info – вивести інформацію про модуль;
– search – знайти певний модуль;
– check – перевірка можливості цільової системи до вразливості;
– sessions - вивести список доступних сесій.

Приклад пошуку експлойтів, що стосуються веб – серверів Apache виконується командою: searchsploit apache. Результати роботи утиліти searchsploit вказано на рисунку 2.2:

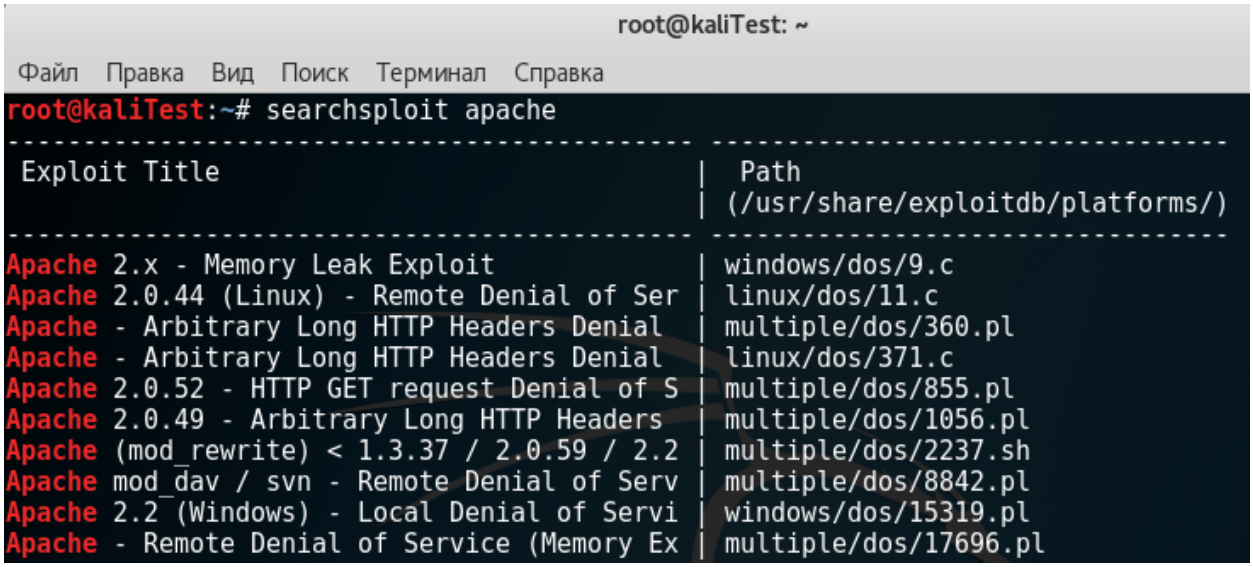

Рисунок 2.2 – Результат роботи утиліти searchsploit

Прикладом роботи програми Metasploit Framework може бути пошук та експлуатація вразливості за допомогою експлойта у системі керування контентом WordPress [64]. Пошук починається з запуску Metasploit Framework командою msfconsole. Пошук модулів, які містять інформацію про вразливість

WordPress, виконується командою search wordpress. Результат пошуку модулів вказано на рисунку 2.3.

Рисунок 2.3 – Результат роботи Metasploit Framework з пошуку модулів, що описують вразливості WordPress

Згідно рисунку 2.3, для атаки на WordPress було обрано модуль auxiliary/scanner/http/wordpress_login_enum, який відповідає за злам процесу автентифікації методом грубої сили.

Обирається для роботи вказаний модуль командою:
use auxiliary/scanner/http/wordpress_login_enum.

Інформацію про особливості використання обраного модулю можна отримати, за допомогою команди:
info use auxiliary/scanner/http/wordpress_login_enum

Наступний етап – вказання адреси, на якій розташований веб – сайт. Для цього використовується команда set RHOSTS <адреса_веб - сайту>. Також необхідно вказати шлях до файлу з паролями, перебір яких буде виконуватися: set PASS_FILE /root/10k-common-passwords.txt. На рисунку 2.4 вказано процес перебору паролів за словником за допомогою Metasploit Framework

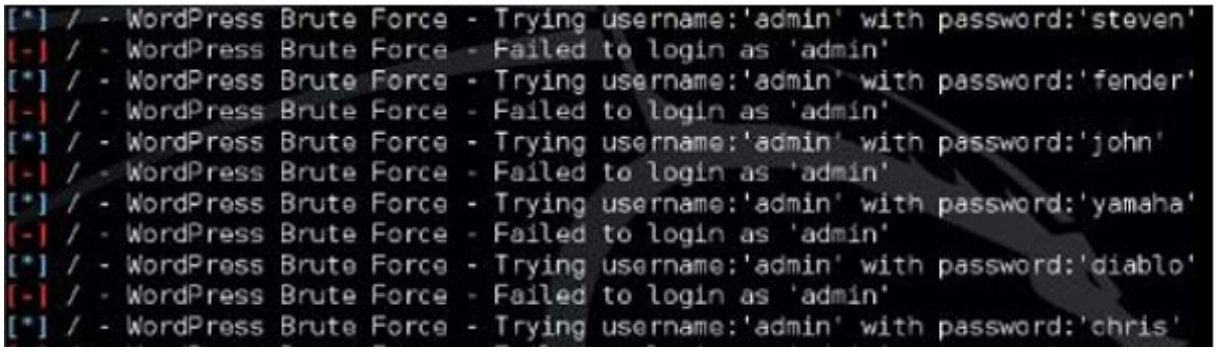

Рисунок 2.4 – Процес перебору паролів за словником за допомогою Metasploit Framework

Після перебору паролів зі словника при знаходженні співпадіння відображається інформація про успішну автентифікацію, вказану на рисунку 2.5:

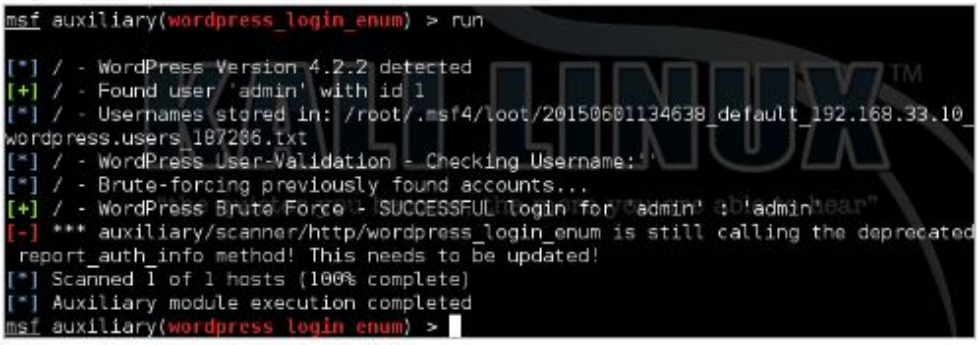

Рисунок 2.5 – Інформація про знайдений пароль за методом перебору за словником

2.3.3 Тестування засобів перевірки вхідних даних до веб – додатка

Тестування засобів перевірки вхідних даних до веб – додатка включає наступні дії:

- перевірка параметрів GET, POST у HTTP – запитах;
- перевірка можливості переповнення буфера памяті, та чи перешкоджають існуючі засоби веб – додатка його переповненню;
- перевірка того, чи використовується в веб - додатку єдиний елемент керування перевіркою введення для кожного прийнятого типу даних.

– тестування можливості реєстрації помилки перевірки вхідних даних на стороні сервера
– тестування можливості вставки SQL, XML – інструкцій, міжсайтового виконання сценаріїв;

Перш за все, необхідно визначити вхідні точки веб – додатка, які можуть бути відправними точками для потенційних атак. Точками входу даних можуть бути частини веб – додатка, які виконують обробку HTTP – запитів (методи GET та POST), порти, сокети. Також необхідно визначити всі параметри, що передаються у HTTP – запитів, з метою визначення можливих місць, де може маніпулювати даними зловмисник. Для проведення тестування на проникнення за цим етапом необхідно:

– визначити, де використовуються запити GET і де використовуються POST;
– визначити всі параметри, що використовуються в запиті POST;
– визначити всі параметри, що використовуються в запиті GET;
– визначте всі параметри рядка запиту, багато параметрів можуть бути в одному рядку, розділені символами &, ~,: або будь-яким іншим спеціальним символом або кодуванням.

Прикладом GET та POST – запитів можуть бути:

GET
https://x.x.x.x/shoppingApp/buyme.asp?CUSTOMERID=100
&ITEM=z101a&PRICE=62.50&IP=x.x.x.x
POST https://x.x.x.x/App/authenticate.asp?service=login

Аналіз GET та POST – запитів можна виконати модулем Burp Proxy, що є елементом програми Burp Suite. Burp Proxy є проксі – елементом, який дозволяє вам перехоплювати та перевіряти трафік між браузером та веб – додатком [65]. На рисунку 2.6 показано приклад роботи Burp Proxy, що прехоплює GET – запити.

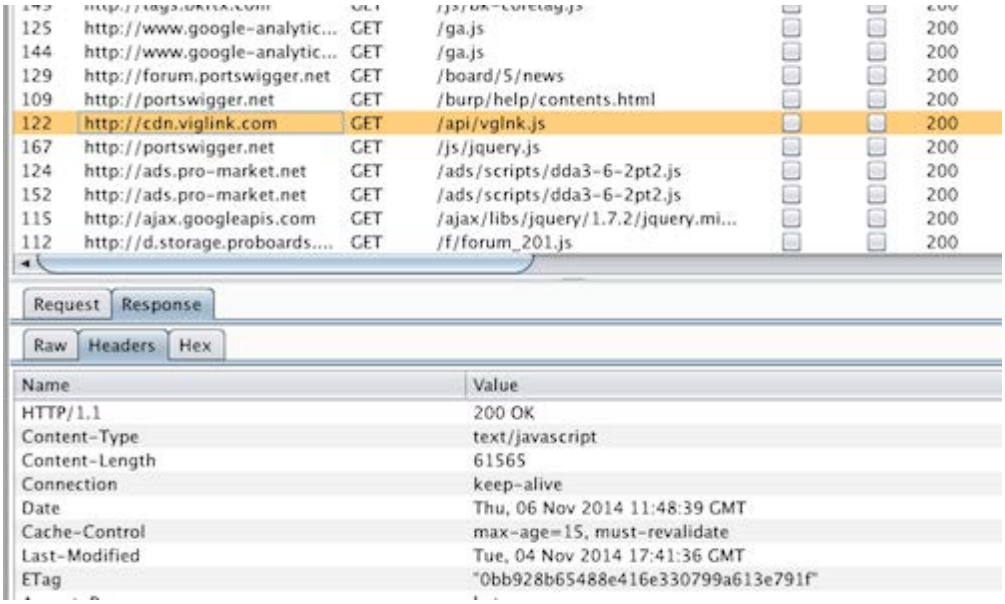

Рисунок 2.6 – Приклад роботи Burp Proxy з аналізу HTTP - запитів

Перевірку на можливість вставки SQL – інструкцій, слід починати з визначення типу URL. Існує багато разноманітних шаблонів URL, наприклад:

nurl:item_id=

nurl:readnews.php?id=

inurl:news.php?id=

inurl:shopping.php?id=

URL веб – додатка може мати наступний вигляд:

http://www.example.com/page.php?id=28

Необхідно у кінець URL додати символ ', строка мала наступний вигляд:

http://www.example.com/rubrika.php?id=28'

Після переходу за посиланням може відкритися вікно з повідомленням, що сторінка не існує та інформацією про помилку, яка може мати наступний вигляд:

java.sql.SQLException: ORA-00933: SQL command not properly ended at oracle.jdbc.dbaaccess.DBError.throwSqlException(DBError.java:180) at oracle.jdbc.ttc7.TTIoer.processError(TTIoer.java:208)

Отримані дані можуть свідчити про наявність вразливостей при обробці вводу інформації користувачем. Подальші дії необхыдно виконувати за допомогою утиліти sqlmap. Можливості sqlmap [66]:

– підтримка баз даних MySQL, Oracle, PostgreSQL, Microsoft Access, IBM DB2, SQLite, Firebird;
– підтримка шести методів SQL – інструкцій;
– підтримка підключення до баз даних;
– підтримка переліку користувачів, хеш-паролів, привілеїв, ролей, баз даних, таблиць і стовпців.

Визначення існуючих баз даних, які використовуються веб – додатком проводиться за допомогою команд -u та --dbs:

sqlmap -u http://www.example.com/page.php?id=28 --dbs

Приклад результату роботи sqlmap представлено на рисунку 2.7.

```
[*] starting at 12:12:56

[12:12:56] [INFO] resuming back-end DBMS 'mysql'
[12:12:57] [INFO] testing connection to the target url
sqlmap identified the following injection points with a total of 0 HTTP(s) requests:
---
Place: GET
Parameter: id
    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE or HAVING clause
    Payload: id=51 AND (SELECT 1489 FROM(SELECT COUNT(*),CONCAT(0x3a73776c3a,(SELECT (CASE WHEN (14
---
[12:13:00] [INFO] the back-end DBMS is MySQL
web server operating system: FreeBSD
web application technology: Apache 2.2.22
back-end DBMS: MySQL 5
[12:13:00] [INFO] fetching database names
[12:13:00] [INFO] the SQL query used returns 2 entries
[12:13:00] [INFO] resumed: information_schema
[12:13:00] [INFO] resumed: safecosmetics
available databases [2]:
[*] information_schema
[*] safecosmetics
```

Рисунок 2.7 – Результату роботи sqlmp

Згідно рисунку 2.7 видно, що використовується СКБД MySQL 5, операційна система веб – сервера FreeBSD, Apache 2.2.22 та таблиці information_schema та safecosmetics. Подальші перевірки можуть зводитися до перевірки можливості доступу до записів у таблицях та пошуку експлойтів для версій знайденого програмного забезпечення. Інші команди доступні за

посиланням розробника sqlmap [66]. Пошук експлойтів можна здійснювати за допомогою Metasploit Framework.

Перевірка можливості міжсайтового виконання сценаріїв полягає у відправленні до веб – додатка даних, які повинні або проходити перевірку, або призводити до порушення логіки роботи веб – додатка. Прикладом таких даних може бути заповнення полів у формі реєстрації, яка заповнюється користувачем та відправлення їх на сервер. Дані для заповнення можуть виглядати наступним чином: *<script> alert(1) </script>*. За наявності вразливостей, що призводять до міжсайтового виконання сценаріїв на стороні користувача відобразиться вікно, зображене на рисунку 2.8.

Рисунок 2.8 – Результат відповіді від сервера при наявності вразливостей міжсайтового виконання сценаріїв

Тестування на можливість міжсайтового виконання сценаріїв проводиться за допомогою Burp Suite. Як видно на рисунку 2.9 з веб – сайту розробника Burp Suite [67], програма дозволяє задавати необхідні параметри для запиту та відправляти запит на сервер.

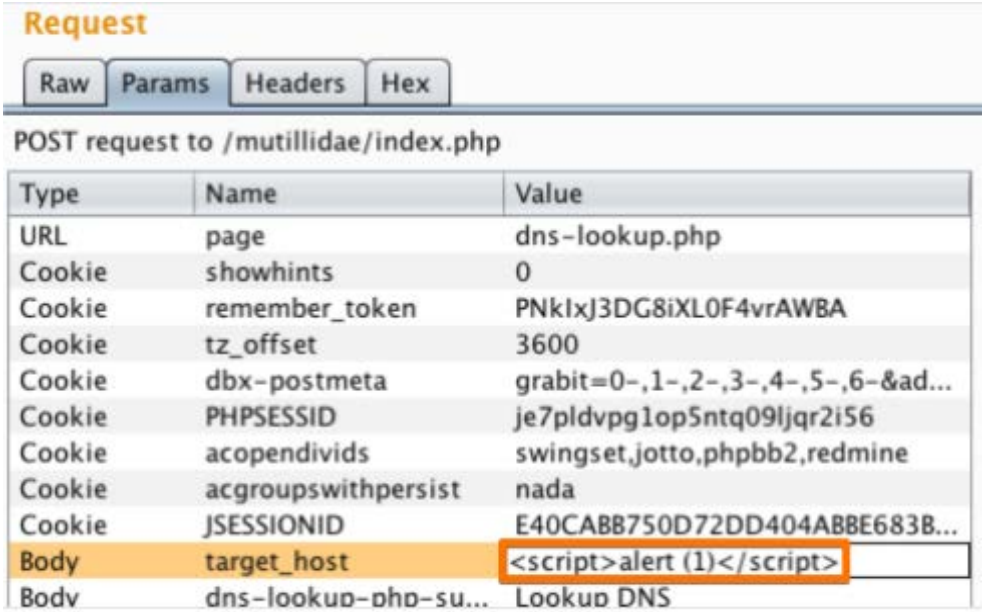

Рисунок 2.9 – Вікно відображення параметрів, що передаються до веб – додатка програмою Burp Suite

2.3.4 Тестування засобів автентифікації

Під час тестування веб – додатка на даному етапі враховуються дані, отримані під час виконання попереднього етапу – тестування засобів перевірки вхідних даних до веб – додатка. Додаткові перевірки здійснюються наступним чином:

- перевірка використання HTTPS – протоколу при роботі з GET та POST – запитами для визначення можливості перехоплення даних;
- перевірка наявності та роботи механізмів захисту при вході від методів перебору паролів (визначення порогового значення невдалих спроб набору паролю);
- перевірка механізмів розблокування облікового запису адміністратора;
- визначення періоду блокування доступу через неправильне введення паролю;
- перевірка того, чи потребується обов’язкова автентифікація перед доступом до всіх сторінок та ресурсів за винятком тих, для яких попередня автентифікація не потребується;

- перевірка можливості підміни параметрів у запитах;
- перевірка реалізації механізмів призначення ідентифікаторів сесії;
- перевірка того, які символи дозволяються та забороняються для паролів (чи потрібно користувачеві використовувати малі та великі літери, цифри та спеціальні символи);
- перевірка того як часто користувач може змінити свій пароль;
- перевірка того чи може користувач може повторно використовувати пароль;
- перевірка наявності в веб – додатку функції двохфакторної автентифікації, яка забезпечує захист від розпізнавання логіну та паролю;
- перевірка того, що функція зміни пароля включає старий пароль, новий пароль та підтвердження пароля;
- перевірка того, що всі спроби автентифікації реєструються системою.

Необхідні засоби для реалізації тестування на проникнення на даному етапі: nmap, Burp Suite, Metasploit Framework.

2.3.5 Тестування можливості витоку конфіденційних даних

Конфіденційні дані можуть містити дані про конкретні програми, такі як номери кредитних карток, реквізити працівників, фінансові записи, облікові дані облікового, бази даних та записи у них. Отже, необхідно провести тестування засобів шифрування конфіденційної інформації у файлах та базах даних.

Перш за все необхідно визначити порти, що містять SSL / TLS – служби. Це можна зробити за допомогою сканера nmap та команди:

nmap -sV --reason -PN -n --top-ports 100 www.example.com, де

-sV – перевірка сервісів на портах

--reason – спосіб підключення до порту

-PN - перевіряти всі хости, які доступні

--top-ports 100 – команда сканування портів та їх кількість

www.example.com – веб – адреса, що сканується

Результат роботи програми nmap за вказаною командою може бути таким, як вказано на рисунку 2.10 [68].

```
$ nmap -sV --reason -PN -n --top-ports 100 www.example.com
Starting Nmap 6.25 ( http://nmap.org ) at 2013-01-01 00:00 CEST
Nmap scan report for www.example.com (127.0.0.1)
Host is up, received user-set (0.20s latency).
Not shown: 89 filtered ports
Reason: 89 no-responses
PORT    STATE SERVICE  REASON  VERSION
21/tcp  open  ftp      syn-ack Pure-FTPd
22/tcp  open  ssh      syn-ack OpenSSH 5.3 (protocol 2.0)
25/tcp  open  smtp     syn-ack Exim smtpd 4.80
26/tcp  open  smtp     syn-ack Exim smtpd 4.80
80/tcp  open  http     syn-ack
110/tcp open  pop3     syn-ack Dovecot pop3d
143/tcp open  imap     syn-ack Dovecot imapd
443/tcp open  ssl/http syn-ack Apache
465/tcp open  ssl/smtp syn-ack Exim smtpd 4.80
993/tcp open  ssl/imap syn-ack Dovecot imapd
995/tcp open  ssl/pop3 syn-ack Dovecot pop3d
Service Info: Hosts: example.com
Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 131.38 seconds
```

Рисунок 2.10 – Приклад сканування портів

З рисунку 2.10 видно, що з просканованих 100 портів 89 фільтруються, а решта відкриті і доступна інформація про служби, що працюють на цих портах. Перевірити інформацію про стійкість алгоритмів шифрування, що містяться у SSL/HTTP – службі на порту 443 можна за допомогою команди

nmap -sV --script ssl-enum-ciphers -p 443 <host> , де

-sV

--script

ssl-enum-ciphers

-p 443

<host>

Результат роботи програми nmap за вказаною командою має вигляд, вказаний на рисунку 2.11 [69].

```
PORT    STATE SERVICE REASON
443/tcp open  https   syn-ack
| ssl-enum-ciphers:
|   TLSv1.0:
|     ciphers:
|       TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp256r1) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 2048) - A
|       TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA (secp256r1) - C
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp256r1) - C
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 2048) - C
|       TLS_ECDHE_ECDSA_WITH_RC4_128_SHA (secp256r1) - C
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp256r1) - C
|       TLS_RSA_WITH_RC4_128_SHA (rsa 2048) - C
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 2048) - C
|     compressors:
|       NULL
|     cipher preference: server
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
```

Рисунок 2.11 – Фрагмент результату сканування алгоритмів, що підтримуються SSL/HTTP – службою на порту 443

Подальше тестування на етапі тестування можливості витоку конфіденційних даних полягає в пошуку експлойтів до запущених служб та інформації про веб - сервери та операційні ситсеми за допомогою Metasploit Framework.

2.3.6 Тестування засобів контролю доступу

Контроль доступу полягає у надання доступу до ресурсів лише тим, хто має право на їх використання. Враховуючи це, необхідно проводити тестування, яке передбачає:

- тестування можливості ескалації привілеїв;
- тестування наявності незахищених посилань на об’єкти.

Тестування можливості ескалації привілеїв необхідне для визначення того, чи може зловмисник змінювати у веб – додатку свої привілеї при доступі та мати доступ до ресурсів, який не передбачений для більшості користувачів. Ескалація привілеїв може бути у наданні більш високих привілеїв до ресурсів (повноваження адміністратора), або доступ до ресурсів іншого користувача. Прикладом тестування може слугувати наступний POST – запит:

POST /user/viewOrder.jsp HTTP/1.1

Host: www.example.com

groupID=grp001&orderID=0001

У запиті використовується groupID та orderID. Необхідно перевірити, чи зможе користувач з іншим значенням groupID та orderID отримати доступ.

Незахищені посилання на об’єкти виникають тоді, коли веб – додаток забезпечує прямий доступ до об'єктів (баз даних або файлів). Модифікуючи параметри, зловмисник може обійти процес авторизації та отримати доступ до інформації. Для тестування необхідно визначити параметри, які можна модифікувати та отримати доступ до незахищених обєктів. Прикладом таких параметрів може бути несупне посилання:

http://example.com/somepage?value=12345, де параметр value має значення 12345 в якості індексу запису у базі даних. Для перевірки можливості доступу до іншого запису необхідно замінити значення value іншим значенням. В результаті запис буде або відсутній, або буде виведено інформацію про новий запис.

2.3.7 Тестування веб – додатка на наявність компонентів, що відповідають за десеріалізацію об’єктів

Серіалізація використовується для перетворення деякого об'єкта у формат даних, який можна зберігати та відновити пізніше. Десеріалізація виконується для обробки даних у певному форматі та перебудови їх у об'єкт. Веб – додатки можуть бути вразливими до атак на основі наявності реалізації

незахищеної десеріалізації, якщо вони десеріалізують об’єкти, що можуть надавати користувачі та зловмисники. Зловмисник може змінити логіку роботи програми, десеріалізуючи у додатку об’єкт, що містить шкідливий код.

Наступні компоненти веб – додатка необхідно тестувати на наявність вразливостей механізмів десеріалізації:

– протоколи;
– веб-сервіси;
– системи керування базами даних;
– файлові системи;
– параметри HTML-форми;
– токени автентифікації.

Програмні засоби, які можна використовувати для тестування компонентів веб – додатків: Nmap для збору інформації, Metasploit Framework для пошуку експлойтів для вразливих модулів.

2.3.8 Тестування засобів моніторингу та ведення журналів подій

Тестування механізмів моніторингу та ведення журналів подій необхідно проводити за наступними критеріями:

перевірка веб – додатка на можливість видачі повідомлення користувачу про помилки та перевід;

Засобами веб – додатка повинні реєструватися:

– дата та час створення кожного запису;
– дата та час події;
– ідентифікатори програм, їх назва та версія;
– веб – адреса додатка, номер порту;
– служби, які запущені;
– невдалі перевірки вхідних даних (протокольні порушення, неприпустимі кодування, невірні параметри імен та значень);
– події виправлення помилок перевірки вихідних даних;

- випадки вдалої автентифікації;
- випадки невдалої автентифікації (контроль доступу);
- випадки модифікації конфігурації;
- випадки модифікації файлів програмного коду;
- ідентифікатор пристрою/машини користувача, IP-адреса користувача;
- випадки використання функцій підвищеного ризику (мережеві з'єднання, додавання чи видалення користувачів, зміна привілеїв, використання системних адміністративних привілеїв, доступ адміністраторів до веб – додатків, всі дії користувачів з адміністративними правами, доступ до даних власників платіжних карток, використання ключа шифрування даних, зміни ключа, створення та видалення об'єктів системного рівня, імпорту та експорту даних, завантаження файлів);
- детальна інформація про повідомлення щодо системних помилок, інформацію про процес налагодження, заголовки HTTP – запитів та їх основна частина;
- до журналів подій організовано контроль доступу.

Для створення загальної картини реалізації моніторингу та ведення журналів подій, можна записувати у спеціальну відомість у вигляді таблиці інформацію, чи реєструються описані вище дані або ні. Приклад таблицы наведено нижче.

Таблиця 2.1

Зразок відомості про події та атрибути, що підлягають запису у журнал подій

Назва події/атрибуту	Чи реєструються у журналах події (так/ні)	Примітки
…	…	…

Повідомлення про помилки, які відображаються користувачу, не повинні містити дані, які можуть допомогти зловмисникові (ідентифікатор сеансу, назви та версії програм, які використовуються). Перевірку за цим критерієм можна реалізувати за допомогою команди *telnet,* яка доступна у Kali

Linux. Синтаксис команди повинен бути наступним: *telnet <веб_адреса> 80 GET <сторінка_що_не_існує> HTTP/1.1*. Відповідь сервера про сторінку, що не існує, може містити додаткову інформацію, наприклад:

HTTP/1.1 404 Not Found

Date: Sat, 04 Nov 2017 15:26:48 GMT

Server: Apache/2.2.3 (Unix) mod_ssl/2.2.3 OpenSSL/0.9.7g

Content-Length: 310

Connection: close

Content-Type: text/html; charset=iso-8859-1

Вказана інформація може бути корисною для зловмисників та не повинна відображатися для користувачів.

2.3.9 Підготовка звіту

Останнім етапом є формування звіту про результати проведеного тестування на проникнення. Інформація у звіті необхідна для аналізу поточного стану захищеності веб – додатка та можливості атаки його зловмисником та складаня подальшого плану підвищення його захищеності. Звіт повиненн містити:

- назву веб – додатка, тестування якого проводиться;
- методику, за якою проводилося тестування;
- терміни початок та закінчення кожного етапу тестування;
- перелік програмних засобів, які використовувались при тестуванн на проникнення;
- перелік зібраної інформації про веб додаток, знайденої під час тестування;
- перелік знайдених вразливостей та рівнів їх загроз;
- рекомендації для запобігання наявним вразливостям;

2.4 Висновки до другого розділу

Запропонована у дипломній роботі методика враховує міжнародні досягнення у тестуванні веб – додатків та дозволяє перевіряти наявність вразливостей зі списку OWASP Top – 10. Від існуючих методик ця методика відрізняється тим, що вона охоплює тількі ті питання, що стосуються пошуку вразливостей веб – додатків зі списку OWASP Top – 10, тим самим прискорюючи аналіз кількості необхідних перевірок для тестування. Порівняльний аналіз структури запропонованої та існуючих методик надедено у додатку Б.

Структуру методик складають етапи, на кожному з яких визначено необхідні перевірки. Приклади у етапах тестування на проникнення за запропонованою методикою наведено з метою відображення можливих результатів, які можуть свідчити про наявність вразливості, а також з метою показати роботу з засобами виявлення вразливостей та тестування на проникнення.

У запропонованій методиці використовується обмежена кількість етапів тестування та засобів тестування, які дозволяють ефективно проводити тестування на проникнення за списком вразливостей OWASP Top – 10.

РОЗДІЛ 3
ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ДОЦІЛЬНОСТІ ВИКОРИСТАННЯ РОЗРОБЛЕНОЇ МЕТОДИКИ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ ВЕБ – ДОДАТКІВ

3.1 Розрахунок економічної ефективності використання розробленої методики

Передбачається, що використання запропонованої методики дозволить зменшити витрати на проведення тестування на проникнення, при цьому методика дозволяє визнчити головні загрози та вразливості веб – додатків.

Для оцінки економічної ефективності використання запропонованої методики необхідно використовувати значення вартості та термінів поведення тестування на проникнення за існуючими методиками. Орієнтовна вартість проведення тестування на проникнення складає 85050 грн. [70].

Запропонована методика дозволяє проводити тестування на проникнення спеціалістам з базовим рівнем підготовки у галузі захисту інфомації.

Для розрахунку витрат на заробітну плату спеціаліста, що виконує тестування на проникнення, нобхідно визначити:

- час на ознайомлення з технічним завданням та методикою, $t_{тз}$;
- час на підготовку середовища для виконання тестування на проникнення (встановлення необхідних програм та їх налаштування), $t_{п}$;
- час на виконання етапів методики, $t_{м}$;

Перш за все необхідно обчислити умовну кількість етапів у методиці з урахуванням можливих уточнень у процесі роботи з методикою [71]:

$$Q = q \cdot c\ (1 + p), \text{ штук}, \quad (3.1)$$

де q – очікувана кількість етапів методики;

c – коефіцієнт складності методики, с = 1,25...2,0;

p – коефіцієнт корекції методики в процесі її опрацювання (p = 0,05...0,1, що відповідає внесенню 3...5 корекцій і переробці 5-10% методики в залежності від архітектури веб – додатка).

Кількість етапів тестування на проникнення згідно методик:

– OWASP Testing guide – 11;
– методика, що запропонована у дипломній роботі – 8;

Оскільки розрахунок економічної ефективності проводиться з врахуванням можливості проведення тестування на проникнення спеціалістом з базовим рівнем підготовки у галузі захисту інфомації та можливих корекцій або переробки методики в залежності від архітектури веб – додатка, необхідні коефіцієнти мають наступне значення:

– коефіцієнт складності методики, c = 2,0;
– коефіцієнт корекції методики в процесі її опрацювання p = 0,1;

Умовна кількість етапів для методик має наступні значення:

– для методики OWASP Testing guide:

$$Q = 11 \cdot 2{,}0\ (1 + 0{,}1) = 24{,}2\ шт.$$

– для методики, що запропонована у дипломній роботі:

$$Q = 8 \cdot 2{,}0\ (1 + 0{,}1) = 17{,}6\ шт.$$

Час на опрацювання технічного завдання, додаткової літератури та ознайомлення з методикою можливо оцінити за формулою [71]:

$$t_в = \frac{Q \cdot B}{(7 \ldots 8) \cdot k}, \text{ годин,} \qquad (3.2)$$

де В – коефіцієнт збільшення тривалості етапу внаслідок недостатнього опису завдання, В = 1,2...1,5;

k – коефіцієнт, що враховує кваліфікацію програміста і визначається стажем роботи за фахом:

– до 2 років – 0,8;
– від 2 до 3 років – 1,0;
– від 3 до 5 років – 1,1...1,2;
– від 5 до 7 років – 1,3...1,4;
– понад 7 років – 1,5...1,6.

Таким чином, час на опрацювання технічного завдання, додаткової літератури та ознайомлення з методикою має наступні значення:

– для методики OWASP Testing guide:

$$t_{в} = \frac{24{,}2 \cdot 1{,}5}{85 \cdot 0{,}8} = 0{,}53 \text{ години}$$

– для методики, що запропонована у дипломній роботі:

$$t_{в} = \frac{17{,}6 \cdot 1{,}5}{85 \cdot 0{,}8} = 0{,}38 \text{ години}$$

Час на підготовку середовища для виконання тестування на проникнення обчислюється за фоомулою [71]:

$$ta = \frac{Q}{(2\ .02.\ Б k} \text{, годин} \qquad (3.3)$$

Враховуючи це, обчислюємо час на підготовку середовища для виконання тесту на проникнення:

– для методики OWASP Testing guide:

$$t_{а} = \frac{36{,}3}{25 \cdot 0{,}8} = 1{,}8 \text{ години}$$

– для методики, що запропонована у дипломній роботі:

$$t_{а} = \frac{26{,}4}{25 \cdot 0{,}8} = 1{,}3 \text{ години}$$

За наявною інформацією, тривалість тестування на проникнення становить 2 – 3 тижні [72]. Враховуючи кількість етапів тестування на проникнення за методикою OWASP Testing guide, та їх особливості, приймемо тривалість тестування за цією методикою – 2 тижні. 2 робочі тижні при 5 – денному робочому тижні становлять 10 днів або 80 годин при 8 – годинному робочому дні [73]. Приймаючи до уваги особливості методики, що запропонована у дипломній роботі, орієнтовна тривалість тестування на проникнення сладає 40 годин. Отже, тривалість тестування за методиками має значення:

– для методики OWASP Testing guide:

$$t_{м} = 80 \text{ годин}$$

– для методики, що запропонована у дипломній роботі:

$$t_м = 40 \text{ годин}$$

Загальний час, що має бути витрачений на виконання тестування на проникнення за запропонованою методикою має значення:

– для методики OWASP Testing guide:

$$t_з = t_в + t_а + t_м = 0{,}53 + 1{,}8 + 80 = 82{,}3 \text{ години}$$

– для методики, що запропонована у дипломній роботі:

$$t_з = t_в + t_а + t_м = 0{,}38 + 1{,}3 + 40 = 41{,}6 \text{ години}$$

Зменшення часу на проведення тестування за запропонованою методикою складає: 82,3 – 41,6 = 40,7 годин.

Розмір заробітної плати спеціаліста з захисту інформації в Україні складає 15 тис. грн. на місяць [74]. Враховуючи те, що середня кількість робочих годин за рік складає 166 годин [75], одна година робочого часу спеціаліста з захисту інформації складає:

$$З_{зп} = \frac{15000}{166} = 90{,}4 \text{ грн/год},$$

де $З_{зп}$ – заробітна плата спеціаліста з захисту інформації за одну годину

Таким чином, витрати на заробітну плату спеціаліста з захисту інформації за тестування конкретного веб – додатку, обчислюються за формулою:

$$K_з = З_{зп} \cdot t_з,$$

де $K_з$ – витрати на заробітну плату спеціаліста з захисту інформації, грн.

Витрати на заробітну плату спеціаліста з захисту інформації складають::

– для методики OWASP Testing guide:

$$K_з = З_{зп} \cdot t_з = 90{,}4 \cdot 82{,}3 = 7439{,}9 \text{ грн}$$

– для методики, що запропонована у дипломній роботі:

$$K_з = З_{зп} \cdot t_з = 90{,}4 \cdot 41{,}68 = 3767{,}8 \text{ грн}$$

Таким чином, використання запропонованої методики тестування на проникнення дозволяє проводити тестування на наявність загроз та вразливостей веб – додатків, та зменшує час проведення тестування, який витрачає спеціаліст з захисту інформації, отже заощаджений час може бути

використаний на проведення тестування іншого веб – додатка. Результати розрахунків вказано у таблиці 3.1.

Таблиця 3.1

Порівнювальна таблиця, що характеризує витрати на проведення тестування на проникнення за різними методиками

	Методики для розрахунку		Різниця
Критерії порівняння методик	OWASP Testing guide	Методика, що запропонована у дипломній роботі	
Кількість етапів тестування на проникнення у методиці, шт.	11	8	3
Час, витрачений на виконення тестування за методикою, годин	82,3	41,6	40,7
Витрати на заробітну плату спеціаліста з захисту інформації, грн.	7439,9	3767,8	3672,1

Таким чином, як видно з таблиці 3.1, тестування на проникнення веб – додатка за запропонованою у дипломній роботі методикою дозволяє зменшити витрати часу на тестування на 40,7 годин, при цьому протестувати веб – додаток на наявність вразливостей та ризиків зі списку OWASP Top – 10. Враховуючи зменшений час на тестування, зменшуються витрати на оплату роботи спеціаліста з захисту інформації за тестування одного конкретного веб – додатка.

3.2 Висновки до третього розділу

Було проаналізовано тривалість проведення тестування на проникнення за існуючими методиками, такими як OWASP Testing guide. Враховуючи ці дані було визначено орієнтовна тривалість тестування на проникнення за запропонованою у дипломній роботі методикою. Економія часу на тестування на проникнення складає 40,7 годин. Також було проведено витрати на оплату роботи спеціаліста з захисту інформації за тестування одного конкретного веб – додатка. Згідно розрахунків, економія витрат на заробітну плату складає 3672,1 грн. за запропонованою у дипломній роботі методикою.

Дані з розрахунків підтверджують економічну доцільність використання запропонованої у дипломній роботі методики для тестування на проникнення веб – додатків.

ВИСНОВКИ

У дипломній роботі було досліджено можливість підвищення ефективності тестування на проникнення. Для досягнення поставленої мети був виконаний аналіз сучасного стану захищеності веб – додатків. Оскільки пошук вразливостей є необхідним етапом тестування на проникнення, було визначено поширені вразливості веб – додатків, особливості атак через існуючі вразливості. Також було проаналізовано структури, які займаються пошуком вразливостей. Згідно досліджень, існують міжнародні стандарти обробки інформації про вразливості, бази даних вразливостей, та засоби їх пошуку. Окремі організації займаються розробкою методик тестування на проникнення. Проте, більшість методик охоплюють широке коло питань кібербезпеки, отже виникає необхідність додаткових витрат часу на аналіз вразливостей згідно існуючих методик та обрання тих складових, зокрема, які підходять для тестування веб – додатків.

За результатами досліджень було розроблено, адаптовано методику тестування веб – додатків на проникнення, яка враховує міжнародні досягнення у цьому напрямку, яка дозволяє перевіряти наявність найбільш поширених вразливостей.

З метою підтвердження економічної доцільності використання запропонованої у дипломній роботі методики, було розраховано вартість проведення тестування на проникнення за цією методикою та за методикою OWASP Testing guide. Враховуючи результати розрахунків, економія витрат на заробітну плату складає 3672,1 грн. за запропонованою у дипломній роботі методикою, що підтверджує економічну доцільність використання методики, що запропонована у дипломній роботі, для тестування на проникнення веб – додатків.

Практичне значення роботи полягає у зниженні тривалості та обгрунтуванні вибору засобів тестування на проникнення. Результати здійснених у дипломній роботі досліджень можуть бути використані при

тестуванні на проникнення веб – додатків та проведенні лабораторних робіт спеціалізованих курсів спеціальності «Кібербезпека».

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1 Web application Wikipedia [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Web_application

2 Angela Stringfellow «What is Web Application Architecture» [Електронний ресурс]. – Режим доступу: https://stackify.com/web-application-architecture/

3 Стаття «Призначення та функції веб - сервера» [Електронний ресурс]. – Режим доступу: http://zametkinapolyah.ru/servera-i-protokoly/http-server-ili-veb-server-naznachenie-funkcii-i-rol-servera-v-http.html

4 Timothy Abel «Different types web application» [Електронний ресурс]. – Режим доступу: https://www.linkedin.com/pulse/6-different-types-web-application-development-timmothy-abel

5 Міжнародний стандарт ISO/IEC 29147 [Електронний ресурс]. – Режим доступу:http://standards.iso.org/ittf/PubliclyAvailableStandards/c045170_ISO_IEC_29147_2014.zip#en

6 Міжнародний стандарт ISO/IEC 27000 [Електронний ресурс]. – Режим доступу:http://standards.iso.org/ittf/PubliclyAvailableStandards/c066435_ISO_IEC_27000_2016(E).zip

7 Термінологія в галузі захисту інформації в комп'ютерних системах від несанкціонованого доступу НД ТЗІ 1.1-003-99 [Електронний ресурс]. – Режим доступу:http://iszzi.kpi.ua/images/Info_bezpeka/ND_TZI/4_НД_ТЗІ_1.1-003-99.pdf

8 The Web Application Security Consortium [Електронний ресурс]. – Режим доступу: http://www.webappsec.org

9 Web Application Security Consortium – Класифікація загроз [Електронний ресурс]. – Режим доступу: http://projects.webappsec.org/f/WASC-TC-v2_0.pdf

10 Офіційний сайт проекту Common Weakness Enumeration [Електронний ресурс]. – Режим доступу: https://cwe.mitre.org/about/index.html

11 Документ Common Weakness Enumeration [Електронний ресурс]. – Режим доступу: https://cwe.mitre.org/data/published/cwe_v3.0.pdf

12 Офіційний сайт проекту Open Web Application Security Project [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/ About_The_Open_Web_Application_Security_Project

13 Рейтинг вразливостей OWASP Top – 10 – 2017 Електронний ресурс]. – Режим доступу: https://www.owasp.org/images/7/72/OWASP_Top_10- 2017_%28 en%29.pdf.pdf

14 Veracode manual penetration testing [Електронний ресурс]. – Режим доступу: https://www.veracode.com/services/penetration-testing

15 Статья «Внедрение SQL кода» [Електронний ресурс]. – Режим доступу: https://ru.wikipedia.org/wiki/Внедрение_SQL-кода

16 Дэйв Уайтлегг, статья «Проверьте ваши приложения на уязвимости из списка OWASP Top 10 за 2013 год» [Електронний ресурс]. – Режим доступу: https://www.ibm.com/developerworks/ru/library/se-owasp-top10/index.html

17 Broken Authentication [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/Top_10-2017_A2-Broken_Authentication

18 Sensitive data exposure [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/Top_10-2017_A3-Sensitive_Data_Exposure

19 External Entity Attack [Електронний ресурс]. – Режим доступу: https://www.owasp.org/images/5/5d/XML_Exteral_Entity_Attack.pdf

20 Ian Muscat - What is XML External Entity [Електронний ресурс]. – Режим доступу:https://www.acunetix.com/blog/articles/xml-external-entity-xxe-vulnerabilities/

21 https://www.owasp.org/index.php/Top_10-2017_A5-Broken_Access_Control

22 https://www.owasp.org/index.php/Category:Access_Control

23 https://www.owasp.org/index.php/Top_10-2017_A6-Security_Misconfiguration

24 Cross – site scripting attack [Електронний ресурс]. – Режим доступу: https://www.acunetix.com/websitesecurity/cross-site-scripting/

25 Блог компанії Acunetix, стаття «What is insecure deserialization ?» [Електронний ресурс]. – Режим доступу: https://www.acunetix.com/blog/articles/what-is-insecure-deserialization/

26 Стаття Insecure_Deserialization Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/Top_10-2017_A8-Insecure_Deserialization

27 Стаття Insufficient_Logging and Monitoring [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/Top_10-2017_A10Insufficient_Logging%26Monitoring

28 Офіційний сайт організації CERT [Електронний ресурс]. – Режим доступу: https://www.cert.org/about/

29 Офіційний сайт організації TF-CSIRT [Електронний ресурс]. – Режим доступу: https://tf-csirt.org/tf-csirt/

30 Офіційний сайт Державної служби спеціального зв’язку та захисту інформації: [Електронний ресурс]. – Режим доступу: http://www.dsszzi.gov.ua /dsszzi/control/uk/publish/article?showHidden=1&art_id=101338&cat_id=87062&ctime=1342168963970

31 Офіційний сайт організації CERT – UA [Електронний ресурс]. – Режим доступу: https://cert.gov.ua/

32 Офіційний сайт Національного банку України [Електронний ресурс]. – Режим доступу: https://bank.gov.ua/control/uk/publish/article?art_id=37602713

33 Офіційний сайт Дніпровської міської ради [Електронний ресурс]. – Режим доступу: https://dniprorada.gov.ua/uk/articles/item/22004/u-dnipri-vidkrili-centr-reaguvannya-na-kiberataki

34 Указ Президента України про рішення Ради національної безпеки і оборони України від 27 січня 2016 року "Про Стратегію кібербезпеки України" [Електронний ресурс]. – Режим доступу: http://zakon5.rada.gov.ua /laws/show/96/2016

35 Офіційний сайт форуму команд реагування на компютерні надзвичайні події [Електронний ресурс]. – Режим доступу: https://www.first.org/about/

36 Офіційний сайт проекту Common Vulnerability Scoring System [Електронний ресурс]. – Режим доступу: https://www.first.org/cvss/

37 Common Vulnerability Scoring System v3.0: Specification Document [Електронний ресурс]. – Режим доступу: https://www.first.org/cvss/specification-document

38 https://cve.mitre.org/about/index.html

39 https://cve.mitre.org/about/faqs.html#what_is_cve_id

40 https://cve.mitre.org/about/faqs.html#MITRE_role_in_cve

41 https://cve.mitre.org/cve/cna.html

42 CVE Numbering authorities [Електронний ресурс]. – Режим доступу: https://cve.mitre.org/cve/request_id.html

43 База даних вразливостей CVE Details [Електронний ресурс]. – Режим доступу: https://www.cvedetails.com

44 База даних вразливостей NVD [Електронний ресурс]. – Режим доступу: https://nvd.nist.gov/vuln

45 База даних вразливостей Vulnerability Notes Database [Електронний ресурс]. – Режим доступу: http://www.kb.cert.org/vuls

46 База даних вразливостей SecurityFocus [Електронний ресурс]. – Режим доступу: https://www.securityfocus.com/

47 Міжнародний стандарт ISO/IEC 29147 [Електронний ресурс]. – Режим доступу:http://standards.iso.org/ittf/PubliclyAvailableStandards/c045170_ISO_IEC_29147_2014.zip#en

48 Документ Symantec Software Security Vulnerability Management Process [Електронний ресурс]. – Режим доступу: https://www.symantec.com/content/en/us/global/security_response/data/security/Symantec_Software_Security_Vulnerability_Management_Process.pdf

49 Bug Bounty List [Електронний ресурс]. – Режим доступу: https://www.bugcrowd.com/bug-bounty-list/

50 А.В.Лукацький - «Як працює сканер безпеки» [Електронний ресурс]. – Режим доступу: http://citforum.ck.ua/internet/securities/scaner.shtml

51 Довідкове керівництво сканера nmap [Електронний ресурс]. – Режим доступу: https://nmap.org/man/ru/index.html

52 https://portswigger.net/burp/help/suite_gettingstarted

53 https://metasploit.help.rapid7.com/docs/metasploit-basics

54 Офіційний сайт виробника Kali Linux [Електронний ресурс]. – Режим доступу: https://www.kali.org

55 Офіційний сайт виробника BlackArch [Електронний ресурс]. – Режим доступу: https://blackarch.org/index.html

56 Офіційний сайт виробника ParrotSecurity [Електронний ресурс]. – Режим доступу: https://www.parrotsec.org

57 Офіційний сайт виробника BlackBox [Електронний ресурс]. – Режим доступу:https://backbox.org

58 Офіційний сайт виробника сканера Nessus [Електронний ресурс]. – Режим доступу: https://www.tenable.com/products/nessus/nessus-professional

59 Офіційний сайт компанії Агенство активного аудита [Електронний ресурс]. – Режим доступу: http://auditagency.com.ua/?r=blog&p=Pentest&lang=ru

60 Методика тестування OWASP Testing guide [Електронний ресурс]. – Режим доступу: https://www.owasp.org/images/1/19/OTGv4.pdf

61 Методика тестування PTES [Електронний ресурс]. – Режим доступу: http://www.pentest-standard.org/index.php/Main_Page

62 Information Systems Security Assessment Framework [Електронний ресурс]. – Режим доступу: http://www.oissg.org/files/issaf0.2.1.pdf

63 База даних експлойтів веб – додатків [Електронний ресурс]. – Режим доступу: https://www.exploit-db.com/webapps/

64 Стаття «Metasploit инструкция по применению» [Електронний ресурс]. – Режим доступу:
https://cryptoworld.su/metasploit-%D0%B8%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F-%D0%BF%D0%BE-%D0%B

F%D1%80%D0%B8%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D1%8E/

65 Using Burp Proxy [Електронний ресурс]. – Режим доступу: https://support.portswigger.net/customer/portal/articles/1783119-using-burp-proxy

66 Сайт виробника сканера sqlmap [Електронний ресурс]. – Режим доступу: http://sqlmap.org

67 Офіційний сайт виробника сканера Burp Suite, тестування XSS – атак [Електронний ресурс]. – Режим доступу: https://support.portswigger.net/customer/portal/articles/2325939-Methodology_Attacking%20Users_XSS_Using%20Burp%20to%20Manually%20Test%20For%20Reflected%20XSS.html

68 Testing for Weak SSL/TLS Ciphers [Електронний ресурс]. – Режим доступу: https://www.owasp.org/index.php/Testing_for_Weak_SSL/TLS_Ciphers,_Insufficient_Transport_Layer_Protection_(OTG-CRYPST-001)

69 Офіційний сайт виробника сканера nmap [Електронний ресурс]. – Режим доступу:https://nmap.org/nsedoc/scripts/ssl-enum-ciphers.html

70 Офіційний сайт компаній Berzha Security [Електронний ресурс]. – Режим доступу: https://berezhasecurity.com/index_uk.html

71 О.Г. Вагонова, Ю.О. Волотковська, Н.М. Романюк. – Методичні вказівки до виконання економічної частини дипломного проекту для студентів напряму підготовки 1701 Інформаційна безпека

72 Penetration testing, frequently asked questions [Електронний ресурс]. – Режим доступу: https://highbitsecurity.com/FAQ-penetrationtesting.php

73 Кодекс законів про праці України [Електронний ресурс]. – Режим доступу: http://zakon3.rada.gov.ua/laws/show/322-08/page3

74 Інформаія про заробітну плату спеціаліста з інформаційної безпеки [Електронний ресурс]. – Режим доступу: https://rabota.ua/zapros/%D1%81%D0%BF%D0%B5%D1%86%D0%B8%D0%B0%D0%BB%D0%B8%D1%81%D1%82-%D0%BF%D0%BE-%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D

0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9-%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D0%B8/pg2

75 Норми робочого часу [Електронний ресурс]. – Режим доступу: http://vizitc.kiev.ua/ru/infokonsultant/normi_rabochego_vremeni.html

ДОДАТОК А
КОПІЇ НАУКОВИХ ПУБЛІКАЦІЙ

УДК 004.415.53

Кот Леонид Леонидович, ст. гр. 125м-16-1

Научный руководитель: Кручинин Александр Владимирович, ст. преп. кафедры безопасности информации и телекоммуникаций

(Государственное ВУЗ «Национальный горный университет», г. Днепр, Украина)

УЯЗВИМОСТИ ВЕБ – ПРИЛОЖЕНИЙ И СРЕДСТВА ИХ ОБНАРУЖЕНИЯ

В данных тезисах выполнен обзор и анализ уязвимостей веб - приложений, существующих баз данных уязвимостей, их оценок и средств обнаружения.

Ключевые слова: уязвимость, CVE, vulnerability.

ВВЕДЕНИЕ

В современном обществе и бизнесе высока роль веб – приложений в построении эффективной работы.

Увеличение количества разнообразных веб – приложений приводит к возникновению в них уязвимостей, поскольку как в процессе разработки, так и эксплуатации приложений могут происходить ошибки их функционирования, приводящие к уничтожению или утечке информации.

Согласно данным отчета компании Trustwave [1], [2]:

— 98% протестированных приложений имели уязвимости;

— 95% мобильных приложений, имели уязвимости (из которых 35 % имели критические уязвимости);

— в 2015 году была обнаружена 21 уязвимость нулевого дня;

ОПРЕДЕЛЕНИЕ ПОНЯТИЯ УЯЗВИМОСТИ

Уязвимость программного обеспечения (ПО) – недостаток, который может привести к нарушению конфиденциальности, целостности или доступности информации.

Уязвимость может быть результатом ошибок программирования, недостатков, допущенных при проектировании, эксплуатации, а также ненадежных паролей, вирусов, скриптовых и SQL-инъекций[3].

Уязвимости можно классифицировать по этапам жизненного цикла ПО, на которых они появляются:

— уязвимости этапа проектирования;

— уязвимости этапа реализации;

— уязвимости этапа эксплуатации.

Для обнаружения и использования уязвимостей злоумышленники используют эксплойты.

Эксплойт – это программа, фрагмент кода программы или последовательность команд, которые используют уязвимости в ПО.

Целями атак могут быть: кража личных данных, захват контроля над системой (повышение привилегий), использование компьютера в качестве элемента ботнета для рассылки спама или выполнения DDoS-атак и т.д. [3].

Следует отметить, что существует временной интервал между открытием уязвимости и выходом патча ее исправления. Данный факт свидетельствует о дополнительной угрозе, поскольку в этом интервале эксплойты могут без проблем функционировать в системе.

Также существует понятие уязвимости нулевого дня или 0day эксплойты. 0day - уязвимости, а также вредоносные программы, против которых ещё не разработаны защитные механизмы. Это означает, что у разработчиков было 0 дней на исправление дефекта: уязвимость или атака становится публично известна до момента выпуска производителем программного обеспечения исправлений ошибки [4].

ОБЗОР БАЗ ДАННЫХ УЯЗВИМОСТЕЙ

Информация про уже найденные уязвимости содержится в специально созданных базах данных.

Одной из баз уязвимостей является Common Vulnerabilities and Exposures (CVE) компании MITRE [5].

CVE является единым промышленным стандартом описания уязвимостей. Каждой уязвимости присваивается свой идентификатор, который имеет следующий формат: CVE-YYYY-NNNN, где:

— CVE – префикс;
— YYYY – год обнаружения уязвимости;
— NNNN – порядковый номер (последовательность из 4 и более цифр);

Каждая запись об уязвимости включает:

— CVE – идентификатор;
— краткое описание уязвимости;
— ссылки на другие ресурсы, имеющие отношение к обнаружению уязвимости (отчеты, рекомендации и т.д.);

Процесс добавления уязвимости в базу содержит три этапа [5]:

— обработку — анализ, исследование и процесс приведения уязвимости к формату CVE;
— присвоение — назначение конкретной записи уязвимости идентификатора CVE;
— публикацию — добавление новой записи и публикация ее на интернет-ресурсе CVE;

Краткое описание уязвимости составляется сотрудниками специального отдела компании MITRE (MITRE's CVE Content Team), которые анализируют отчеты о найденных уязвимостях, исследуют любую противоречивую информацию или несовместимое использование терминологии, а затем составляют описание уязвимости, включающее всю необходимую информацию, чтобы пользователи могли легко найти уязвимость по идентификатору или различить похожие уязвимости [5].

Список известных уязвимых мест в ПО содержится в CWE (Common weakness enumeration) [6].

К другим крупным центрам детектирования уязвимостей относятся:

— национальный институт стандартов и технологий (National Institute of Standards and Technology — NIST) и его Национальная база данных уязвимостей (National Vulnerabilities Database — NVD) [7];

— группа чрезвычайного компьютерного реагирования Соединенных Штатов (United States Computer Emergency Readiness Team — US-CERT) с базой данных записей уязвимостей (Vulnerability Notes Database — VND) [8];

— проект SecurityFocus [9];

— компания Secunia [10];

ОБЗОР СИСТЕМЫ CVSS

Критичность уязвимостей, содержащихся в базах данных уязвимостей, рассчитывается с помощью системы CVSS [11].

Общая система оценки уязвимостей (CVSS) – это открытая схема, которая позволяет обмениваться информацией об IT-уязвимостях. Система оценки CVSS состоит из 3 метрик: базовая метрика, временная метрика и контекстная метрика. Каждая метрика представляет собой число (оценку) в интервале от 0 до 10 и вектор – краткое текстовое описание со значениями, которые используются для вывода оценки. Цель системы состоит в расставлении приоритетов для уязвимостей, чтобы в первую очередь исправлять те из них, которые представляют наибольшую опасность.

СРЕДСТВА ОБНАРУЖЕНИЯ УЯЗВИМОСТЕЙ

Современные средства обнаружения уязвимостей представлены в виде сканеров уязвимостей, как бесплатных, так и коммерческих, где бесплатно предоставляется только пробная версия. Основной принцип их функционирования заключается в эмуляции действий потенциального злоумышленника по осуществлению сетевых атак.

Современный сканер уязвимостей выполняет основные задачи:

— идентификацию доступных сетевых сервисов;

— идентификацию имеющихся уязвимостей сетевых сервисов;

— выдачу рекомендаций по устранению уязвимостей.

Немаловажной характеристикой сканера уязвимостей является совместимость с существующими базами данных уязвимостей (CVE). В этом случае при обнаружении в системе уязвимости сразу отображается актуальная информация по найденной уязвимости со ссылкой на ее описание в базе уязвимостей.

Эффективнее всего производить поиск уязвимостей с помощью специальных методик, таких как OWASP Testing guide [12] и инструментов, включенных в операционную систему Kali Linux [13].

Kali Linux содержит много полезных инструментов для тестирования защищенности веб – приложений путем проверки их отдельных компонентов, функций. Методика OWASP Testing guide содержит перечень компонентов/функций приложения, которые обязательны для проверки защищенности, а также рекомендованный набор инструментов для тестирования защищенности. Цель методик тестирования защищенности – определить последовательность использования инструментов для тестирования, чтобы промежуточные результаты одного этапа тестирования могли использоваться как входные данные для следующего этапа. Например, Metasploit Framework позволяет имитировать сетевую атаку и выявлять уязвимости системы. Для проведения атаки необходима информация об установленных на удаленном сервере сервисах и их версии, т.е. нужно дополнительное исследование с помощью таких инструментов, как Nmap.

ВЫВОД

Эффективность тестирования защищенности и результат поиска уязвимостей зависит от правильности выбора методик тестирования и инструментов тестирования, что подразумевает адаптацию существующих методик для тестирования конкретных веб – приложений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Отчет Trustwave [Электронный ресурс]. – [Режим доступа]: http://www2.trustwave.com/rs/815-RFM-693/ /images/2015_TrustwaveGlobalSecurityReport.pdf
2. Блог Trustwave [Электронный ресурс]. – [Режим доступа]: https://www.trustwave.com/Resources/Trustwave-Blog/Introducing-the-2016-Trustwave-Global-Security-Report/
3. Уязвимости. Википедия. [Электронный ресурс]. – [Режим доступа]: https://ru.wikipedia.org/wiki/Уязвимость_ (компьютерная_безопасность)
4. Уязвимость нулевого дня. Википедия. [Электронный ресурс]. – [Режим доступа]: https://ru.wikipedia.org/wiki/Уязвимость_нулевого_дня
5. CVE Mitre [Электронный ресурс]. – [Режим доступа]: http://cve.mitre.org
6. CWE Mitre [Электронный ресурс]. – [Режим доступа]: http://cwe.mitre.org
7. База уязвимостей NIST [Электронный ресурс]. – [Режим доступа]: https://nvd.nist.gov
8. Vulnerability Notes Database [Электронный ресурс]. – [Режим доступа]: http://www.kb.cert.org/vuls
9. База уязвимостей SecurityFocus [Электронный ресурс]. – [Режим доступа]: http://www.securityfocus.com
10. База уязвимостей Secunia [Электронный ресурс]. – [Режим доступа]: https://secuniaresearch.flexerasoftware.com /community/research/
11. Система CVSS [Электронный ресурс]. – [Режим доступа]: https://www.first.org/cvss/
12. OWASP Testing Guide [Электронный ресурс]. – [Режим доступа]: https://www.owasp.org/index.php/OWASP_ Testing_Guide_v4_Table_of_Contents
13. Kali Linux [Электронный ресурс]. – [Режим доступа]: https://www.kali.org
14. Metasploit framework [Электронный ресурс]. – [Режим доступа]: https://www.metasploit.com
15. Сканер Nmap [Электронный ресурс]. – [Режим доступа]: https://nmap.org

ДОДАТОК Б

Порівняльний аналіз структури запропонованої та існуючих методик

На рисунку Б.1 представлено схему відповідності запропонованої у дипломній роботі методики та існуючої методики OWASP Testing guide. Стрілками показано схожі етапи, з цього можна зробити висновок, що тривалість тестування за запропонованою методикою буде швидшим через те, що один етап цієї методики охоплює кілька етапів методики OWASP Testing guide. Проте для кожного випадку тестування та кожного веб – додатка вибір між цими методиками може бути різним, враховуючи складність архітектури додатка та кількість вимог до захищеності компонентів. Запропонована методика дозволяє оцінити базовий рівень захищеності від атак, проте OWASP Testing guide охоплює більше можливостей тестування.

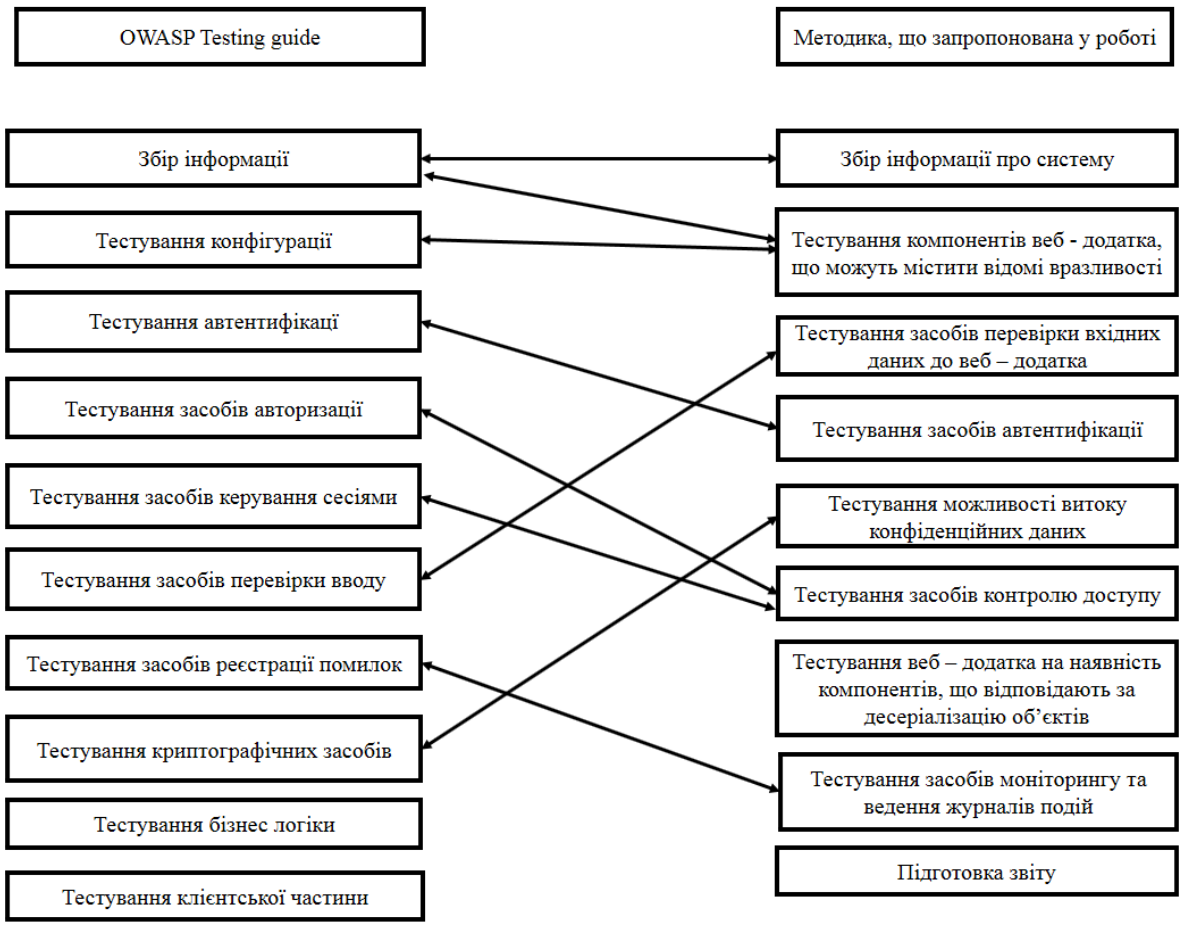

Рисунок Б.1. Відповідність етапів методики OWASP Testing Guide і запропонованої методики

ДОДАТОК В

ВІДГУК

на дипломну роботу магістра на тему:

«*Методики тестування на проникнення веб - додатків»*

студента групи 125м–16–1 *Кота Леоніда Леонідовича*

Мета дипломної роботи – підвищення ефективності методик тестування на проникнення веб - додатків.

Тема дипломної роботи безпосередньо пов’язана з об’єктом діяльності фахівця за спеціальністю 125 Кібербезпека – розвиток методик та засобів тестування на проникнення.

Задачі дипломної роботи (аналіз особливостей функціонування та вразливостей веб-додатків, аналіз законодавчої бази та організаційної структури протидії кібератакам в світі та в Україні, аналіз існуючих методик тестування на проникнення, обґрунтування вимог до методики, що розробляється, аналіз типових умов функціонування ІТС, розробка адаптованої методики тестування на проникнення, обґрунтування вибору засобів тестування) віднесені в освітньо-кваліфікаційній характеристиці магістра до класу евристичних, вирішення яких основане на знаково-розумових уміннях фахівця.

Оригінальність технічних рішень полягає у адаптації існуючих методик тестування на проникнення до тестування веб-додатків.

Практичне значення результатів проектування полягає у оптимізації проведення робіт з тестування веб-додатків на проникнення.

До недоліків дипломної роботи відносяться:

−недостатньо обґрунтовано вибір засобів проведення тестування;

−не в повному обсязі проведено випробування запропонованої методики.

Оформлення пояснювальної записки до дипломного проекту виконано з деякими відхиленнями від стандартів.

Ступінь самостійності виконання дипломної роботи висока.

За час дипломування Кот Л.Л. виявив себе фахівцем, здатним самостійно, на високому рівні вирішувати поставлені задачі.

В цілому дипломна робота виконана у відповідності до вимог, що ставляться до дипломної роботи магістра, заслуговує оцінки “добре”, а Кот Л.Л. присвоєння йому кваліфікації професіонал із організації інформаційної безпеки.

Керівник спеціальної частини
дипломної роботи магістра,
старший викладач _____ О.В. Кручінін

Керівник дипломної
роботи магістра,
к.ф-м.н., доц. _____ О.Ю. Гусєв

ДОДАТОК Г. Відгук керівника економічного розділу

ДОДАТОК Д. Перелік документів на оптичному носії

1 Пояснювальна записка

2 Презентація