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

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеню магістра

студента *Бардака Ігоря Андрійовича*

академічної групи *125м-19-2*

спеціальності *125 Кібербезпека*

спеціалізації[1]

за освітньо-професійною програмою *Кібербезпека*

на тему *Обґрунтування методів виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку*

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	к.т.н., доц. Герасіна О.В.			
розділів:				
спеціальний	к.т.н., доц. Герасіна О.В.			
економічний	к.е.н., доц. Пілова Д.П.			

Рецензент				

Нормоконтролер	ст. викл. Мєшков В.І.			

Дніпро
2020

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
_____ д.т.н., проф. Корнієнко В.І.

«_____»_____20___року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня магістр

студенту *Бардаку Ігорю Андрійовичу* **академічної групи** *125м-19-2*
(прізвище ім'я по-батькові) (шифр)

спеціальності *125 Кібербезпека*
(код і назва спеціальності)

на тему ***Обґрунтування методів виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку***

затверджену наказом ректора НТУ «Дніпровська політехніка» від _____ № ____

Розділ	Зміст	Термін виконання
Розділ 1	Розглянути методи пошуку вразливостей веб-додатків, розробити комплексну методику оцінки захищеності веб-додатків, реалізувати автоматизовану підсистему пошуку вразливостей згідно частини методики та висунутих до неї вимог	10.10.2020
Розділ 2	Виконати аналіз загроз веб-додатків та їх класифікацію, дослідити обрані вразливості та розробити методику їх виявлення	20.11.2020
Розділ 3	Обґрунтування економічної доцільності впровадження автоматизованої підсистеми пошуку вразливостей, та можливі збитки комерційного банку в разі наявності вразливостей в дистанційній системі обслуговування клієнтів	05.12.2020

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

Дата видачі: 01.09.2020р.

Дата подання до екзаменаційної комісії: 11.12.2020р.

Прийнято до виконання _____ Бардак І.А.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: ___ с., __ рис., __ табл., __ додатки, __ джерел.

Об’єкт дослідження: інформаційно-комунікаційна система комерційного банку. Предметом дослідження є методи та засоби виявлення вразливостей веб-додатків.

Мета роботи: підвищення ефективності захисту веб-додатків та адаптація існуючих методів пошуку вразливостей до задач забезпечення безпеки веб-додатків комерційного банку шляхом автоматизації.

У спеціальній частині розглянуті основні методи пошуку вразливостей веб-додатків, розроблена комплексна методика оцінки захищеності веб-додатків, реалізована автоматизована підсистема пошуку вразливостей згідно частини методики та висунутих до неї вимог. В роботі проведено аналіз загроз веб-додатків та їх класифікація, дослідження обраних вразливостей та автоматизованих систем їх пошуку.

В економічному розділі виконано розрахунки, що довели економічну доцільність впровадження автоматизованої підсистеми пошуку вразливостей, та можливі збитки комерційного банку в разі наявності вразливостей в дистанційній системі обслуговування клієнтів.

Результати роботи мають практичне значення та будуть використані комерційним банком для подальшого аналізу захищеності веб-додатків.

ВЕБ-ДОДАТКИ, КОМЕРЦІЙНИЙ БАНК, АНАЛІЗ ЗАГРОЗ, ВРАЗЛИВОСТІ, НЕСАНКЦІОНОВАНИЙ ДОСТУП, АВТОМАТИЗАЦІЯ.

РЕФЕРАТ

Пояснительная записка: ___ с., __ рис., __ табл., __ приложений, __ источников.

Объект исследования: информационно-коммуникационная система коммерческого банка. Предметом исследования являются методы и средства обнаружения уязвимостей веб-приложений.

Цель работы: повышение эффективности защиты веб-приложений и адаптация существующих методов поиска уязвимостей к задачам обеспечения безопасности веб-приложений коммерческого банка путем автоматизации.

В специальной части рассмотрены основные методы поиска уязвимостей веб-приложений, разработана комплексная методика оценки защищенности веб-приложений, реализована автоматизированная подсистема поиска уязвимостей согласно части методики и предъявляемых к ней требований. В работе проведен анализ угроз веб-приложений и их классификация, исследование выбранных уязвимостей и автоматизированных систем их поиска.

В экономическом разделе выполнены расчеты, которые доказали экономическую целесообразность внедрения автоматизированной подсистемы поиска уязвимостей, и возможные убытки коммерческого банка в случае наличия уязвимостей в дистанционной системе обслуживания клиентов.

Результаты работы имеют практическое значение, и будут использованы коммерческим банком для дальнейшего анализа защищенности веб-приложений.

ВЕБ-ПРИЛОЖЕНИЯ, КОММЕРЧЕСКИЙ БАНК, АНАЛИЗ УГРОЗ, УЯЗВИМОСТИ, НЕСАНКЦИОНИРОВАННЫЙ ДОСТУП, АВТОМАТИЗАЦИЯ.

ABSTRACT

Explanatory note: ___ p., __ fig., __ tab., __ application, __ sources.

Object of study: information and communication system of commercial bank. The subject of the study are methods and tools to detect vulnerabilities of Web applications.

The aim of the degree work: improving the efficiency of web application protection and adaptation of existing methods of search for vulnerabilities to problems of secure Web-based applications by automating in the commercial bank.

In the special part of the basic methods of searching web application vulnerabilities, developed a comprehensive method of estimating web application security, implemented an automated vulnerability scan subsystem according to the methodology and of the demands made on it. The paper analyzes the threats to web applications and their classification, the study selected vulnerabilities and automated retrieval.

In the economic section, calculations that are proven economic feasibility of an automated vulnerability scan subsystem, and possible loss of commercial bank in the case of remote vulnerabilities in the system of customer service.

The results of the work are of practical importance and will be used by a commercial bank for further analysis of web application security.

WEB APPLICATIONS, COMMERCIAL BANKS, THREAT ANALYSIS, VULNERABILITY, UNAUTHORIZED ACCESS, AUTOMATION.

СПИСОК УМОВНИХ СКОРОЧЕНЬ

АС – автоматизована система;

БД – база даних;

ДПА – державна податкова адміністрації;

ЕОМ – електрона обчислювальна машина;

ІБ – інформаційна безпека;

ІзОД – інформація з обмеженим доступом;

КБ – комерційний банк;

КЗЗ – комплекс засобів захисту;

КС – комп'ютерна система;

КСЗІ – комплексна система захисту інформації;

МВС – міністерство внутрішніх справ;

НД ТЗІ – нормативний документ системи технічного захисту інформації;

НСД – несанкціонований доступ;

ОЕ – об'єкт експертизи;

ОІД – об'єкт інформаційної діяльності;

ОС – операційна система;

ПАТ – публічне акціонерне товариство;

ПД – персональні дані;

ПЗ – програмне забезпечення;

ПІБ – Прізвище, ім'я, по-батькові;

СБУ – служба безпеки України;

СУБД – система управління базою даних;

ANSI – American National Standards Institute;

CMS – Content Management System;

DDoS – Distributed Denial of Service;

DNS – Domain Name System;

E-Mail – Electronic Mail;

FTP – File Transfer Protocol;

HTML – Hypertext Markup Language;

HTTP – Hypertext Transfer Protocol;

HTTPS – Hypertext Transfer Protocol Secure;

IP – Internet Protocol;

ISO – International Organization for Standardization;

NVD – National Vulnerability Database;

OSI – Open Systems Interconnection Basic Reference Model;

OSVDB – Open Sourced Vulnerability Database;

OWASP – The Open Web Application Security Project;

PA-DSS - The Payment Application Data Security Standard;

PCI-DSS – Payment Card Industry Data Security Standard;

PDO – PHP Data Objects;

ROSI – Return on Investment for Security;

SMTP – Simple Mail Transfer Protocol;

SQL – Structured Query Language;

TCP - Transmission Control Protocol;

URI – Uniform Resource Identifier;

URL – Uniform Resource Locator;

WASC TC – Web-Application Security Consortium Threat Classification;

XSS – Cross-site scripting.

ЗМІСТ

с.

ВСТУП

Розвиток інформаційних технологій, активне впровадження комп'ютерів у сфери діяльності людини зумовлює економічний розвиток держави, що неможливий без добре розвиненої банківської системи, на стан якої впливають як внутрішня економічна і політична ситуації, так і зміни світової банківської системи.

Однією з основних особливостей сучасної банківської системи є стрімкий розвиток комп'ютерних і телекомунікаційних технологій, розвиток мережевих технологій, що скорочує час обробки інформації, дозволяє провести комплексну автоматизацію діяльності, розробити механізми дистанційного обслуговування клієнтів і запропонувати новий асортимент послуг.

Зростають частки ринку мобільних пристроїв, а разом з тим росте і частка залученості користувачів в Інтернет (за даними компанії Gfk Ukraine на 1 квартал 2019 року активна частка Інтернет користувачів України складає 14.71 млн. чоловік[1]). Разом із зростанням активних користувачів веб-ресурсів зростає і сама мережа Інтернет, кількість сайтів в останні роки збільшилася на сотні мільйонів. За даними міжнародної компанії Netcraft, в листопаді 2019 року було більш 625 млрд. доменних імен і більш 186 млрд. активних сайтів[2].

Актуальність дослідження безпеки веб-додатків обумовлена збільшенням кількості банків, що надають послуги дистанційного обслуговування. Інформаційні системи почали істотно впливати на прибутковість банків, їх конкурентоспроможність і привабливість для клієнтів.

Разом з технологічним розвитком зростають і загрози несанкціонованого втручання в роботу обчислювальних машин, перешкоджання їх роботі, або порушення цілісності інформації, що оброблюється в банківських веб-додатках. Постійно виникають нові загрози, знаходяться нові вразливості в програмному забезпеченні. Виникає великий ризик для банку втратити кошти своїх клієнтів, або репутацію внаслідок невдалого захисту.

Захист інформації, що обробляється в автоматизованих системах (АС), полягає в створенні і підтримці в дієздатному стані системи заходів, як технічних (інженерних, програмно-апаратних), так і нетехнічних (правових, організаційних), що дозволяють запобігти або ускладнити можливість реалізації загроз, а також знизити потенційні збитки. Іншими словами, захист інформації спрямований на забезпечення безпеки оброблюваної інформації і АС в цілому, тобто такого стану, який забезпечує збереження заданих властивостей інформації і АС, що її обробляє.

За даними OSVDB (англ. The Open Source Vulnerability Database) за останні десятиліття кожен рік публікувалися кілька тисяч вразливостей, а за даними NVD (англ. National Vulnerability Database) і CVE-MITRE (англ. Common Vulnerabilities and Exposures) всього в веб-додатках було знайдено щонайменше 58000 вразливості[3]. Тому питання захисту інформації в банківському програмному забезпеченні є актуальним та потребує детального дослідження та вирішення.

Метою роботи є дослідження методів виявлення вразливостей веб-додатків комерційного банку та адаптація існуючих методів пошуку вразливостей до задач забезпечення безпеки веб-додатків комерційного банку. Підвищення рівня захищеності програмного забезпечення відбувається шляхом виявлення вразливостей та автоматизації цього процесу.

РОЗДІЛ 1. СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ

1.1 Стан питання

Актуальність дослідження веб-додатків обумовлена збільшенням кількості банків, що надають послуги дистанційного обслуговування. Інформаційні системи почали істотно впливати на прибутковість банків, їх конкурентоспроможність і привабливість для клієнтів.

Метою дослідження є підвищення ефективності захисту веб-додатків шляхом адаптації методів автоматизованого пошуку вразливостей та адаптації існуючих методів пошуку вразливостей до задач забезпечення безпеки веб-додатків комерційного банку.

Наукова новизна роботи полягає в адаптації існуючих методів пошуку вразливостей до задач забезпечення безпеки веб-додатків комерційного банку.

Об'єктом дослідження є і іформаційно-комунікаційна система комерційного банку.

Предметом дослідження є методи та засоби виявлення вразливостей веб-додатків.

Задачами дослідження є:

– аналіз загроз веб-додатків та їх класифікація;

– дослідження обраних вразливостей згідно з НД ТЗІ 2.7-009-09 "Методичні вказівки з оцінювання функціональних послуг безпеки в засобах захисту інформації від несанкціонованого доступу";

– аналіз існуючих автоматизованих систем пошуку вразливостей;

– адаптація існуючих методів пошуку вразливостей до задач забезпечення безпеки веб-додатків комерційного банку.

1.2 Організаційна структура комерційного банку

Вищим органом управління комерційного банку є загальні збори акціонерів, які проводяться не рідше одного разу на рік. Загальне керівництво

діяльністю банку здійснює рада банку. Безпосередньо діяльністю комерційного банку керує правління. Правління складається з голови правління (президента), його заступників (віце-президентів) і інших членів.

Компетенція загальних зборів акціонерів:

– приймає рішення про розширення числа учасників або їх вихід з банку;

– обирає раду банку, ревізійну комісію і визначає термін їх повноважень;

– приймає рішення про розмір і зміну статутного фонду;

– визначає розмір пайового внеску;

– затверджує статут банку, положення про раду, правління, ревізійної комісії та вносить до них зміни;

– розглядає і затверджує річний баланс банку, звіт про прибутки і збитки, висновок і звіт ревізійної комісії;

– розподіляє прибуток і вирішує інші визначальні питання діяльності банку.

Рада банку:

– призначає та звільняє з посади голову і членів правління банку;

– вносить пропозиції зборам акціонерів (пайовиків) про збільшення (зменшення) статутного фонду, зміну і доповнення статуту банку та з інших питань, що підлягають розгляду зборами;

– визначає основні умови надання кредитів;

– вирішує питання про відкриття філій і представництв банку;

– визначає структуру і чисельність банку, його філій та представництв;

– контролює роботу правління.

Правління банку:

– організовує та здійснює керівництво оперативною діяльністю банку та забезпечує виконання рішень зборів акціонерів (пайовиків) і ради банку;

– затверджує положення про структурні підрозділи, філії та представництва банку;

– вирішує питання підбору, підготовки та використання кадрів;

– розглядає і вирішує інші питання діяльності банку.

При правлінні банку звичайно створюються кредитний комітет і ревізійна комісія. У функції кредитного комітету входять: розробка кредитної політики банку, структури залучених засобів і їхнього розміщення; розробка висновків по наданню найбільш великих позик. Ревізійна комісія обирається загальними зборами учасників і підзвітна раді банку. До складу ревізійної комісії не можуть бути обрані члени ради і правління комерційного банку. Правління банку надає в розпорядження ревізійної комісії всі необхідні для проведення ревізії матеріали. Результати проведених перевірок комісія направляє правлінню банку. Організаційна структура формується з різних служб та підрозділів, за якими закріплені певні обов'язки. Типова організаційна структура комерційного банку зображена на рисунку 1.1.

1.3 Інформація, що оброблюється в комерційному банку

У веб-додатках, що використовуються банком може міститись відкрита інформація: про умови та надання банківських послуг, тарифи, курси валют, перелік відділень та банкоматів та інше.

Згідно зі ст. 60 Закону України “Про банки і банківську діяльність” до банківської таємниці належить інформація про діяльність і фінансовий стан клієнта, що стала відома банку у процесі його обслуговування і взаємовідносин із ним або з третіми персонами під час надання послуг банком, розголошення якої може завдати матеріальної чи моральної шкоди банку та клієнту[4].

Тлумачення поняття комерційної таємниці дається у ст. 30 Закону України “Про підприємства в Україні”. Зокрема у статті вказується, що під комерційною таємницею підприємства розуміють відомості, пов’язані з виробництвом, технологічною інформацією, управлінням фінансами та іншою діяльністю підприємства, що не є державною таємницею, розголошення (передавання, витік) яких може завдати шкоди його інтересам[5].

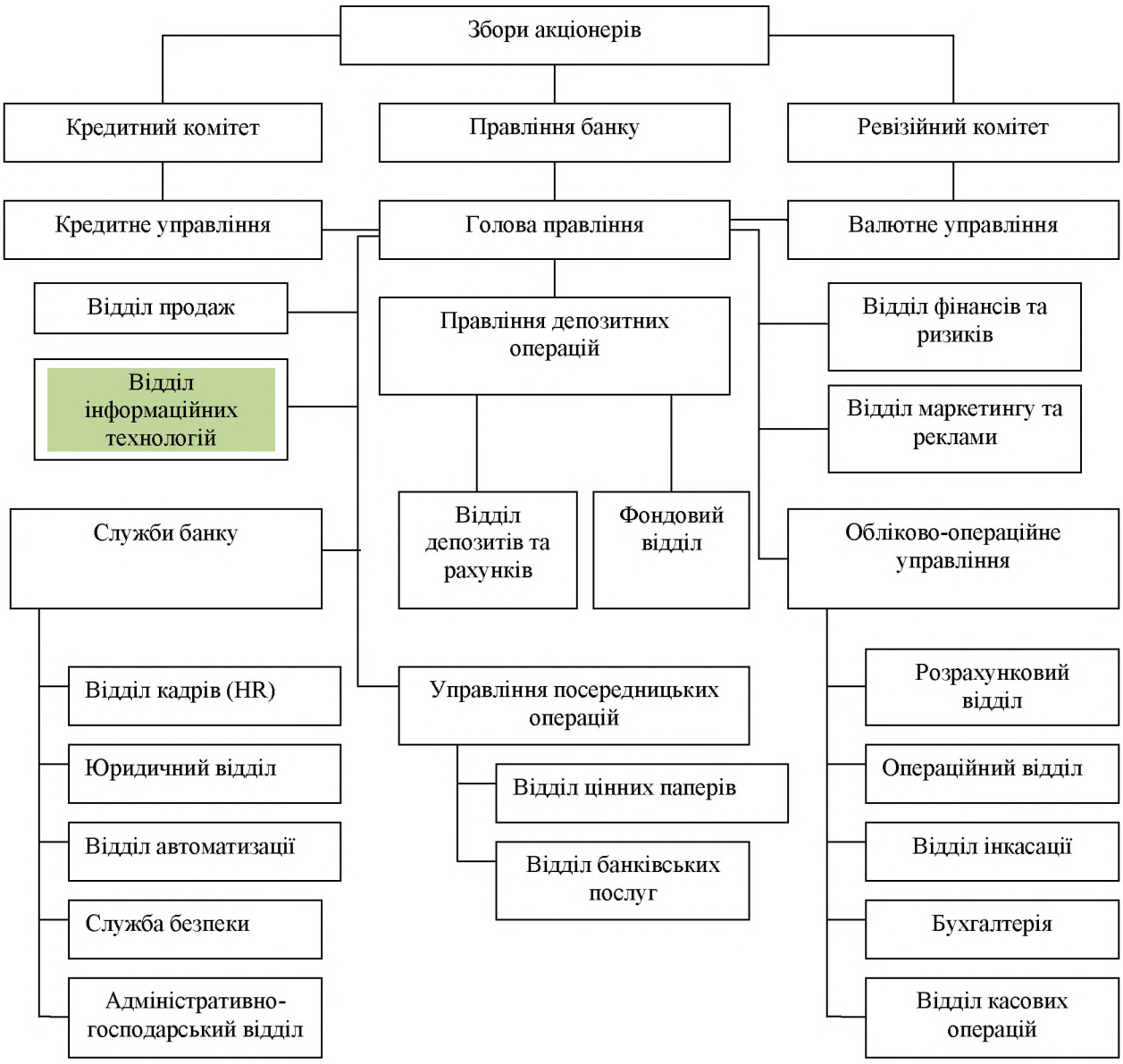

Рисунок 1.1 – Типова структура комерційного банку

Поняття “конфіденційної інформації” наведено в Законі України “Про інформацію” (ст. 30), де зазначено, що така інформація за своїм правовим режимом є інформацією з обмеженим доступом і її становлять “відомості, які знаходяться у володінні, користуванні або розпорядженні окремих фізичних чи юридичних осіб і поширюються за їх бажанням відповідно до передбачених ними умов” [6].

Розрізняючи поняття банківської та комерційної таємниць, конфіденційної інформації, законодавчо встановлено і різний правовий режим доступу до такої інформації.

Правовій режим доступу до банківської таємниці встановлено Законом України “Про банки і банківську діяльність”. Згідно зі ст. 62 цього Закону інформація, що становить банківську таємницю фізичних осіб – клієнтів банку, розкривається банком на письмовий запит або з письмового дозволу власника інформації, а також на письмову вимогу або за рішенням суду. Водночас банківська таємниця юридичних осіб – клієнтів банку, крім зазначених умов, розкривається у таких випадках:

– органам прокуратури Україні, СБУ, МВС Україні на їх письмові вимоги щодо операцій за рахунками конкретної юридичної особини або фізичної особи – суб’єкта підприємницької діяльності за конкретний проміжок часу;

– органам ДПА (Державної Податкової Адміністрації) України на їх письмову вимогу з питань оподаткування або валютного контролю щодо операцій за рахунками конкретної юридичної особи або фізичної особи – суб’єкта підприємницької діяльності за конкретний проміжок часу[4].

Інакше встановлено режим доступу до комерційної таємниці. Згідно з ч. 2 ст. 30 Закону України “Про підприємства в Україні” порядок доступу до відомостей, що становлять комерційну таємницю, визначає керівник підприємства (комерційного банку). Цією ж статтею надається право керівникові визначати склад і обсяг таких відомостей[5].

Так само визначено і режим доступу до конфіденційної інформації. Відповідно до ч. 3 ст. 30 Закону Україні “Про інформацію” власникам конфіденційної інформації надано право самим включати її до категорії конфіденційної, визначати режим доступу до неї та встановлювати систему (способи) її захисту.

Окремо визначено режим доступу до інформації, яка зберігається в автоматизованих системах, що для банків є особливо важливим, оскільки більшість банківської інформації міститься саме в автоматизованих системах. Законодавством встановлено, що доступ до інформації, яка зберігається,

обробляється і передається в автоматизованих системах, здійснюється згідно з правилами розмежування доступу, які встановлюються власником інформації чи уповноваженою ним особою (ст. 6 Закону України “Про захист інформації в інформаційно-телекомунікаційних системах”). Оволодіння інформацією, щодо якої встановлено обмежений доступ, може здійснюватись у разі її мимовільного витоку, розголошення або несанкціонованого доступу до неї. Витік інформації розглядається як мимовільне поширення інформації, зумовлене технічними або експлуатаційними особливостями певного обладнання, втратою, пошкодженням, знищенням документальних та програмних носіїв інформації внаслідок дії стихійного лиха, поширення інформації через потрапляння в інформаційні мережі комп’ютерних вірусів та інші випадки, які не мають навмисного характеру[7].

Захисту (організаційними, інженерними та технічними засобами) підлягають відомості комерційного банку, що віднесені до банківської та комерційної таємниці, а також відкрита інформація. Найвищій гриф обмеження доступу в даному випадку – “конфіденційно”.

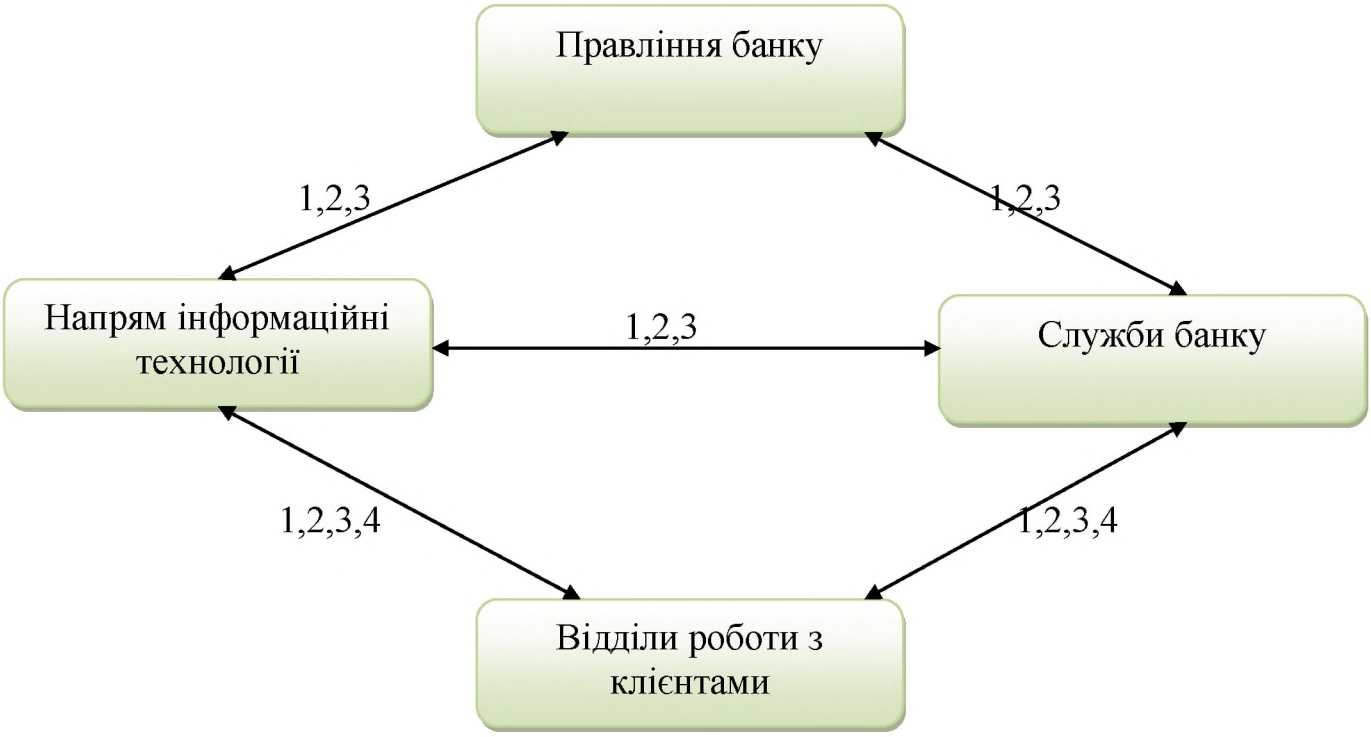

Рисунок 1.2 – Схема інформаційних потоків

Важливий аспект діагностики банку – аналіз інформаційних потоків, він допомагає зрозуміти механізм роботи підприємства. У процесі вивчення інформаційних зв'язків й інформаційних потоків досліджуються процеси виникнення, рухи й обробки інформації, а також спрямованість та інтенсивність документообігу на підприємстві. Типова схема інформаційних потоків наведена на рисунку 1.2. Відкрита інформація позначена цифрою 1, комерційна таємниця – 2, банківська таємниця – 3, персональні дані – 4.

1.4 Аналіз загроз та вразливостей веб-додатків

В механізмі автентифікації (Authentication). Розділ описує загрози для методів перевірки ідентифікатора користувача, служби або програми та спрямовані на обхід або експлуатацію вразливостей в механізмах реалізації автентифікації, що використовуються веб-додатком.

Підбір, або атака грубої сили (Brute force). Це процес перевірки всіх можливих вхідних значень створених псевдовипадковим генератором або за словником, що використовується для того, щоб підібрати ім'я користувача, пароль, ключ шифрування або іншу інформацію. Багато систем дозволяють використовувати слабкі паролі або ключі шифрування. Також може бути наявним людський фактор – користувачі можуть обрати пароль, що легко вгадується або міститься в словниках популярних парольних комбінацій. Використовуючи цю ситуацію, зловмисник може спробувати підібрати необхідні дані власноруч або використати спеціальне програмне забезпечення прискорення та полегшення процесу злому механізму автентифікації. Якщо випробуваний пароль дозволяє отримати доступ до конфіденційної інформації, атака вважається успішною, і атакуючий може протизаконно використати обліковий запис.

Подібна техніка проб і помилок може бути використана для підбору ключів шифрування. У разі використання системою ключів недостатньої

довжини, зловмисник може отримати секретний ключ, протестувавши всі можливі комбінації за певний проміжок часу.

Існує два види підбору: прямий і зворотний. При прямому підборі використовуються різні варіанти пароля для одного ідентифікатора (логіна) користувача. При зворотному – перебираються різні ідентифікатори користувачів, а пароль залишається незмінним. У системах з мільйонами облікових записів ймовірність використання різними користувачами одного пароля досить висока.

Ця атака може бути надзвичайно ефективною, адже число комбінацій може прямувати до нескінченності, але завжди буде обмеженим граничним значенням, проте підбір може займати кілька годин, днів або років.

Недостатня автентифікація або неповне обмеження повноважень (Insufficient Authentication). Вразливість, що дозволяє зловмиснику отримувати доступ до важливої інформації або функцій веб-додатку без належної автентифікації. В залежності від специфіки веб-додатку, його компоненти не повинні бути доступні без належної автентифікації.

Щоб не використовувати автентифікацію деякі ресурси "ховаються" за певною URL-адресою, яка не вказана на основних сторінках веб-додатку або інших загальнодоступних ресурсах. Однак, подібний підхід не більш ніж "безпека через приховування". Важливо розуміти, що, не дивлячись на те, що зловмисник не знає адреси сторінки, що потребує захисту, вона все одно доступна в мережі Internet.

Необхідна URL-адреса може бути знайдена перевіркою типових файлів і директорій (таких як /install/, /admin/, /logs/), з використанням повідомлень про помилки, журналів перехресних посилань або шляхом вивчення документації.

Небезпечне відновлення паролів (Weak Password Recovery Validation). Ця загроза виникає в разі, коли веб-додаток дозволяє атакуючому несанкціоновано отримувати, модифікувати паролі інших користувачів шляхом використання механізму нагадування або відновлення паролю.

Прикладом реалізації подібної функції є використання "секретного питання", відповідь на яке вказується в процесі реєстрації. Зазвичай питання обираються із списку, або вводиться користувачем самостійно, або користувачеві дозволяється вказати "підказку", яка допоможе йому згадати пароль. Інші способи вимагають від користувача вказати частину персональних даних, таких як ПІБ, дату народження, номер телефону, домашню адресу або інше, які потім будуть використовуватися для відновлення паролю. Після того як користувач доведе свою ідентичність, веб-додаток відобразить новий пароль або перешле його електронною поштою.

Вразливості, пов'язані з недостатньою перевіркою при відновленні пароля виникають внаслідок слабкості механізму, коли інформацію, що використовується для перевірки користувача, легко вгадати, або сам процес підтвердження можна обійти іншим несанкціонованим та не задокументованим шляхом, або система відновлення пароля може бути скомпрометована шляхом використання підбору. Додаткова інформація про користувача може бути легко отримана зловмисником із соціальних мереж, телефонного довідника або інших відкритих джерел.

В механізмі авторизації (Authorization). Загрози безпеці в методах, які використовуються веб-додатком для визначення того, чи має користувач, служба або додаток необхідний дозвіл системи для вчинення дії. В більшості системах користувачі діляться на певні ролі, а саме адміністраторів, модераторів, зареєстрованих користувачів та інших, і отримання деякої інформації або функції обмежено і дозволено тільки певним користувачам. Використовуючи різні техніки, зловмисник може підвищити свої привілеї і отримати несанкціонований доступ до захищених ресурсів.

Передбачуване значення ідентифікатора сесії (Credential/Session Prediction). Передбачуване значення ідентифікатора сесії дозволяє перехоплювати сесії інших користувачів. Подібні атаки виконуються шляхом передбачення або вгадування унікального ідентифікатора сесії користувача. У

разі проведення успішної атаки зловмисник зможе виконати дії з правами скомпрометованого користувача. Принцип роботи багатьох ВЕБ-сайтів припускає автентифікацію користувача при першому зверненні і подальше відстеження його сесії. Для цього користувач вказує комбінацію облікового запису та пароля до нього.

Замість повторної передачі імені користувача та пароля при кожній операції, веб-сервер генерує унікальний ідентифікатор, який присвоюється сесії користувача. Наступні запити користувача до сервера містять ідентифікатор сесії як доказ того, що автентифікація була успішно пройдена. Якщо атакуючий може передбачити або вгадати значення ідентифікатора іншого користувача, це може бути використано для несанкціонованого доступу.

Недостатня авторизація (Insufficient Authorization). Недостатня авторизація виникає, коли веб-сервер дозволяє атакуючому отримувати доступ до важливої інформації або функцій, доступ до яких повинен бути обмежений. Те, що користувач пройшов автентифікацію не означає, що він повинен отримати доступ до всіх функцій і вмісту веб-сайту. Крім автентифікації має бути реалізовано розмежування доступу.

Процедура авторизації визначає, які дії може вчиняти користувач, служба або додаток. Правильно побудовані правила доступу повинні обмежувати дії користувача відповідно до політики безпеки. Повний доступ до всієї конфіденційної інформації повинен бути дозволений тільки адміністраторам.

Відсутність тайм-ауту сесії (Insufficient Session Expiration). У разі, якщо для ідентифікатора сесії або облікових даних не передбачений тайм-аут або його значення занадто велике, зловмисник може скористатися застарілими даними для авторизації. Це підвищує вразливість сервера до атак, пов'язаних з крадіжкою ідентифікаційних даних. Оскільки протокол HTTP не передбачає контроль сесії, веб-сервера зазвичай використовують ідентифікатори сесії для визначення запитів користувача. Таким чином, кожний ідентифікатор сесії має бути конфіденційним, щоб запобігти множинний доступ користувачів з одним

обліковим записом. Викрадений ідентифікатор може використовуватися для доступу до облікового запису користувача або здійснення шахрайських операцій. Відсутність таймаута сесії збільшує ймовірність успіху різних атак. Наприклад, зловмисник може отримати ідентифікатор сесії, використовуючи мережевий аналізатор або вразливість типу міжсайтового виконання сценаріїв. Хоча тайм-аут не допоможе у випадку, якщо ідентифікатор буде використаний негайно, обмеження часу дії допоможе у випадку більш пізніх спроб використання ідентифікатора.

В іншій ситуації, якщо користувач отримує доступ до сервера з публічного комп'ютера (бібліотека, Інтернет-кафе і т.д.), відсутність тайм-ауту сесії може дозволити зловмиснику скористатися історією браузера для перегляду сторінок користувача. Велике значення таймаута збільшує шанси підбору чинного ідентифікатора. Крім того, збільшення цього параметра веде до збільшення одночасно відкритих сесій, що ще більше підвищує ймовірність успішного підбору.

Фіксація сесії (Session fixation). Використовуючи даний клас атак, зловмисник присвоює ідентифікатору сесії користувача задане значення. Залежно від функціональних можливостей веб-сервера, існує кілька способів "зафіксувати" значення ідентифікатора сесії. Для цього можуть використовуватися атаки типу міжсайтового виконання сценаріїв або за допомогою попереднього HTTP-запиту. Після фіксації значення ідентифікатора сесії атакуючий очікує моменту, коли користувач увійде в систему. Після входу користувача, зловмисник використовує ідентифікатор сесії для отримання доступу до системи від імені користувача.

Без наявності активного захисту від фіксації сесії, ця атака може бути використана проти будь-якого веб-сервера, автентифікуючого користувачів за допомогою ідентифікатора сесії. Більшість веб-серверів зберігає ідентифікатор користувача в тимчасових cookies-файлах, але це значення так само може бути присутнім в URL-адресі або прихованому полі форми.

На відміну від крадіжки ідентифікатора, фіксація сесії надає зловмисникові набагато більше несанкціонованих повноважень. Це пов'язано з тим, що активна фаза атаки відбувається до входу користувача в систему. Атаки, спрямовані на фіксацію сесії зазвичай проходять в три етапи.

1 Встановлення сесії. Зловмисник встановлює сесію-заглушку на веб-сервері і отримує ідентифікатор або вибирає довільний. У деяких випадках встановлена сесія повинна підтримуватися в активному стані шляхом періодичних звернень до сервера.

2 Фіксація сесії. Зловмисник передає значення ідентифікатора сесії-заглушки браузеру користувача і фіксує його ідентифікатор сесії. Це можна зробити, наприклад, встановивши значення в тимчасових файлах браузера за допомогою XSS атаки.

3 Підключення до сесії. Атакуючий очікує автентифікації користувача на сервері. Після того, як користувач зайшов на сайт, зловмисник підключається до сервера, використовуючи зафіксований ідентифікатор, і отримує доступ до сесії користувача.

Загрози клієнтської частини веб-додатку (client-side security threats). Під час відвідування сайту, між користувачем і системою встановлюються довірчі відносини, як в технологічному, так і в психологічному аспектах. Користувач очікує, що сайт надасть йому легітимний вміст. Крім того, користувач не очікує атак з боку сайту. Експлуатуючи цю довіру, зловмисник може використовувати різні методи для проведення атак на клієнтів системи.

Міжсайтове виконання сценаріїв (Cross-site Scripting, XSS). Наявність вразливості Cross-site Scripting дозволяє атакуючому передати веб-серверу програмний код, який буде перенаправлено браузеру користувача. Цей код зазвичай створюється на мовах HTML / JavaScript, але можуть бути використані VBScript, ActiveX, Java, Flash, або інші технології, що підтримуються браузером. Найбільш небезпечним і поширеним результатом нападу з

допомогою XSS є отримання cookie-файлів цільового користувача в контексті уразливого сайту.

Якщо автентифікація в системі відбувається на основі cookie-файлів і якщо користувач, у якого були перехоплені cookie-файли в контексті уразливого сайту, був у цей момент автентифікований в системі, то для несанкціонованого отримання прав цього користувача нападаючому буде досить замінити cookie-файлів у своєму браузері на перехоплені дані.

Розщеплення HTTP-запиту (HTTP Response Splitting). За наявності даної вразливості зловмисник посилає веб-серверу спеціальним чином сформований запит, відповідь на який інтерпретується ціллю атаки як дві різні відповіді. Друга відповідь повністю контролюється зловмисником, що дає йому можливість підробити відповідь сервера [8]. У реалізації атак с розщепленням HTTP-запиту потрібно щонайменше три умови:

– веб-сервер що містить подібну вразливість;

– ціль атаки, що взаємодіє з веб-сервером під управлінням зловмисника. Типово в якості цілі виступає кешуючий сервер-посередник або кеш браузера;

– атакуючий, що ініціює атаку.

Можливість здійснення атаки виникає, коли сервер повертає дані, надані користувачем в заголовках HTTP-відповіді. Зазвичай це відбувається при перенаправленні користувача на іншу сторінку (коди HTTP 3xx) або коли дані, отримані від користувача, зберігаються в cookie-файлах. В першій ситуації URL, на який відбувається перенаправлення, є частиною заголовка Location HTTP відповіді, а в другому випадку значення cookie-файлів передаються в заголовку Set-cookie.

Основою розщеплення HTTP-запиту є впровадження символів переведення рядка (CR і LF) таким чином, щоб сформувати дві HTTP транзакції, в той час як реально буде відбуватися тільки одна. Зміщення рядка використовується для того, що б закрити першу (стандартну) транзакцію і сформувати другу пару питання / відповідь, повністю контрольовану

зловмисником і абсолютно непередбачену логікою програми. У результаті успішної реалізації цієї атаки зловмисник може виконати наступні дії:

– міжсайтове виконання сценаріїв (XSS);

– модифікація даних кеша сервера-посередника. Деякі кешуючі сервери-посередники (Squid, Net Cache, Apache Proxy і ряд інших), зберігають підроблену зловмисником відповідь на жорсткому диску і на наступні запити користувачів за даною адресою повертають дані з кешу. Це призводить до заміни сторінок сервера на клієнтській стороні. Крім цього, зловмисник може переправити собі cookie-файли користувача або присвоїти їм певне значення. Так само ця атака може бути спрямована на індивідуальний кеш браузера користувача;

– міжкористувацька атака (один користувач, одна сторінка, тимчасова підміна сторінки). При реалізації цієї атаки зловмисник не посилає додатковий запит. Замість цього використовується той факт, що деякі сервери-посередники розділяють одне TCP-з'єднання до сервера між декількома користувачами. У результаті другий користувач отримує у відповідь сторінку, сформовану зловмисником. Крім підміни сторінки зловмисник може також виконати різні операції з cookie-файлами користувача;

– перехоплення сторінок, що містять конфіденційні дані користувача. У цьому випадку зловмисник отримує відповідь сервера замість самого користувача. Таким чином, він може несанкціоновано отримати доступ до обмеженої інформації.

Виконання коду (Command Execution). Ця секція описує атаки, спрямовані на несанкціоноване виконання коду на веб-сервері. Дані, надані веб-серверу, користувачем, при обробці запитів, використовуються при складанні команд, що застосовуються для генерації динамічного вмісту. Якщо при розробці не враховуються вимоги безпеки, зловмисник отримує можливість несанкціоновано модифікувати команди, що виконуються.

Переповнення буфера (Buffer Overflow). Експлуатація переповнення буфера дозволяє зловмисникові змінити шлях виконання програми шляхом перезапису даних у пам'яті системи. Переповнення буфера є найбільш поширеною причиною помилок в програмах. Воно виникає, коли обсяг даних перевищує розмір виділеного під них буфера. Коли буфер переповнюється, дані переписують інші області пам'яті, що призводить до виникнення помилки. Якщо зловмисник має можливість управляти процесом переповнення, це може викликати ряд серйозних проблем. Переповнення буфера може викликати відмови в обслуговуванні, приводячи до пошкодження пам'яті і викликаючи помилки в програмах. Більш серйозні ситуації дозволяють змінити шлях виконання програми та виконати в її контексті різні дії. Це може відбуватися в декількох випадках. Використовуючи переповнення буферу, можна перезаписувати службові області пам'яті, наприклад, адреси повернень з функцій в стек. Також, при переповненні можуть бути переписані значення змінних в програмі. Переповнення буфера є найбільш поширеною проблемою в безпеці і нерідко наявною в веб-серверах. Проте атаки, що експлуатують цю уразливість, використовуються проти веб-додатків не дуже часто. Причина цього криється в тому, що атакуючому, як правило, необхідно проаналізувати вихідний код або образ програми. Оскільки атакуючому доводиться експлуатувати нестандартну програму на віддаленому сервері, йому доводиться атакувати "наосліп", що знижує шанси на успіх.

Переповнення буфера зазвичай виникає при створенні програм на мовах C і C++. Якщо частина сайту створена з використанням цих мов, сайт може бути вразливий для переповнення буфера.

Виконання команд ОС (OS Commanding). Атаки цього класу спрямовані на виконання команд операційної системи на веб-сервері шляхом маніпуляції вхідними даними. Якщо інформація, отримана від клієнта, належним чином не проходить верифікацію, атакуючий отримує можливість виконати команди ОС.

Вони будуть виконуватися з тим же рівнем привілеїв, з яким працює компонент веб-додатка, що виконує запит (сервер СУБД, веб-сервер і т.д.).

Впровадження операторів SQL (SQL Injection). Уразливість SQL source code injection (ін'єкція SQL-коду) виникає тоді, коли нападаючий може впроваджувати довільні дані в SQL-запити. SQL-ін'єкція – в деяких випадках критична вразливість в системі. І, незважаючи на всю небезпеку цієї вразливості, вона, безперечно, є однією з найпоширеніших вразливостей. Це надає зловмисникові можливість отримати несанкціонований доступ до даних.

Впровадження серверних розширень (SSI Injection). Атаки даного класу дозволяють зловмиснику передати виконуваний код, який надалі буде виконаний на веб-сервері. Вразливості, що призводять до можливості здійснення даних атак, зазвичай полягають у відсутності перевірки даних, наданих користувачем, перед збереженням їх в файл, що далі буде виконаний сервером. Перед генерацією HTML сторінки сервер може виконувати сценарії, наприклад Server–site Includes (SSI). У деяких ситуаціях вихідний код сторінок генерується на основі даних, наданих користувачем. Якщо атакуючий передає серверу оператори SSI, він може отримати можливість виконання команд операційної системи або включити в неї заборонений вміст при наступному відображенні.

В основі цієї уразливості лежить використання програмістом змінних всередині конструкції include(). Таким чином, щоб повністю уникнути появи цієї вразливості, потрібно просто не використовувати змінні всередині функції include (). За відсутності змінних всередині include() вразливість подібного типу відсутня за визначенням.

Розголошення інформації (Information Disclosure). Атаки даного класу спрямовані на отримання додаткової інформації про веб-додаток. Використовуючи ці уразливості, зловмисник може визначити використовувані дистрибутиви програмного забезпечення, номери версій клієнта і сервера і встановлені оновлення. В інших випадках, в розголошеній інформації системи

може міститися розташування тимчасових файлів або резервних копій. У багатьох випадках ці дані не потрібні для роботи користувача. Більшість серверів надають доступ до надмірного обсягу даних, однак необхідно мінімізувати обсяг службової інформації. Чим більшими знаннями про програмне забезпечення буде володіти зловмисник, тим легше йому буде скомпрометувати систему.

Індексування директорій (Directory Indexing). Надання списку файлів у директорії являє собою звичайну поведінку веб-сервера, якщо сторінка, яка відображається за замовчуванням (index.html / home.html / default.htm) відсутня. Коли користувач запитує основну сторінку сайту, він зазвичай вказуєте доменне ім'я сервера без імені конкретного файлу (http://www.example.ua/). Сервер переглядає основну папку, знаходить у ній файл, який використовується за замовчуванням, і на його основі генерує відповідь. Якщо такий файл відсутній, в якості відповіді може повернутися список файлів у директорії сервера.

Ця ситуація аналогічна виконанню команди "ls" (Unix) або "dir" (Windows) на сервері і форматуванню результатів у вигляді HTML. У цій ситуації зловмисник може отримати доступ до даних, що можуть бути конфіденційними. Досить часто адміністратори покладаються на "безпеку через приховування", припускаючи, що гіперпосилання на документ відсутнє, то він недоступний нелегітимним користувачам. Таким чином, ззовні безпечне індексування директорій може призвести до витоку важливої інформації, яка в подальшому буде використана для проведення атак на систему.

Ідентифікація додатків (Web-Server/Application Fingerprinting). Визначення версій додатків використовується зловмисником для отримання інформації про використовувані сервером і клієнтом операційні системи. Також ця атака може бути спрямована на інші компоненти веб-додатка, наприклад, службу каталогу або сервер баз даних або використовувані технології програмування.

Зазвичай подібні атаки здійснюються шляхом аналізу різної інформації, що надається веб-сервером, наприклад:

– особливості реалізації протоколу HTTP;

– заголовки HTTP-відповідей;

– використовувані сервером розширення файлів (.asp або.jsp);

– значення cookie-файлів (ASPSESSION і т.д.);

– повідомлення про помилки;

– структура каталогів і використовувана угода про імена (Windows / Unix).

Наявність детальної та точної інформації про програмне забезпечення, що використовується, дуже важлива для зловмисника, оскільки реалізація багатьох атак (наприклад, переповнення буфера) специфічна для кожного варіанту операційної системи або програми. Крім того, детальна інформація про інфраструктуру дозволяє знизити кількість помилок, і як наслідок – загальний «шум», вироблений атакуючим. Даний факт відзначений у HTTP RFC 2068, рекомендуючим щоб значення заголовка Server HTTP відповіді було параметром, який можливо налаштувати.

Витік інформації (Information Leakage). Ці вразливості виникають в ситуаціях, коли сервер публікує важливу інформацію, наприклад коментарі розробників або повідомлення про помилки, яка може бути використана для компрометації системи. Цінні з точки зору зловмисника дані можуть міститися в коментарях HTML, повідомленнях про помилки або просто бути присутнім в відкритому вигляді. Існує величезна кількість ситуацій, в яких може відбутися витік інформації. Не обов'язково вона призводить до виникнення уразливості, але часто дає атакуючому посібник для розвитку атаки. З витоком важливої інформації можуть виникати ризики різного ступеня, тому необхідно мінімізувати кількість службової інформації, доступної на клієнтській стороні.

Аналіз доступної інформації дозволяє зловмисникові провести розвідку і отримати уявлення про структуру директорій сервера, використовуваних SQL

запитах, назвах ключових процесів і програм сервера. Часто розробники залишають коментарі в HTML сторінках і коді сценаріїв для полегшення пошуку помилок та підтримки програми. Ця інформація може варіюватися від простих описів деталей функціонування програми до, в найгірших випадках, імен користувачів і паролів, що використовуються при налагодженні.

Витік інформації може відноситися і до конфіденційних даних, оброблюваних сервером. Це можуть бути ідентифікатори користувача (ПІБ користувачів, дати народження, телефони, номери водійських посвідчень, паспортів і т.д.), а також поточна інформація (баланс особового рахунку або історія платежів). Багато атак цієї категорії виходять за рамки захисту веб-додатків і переходять в галузь фізичної безпеки. Витік інформації в цьому випадку часто виникає, коли в браузері відображається інформація, яка не повинна виводитися у відкритому вигляді навіть користувачеві. Як приклад можна навести паролі користувача, номери кредитних карток і т.д.

Зворотний шлях в директоріях (Path Traversal). Дана техніка атак спрямована на отримання доступу до файлів та директорій, які знаходяться поза основною директорії веб-сервера, наприклад /etc/passwd. Зловмисник може маніпулювати параметрами URL з метою отримати доступ до файлів або здійснити команди, що розташовуються в файлової системі веб-сервера.

Багато Web-серверів обмежують доступ користувача певною частиною файлової системи, зазвичай званої "public_html" або "CGI root". Ці директорії мають файли, призначені для користувача і програми, необхідні для отримання доступу до функцій веб-додатку.

Більшість базових атак, що експлуатують зворотний шлях, засновані на впровадженні в URL символів ".. /", для того, щоб змінити розташування ресурсу, який буде оброблятися сервером. Оскільки більшість веб-серверів фільтрують цю послідовність, зловмисник може скористатися альтернативними кодуваннями для представлення символів переходу по директоріях. Популярні прийоми включають використання альтернативних кодувань, наприклад

Unicode ("..% u2216" або "..% c0%af"), використання зворотного слеша (".. \") на Windows-серверах, символів URL-Encode ("% 2e% 2e % 2f ") або подвійного кодування URL-Encode (" ..% 255c ").

Навіть якщо веб-сервер обмежує доступ до файлів певним каталогом, ця вразливість може виникати в сценаріях або CGI-програмах. Можливість використання зворотного шляху в каталогах досить часто виникає в додатках, що використовують механізми шаблонів або завантажують їх текст сторінок з файлів на сервері. У цьому варіанті атаки зловмисник модифікує ім'я файлу, що передається в якості параметра CGI-програми або серверного сценарію. У результаті зловмисник може отримати вихідний код сценаріїв. Досить часто до імені запитуваного файлу додаються спеціальні символи, такі як "% 00", з метою обходу перевірки розширення файлів.

Передбачуване розташування ресурсів (Predictable Resource Location). Передбачуване розташування ресурсів дозволяє зловмисникові отримати доступ до прихованих даних або функціональних можливостей. Шляхом підбору зловмисник може отримати доступ до вмісту, не призначеного для публічного перегляду. Тимчасові файли, резервні копії, файли конфігурації, скрипти інсталяції систем керування контентом часто є метою подібних атак. У більшості випадків перебір може бути оптимізований шляхом використання стандартної угоди про імена файлів і директорій сервера. Отримані зловмисником файли можуть містити інформацію про дизайн програми, інформацію з баз даних або паролі, шляхи до теки. Також «приховані» файли можуть містити уразливості, відсутні в основному додатку. На цю атаку часто посилаються як на помилку конфігурації сервера – "перерахування файлів і директорій" (Forced Browsing, File Enumeration, Directory Enumeration).

Логічні атаки (Logical Attacks). Атаки даного класу спрямовані на експлуатацію функцій додатка або логіки його функціонування. Логіка програми являє собою очікуваний процес функціонування програми при виконанні певних дій. В якості прикладів можна навести відновлення паролів,

реєстрацію облікових записів, транзакції в системах електронної комерції. Додаток може вимагати від користувача коректного виконання декількох послідовних дій для виконання певного завдання. Зловмисник може обійти або використовувати ці механізми в своїх цілях шляхом використання помилок в бізнес-логіці роботи веб-додатка або зверненням до певного функціонала за прямим посиланням.

Відмова в обслуговуванні (Denial of Service). DoS-атака (від англ. Denial of Service, відмова в обслуговуванні) і DDoS-атака (від англ. Distributed Denial of Service, розподілена відмова в обслуговуванні) – це різновиди атак на обчислювальну систему. Особливістю даного виду комп'ютерного злочину є те, що зловмисники не ставлять своєю метою незаконне проникнення в захищену комп'ютерну систему з метою крадіжки або знищення інформації. Мета даної атаки – паралізувати роботу веб-вузла. Головною небезпекою тут є простота організації і те, що ресурси зловмисників є практично необмеженими, тому що атака є розподіленою.

Недостатнє протидія автоматизації (Insufficient Anti-automation). Недостатня протидія автоматизації виникає, коли сервер дозволяє автоматично виконувати операції, які повинні виконуватись власноруч. Для деяких функцій програми необхідно реалізовувати захист від автоматичних атак. Автоматизовані програми можуть змінюватись від нешкідливих роботів пошукових систем до систем автоматизованого пошуку вразливостей і реєстрації облікових записів. Подібні роботи генерують тисячі запитів за хвилину, що може призвести до падіння продуктивності всієї програми. Протидія автоматизації полягає в обмеженні можливостей подібних утиліт. Наприклад, додаткові засоби ідентифікації людини (використання тесту Тюрінга) можуть запобігати автоматичній реєстрації сотень облікових записів системи електронної пошти.

По відношенню до інформації, що обробляється на веб-сервері та до користувачів веб-додатків існує ряд загроз, відображених в таблиці 1.1.

Ступінь ризику позначена цифрами: 1 (найнижчий), 2 (низький), 3 (середній), 4 (високий), 5 (найвищий).

Таблиця 1.1 – Загрози безпеки інформації веб-додатків

Загроза	Вразливість	Властивість інформації	Ступінь ризику
1	2	3	4
Антропогенні загрози			
Атаки на вразливості операційної системи або системного ПЗ	Вразливості операційної системи або системного ПЗ	конфіденційність, цілісність, доступність	4
Атаки на вразливості або помилки в програмному забезпеченні веб-додатка	Помилки в бізнес-логіці, функціоналі програмного забезпечення веб-додатків	конфіденційність, цілісність, доступність	5
Захоплення облікових записів користувачів веб-додатка	Порушення правил політики безпеки	конфіденційність	3
Підвищення навантаження на веб-сервер внаслідок збільшення користувачів	Помилки в архітектурі або адмініструванні ПЗ	доступність	3
Розподілена атака типу "відказ обслуговування"	Некоректна робота засобів захисту	доступність	3
Навмисне фізичне руйнування серверного обладнання	Недоліки фізичної охорони сервера	цілісність, доступність	1
Випадкове чи навмисне відключення засобів захисту	Помилки адміністрування	конфіденційність, цілісність, доступність	2
Техногенні загрози			
Відмови і збої обладнання, яке приймає, обробляє, зберігає й передає інформацію	Некоректна робота або вихід з ладу технічного компоненту АС	конфіденційність, цілісність, доступність	3
Неякісні програмні засоби	Некоректна робота ПЗ	конфіденційність, цілісність, доступність	2
Саботаж роботи інформаційної системи за допомогою програмних засобів (віруси, шкідливі скрипти та ін.).	Некоректна робота засобів захисту	конфіденційність, цілісність, доступність	4

Продовження таблиці 1.1

1	2	3	4
Стихійні загрози			
Руйнування технічних засобів, що обробляють інформацію, внаслідок стихійних впливів (землетруси, пожежі, електромагнітні поля та блискавки)	Порушення правил пожежної безпеки або електробезпеки	цілісність, доступність	1

1.5 Побудова моделі порушника

Порушник – це особа, яка помилково, внаслідок необізнаності, цілеспрямовано за злим умислом або без нього, використовуючи різні методи та засоби, здійснила спробу виконати операції, які призвели або можуть призвести до порушення, визначених політикою безпеки інформації, властивостей інформації [9], модель впливу порушника на інформаційну систему зображена на рисунку 1.3. Класифікація порушників зазначена в таблиці 1.2.

В залежності від цілей порушників, вони можуть бути класифіковані за такими критеріями як наявність доступу до ОІД, час вчинення дії, мотиви, рівень кваліфікації та за іншими ознаками, що вказані в таблиці 1.3.

Модель порушника — абстрактний формалізований або неформалізований опис дій порушника, який відображає його практичні та теоретичні можливості, апріорні знання, час та місце дії та інше. По відношенню до АС порушники можуть бути внутрішніми (з числа співробітників, користувачів системи) або зовнішніми (сторонні особи або будь-які особи, що знаходяться за межами контрольованої зони) [10]. Модель порушника визначена в таблиці 1.4.

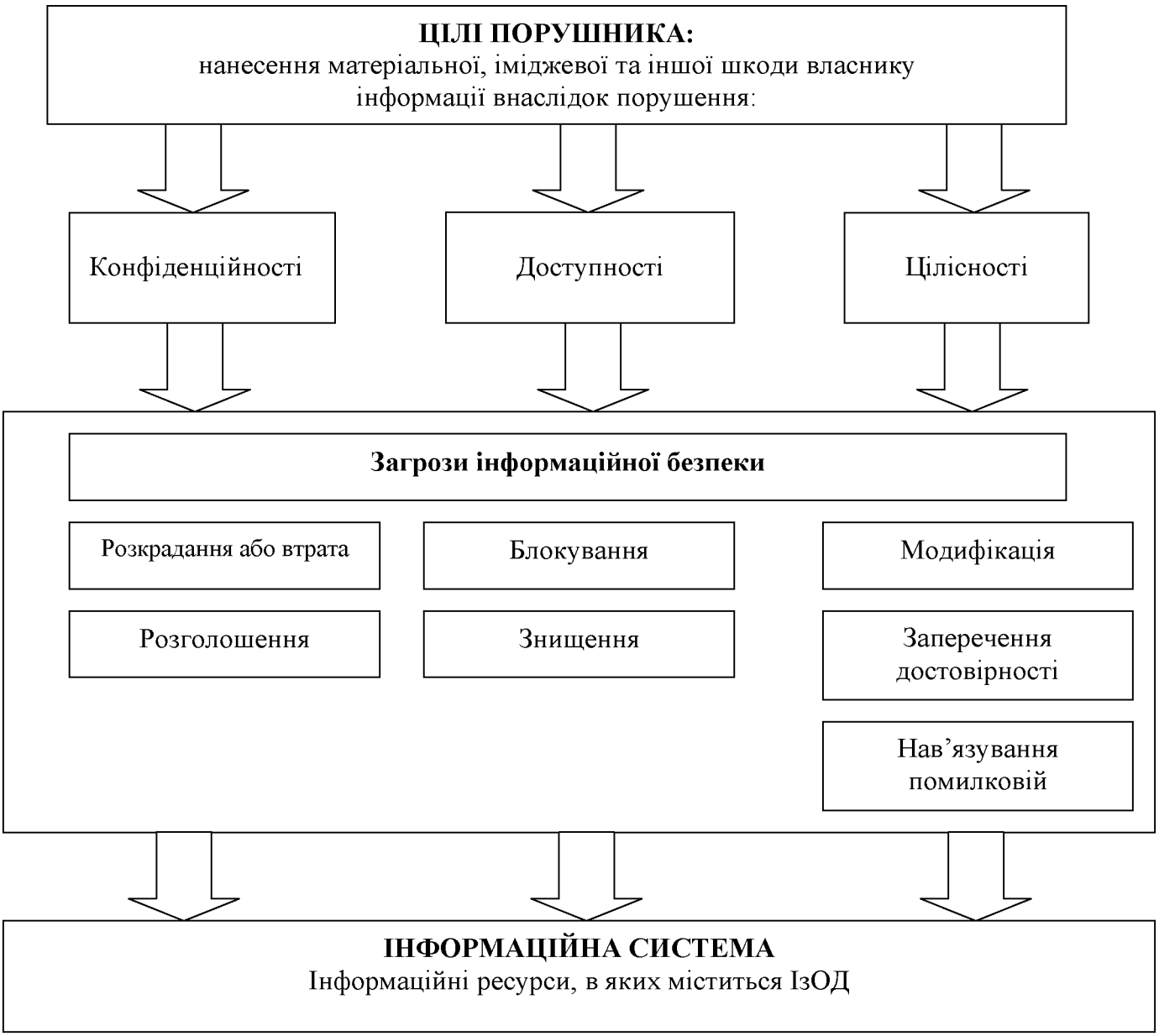

Рисунок 1.3 – Модель впливу порушника на інформаційну систему

Таблиця 1.2 – Класифікація порушників

Позначення	Визначення категорії	Рівень загрози
1	2	3
Внутрішні по відношенню до АС		
ПВ1	Технічний персонал, який обслуговує приміщення (електрики, сантехніки, прибиральниці тощо), в яких розташовані компоненти АС.	1
ПВ2	Персонал, який обслуговує технічні засоби.	2
Зовнішні по відношенню до АС		
ПЗ1	Будь які особи, що знаходяться за межами контрольованої зони.	1
ПЗ2	Відвідувачі(запрошені з деякого приводу)	2
ПЗ3	Конкуруючі організації	4
ПЗ4	Кримінальні структури	2
ПЗ5	Хакери	2
Специфікація за мотивами здійснення порушення		
Позначення	Мотив порушення	Рівень загрози
М1	Безвідповідальність	1
М2	Самозатвердження	2
М3	Корисливий інтерес	3
М4	Професійний обов'язок	4
Специфікація порушника за рівнем кваліфікації та обізнаності щодо АС		
Позначення	Основні кваліфікаційні ознаки порушника	Рівень загрози
К1	Знає особливості системи контролю доступом(СКД) на об’єкт	3
К2	Знає структуру, функції й механізми систем захисту	4

Продовження таблиці 1.2

Специфікація порушника за часом дії		
Позначення	Характеристика можливостей порушника	Рівень загрози
Ч1	В неробочий час, під час планових перерв у роботі і т. ін.	2
Ч2	Під час функціонування підприємства	3
Ч3	Як під час перерв, так і під час функціонування	4
Специфікація порушника за місцем дії		
Позначення	Характеристика можливостей порушника	Рівень загрози
Д1	Без доступу на контрольовану територію	1
Д2	З контрольованої території без доступу у приміщення	2
Д3	Усередині приміщень, але без доступу до технічних засобів	3
Д4	З робочих місць користувачів	4

Таблиця 1.3 – Модель порушника

Порушник / Специфікація	ПВ1	ПВ2	ПЗ2	ПЗ3
Мотив	М1	М1	М2,3	М3
Кваліфікація	К1	К2	К1	К1
Час дії	Ч2	Ч2	Ч3	Ч3
Місце дії	Д2	Д3,4	Д3,4	Д3,4

1.6 Умовна класифікація веб-додатків

В залежності від цілей створення веб-додатки можна розділити на кілька груп.

1 Сайт-візитка – невеликий сайт, який містить відомості про компанії, підприємця, державного службовця або приватну особу. Зазвичай в них міститься коротка інформація: напрямок діяльності, посада, інтерв'ю, невелике портфоліо, контактні дані та адреси. По суті, це докладна віртуальна візитна картка.

2 Персональний сайт – особистий проект, найчастіше створюється простим користувачем, який тільки починає займатися розробкою програмного забезпечення, або створюється веб-майстром для користувача, може містити абсолютно будь-яку інформацію та мати самий несподіваний дизайн. Деякі персональні сайти в процесі розробки перетворюються в професійні проекти.

3 Промо-сайт – це ресурс, що створюється для певного бренду, торгівельної марки або продукту, де розміщується повна інформація про продукцію, фотографії, а також повідомляється про спеціальні рекламні акції, знижки або конкурси.

4 Корпоративний сайт – створюється для компанії або державної установи, на якому розміщується вичерпна інформація про власника. Докладно розписуються всі послуги, корпоративні події, керівний склад. Найчастіше на такому ресурсі розміщуються додаткові сервіси – фотогалереї, стрічки новин, блоги і форуми. Деякі розділи сайту можуть бути закриті від неавторизованих користувачів, доступ до них отримують лише співробітники компанії.

5 Інформаційний сайт-каталог – це тематичний класифікатор Інтернет ресурсів. Об'ємний сайт, що має від декількох десятків до декількох мільйонів сторінок. У ньому розміщуються посилання на інші сайти з описами чи статтями, які розташовуються за категоріями, відповідним тематиці сайтів. Сайт призначається для полегшення користувачам пошуку потрібної

інформації. На такий сайт кожен власник може додати свій ресурс, важлива умова – сайт, що додається повинен відповідати вимогам каталогу.

6 Дошка оголошень – великий сайт, на якому користувачі можуть розміщувати найрізноманітніші оголошення – продаж, купівля, вакансії, знайомства та інше. Дошки оголошень бувають універсальні та вузької тематики, наприклад, тільки про нерухомість. Сайт, як правило, забезпечений багатофункціональним пошуком, який дозволяє робити точні вибірки з наявних оголошень.

7 Інтернет-магазин – цей сайт має, як правило, від сотні до декількох тисяч сторінок, на яких представлені товари. За аналогією зі звичайними магазинами такі ресурси поділяються на універсальні та тематичні (комп'ютерна техніка та аксесуари, мобільні телефони або одяг). Товари розміщуються по категоріях, їх можна сортувати за ціною або брендам, до кожного продукту, зазвичай, складають опис і додають фотографію. У такий сайт вбудовується інтерфейс для замовлення товарів та оплати.

8. Інформаційний портал – це багатофункціональний великий ресурс, основним завданням якого, є надання користувачеві вичерпної інформації за його інтересами. Найчастіше портал включає в себе декілька додаткових функцій – в ньому є каталог ресурсів, поштові сервіси, форуми, чати, віджети погоди та іншої корисної інформації. Портал сьогодні є найскладнішою віртуальною бізнес- системою і здатний вирішувати найважчі комерційні завдання. По суті, це сайт з багаторівневою структурою і значним обсягом інформації, що має велику аудиторію користувачів.

З точки зору вартості інформації, найбільш цінною системою є інформаційний портал та корпоративний сайт. Злом корпоративного сайту може призвести до неавторизованого доступу до самого «серця» компанії, що може нести величезні збитки для власників компанії і нечувану вигоду для кіберзлочинця. Саме тому веб-додатки повинні бути ефективно захищені, а можливість злому має бути якщо не виключена, то хоча б мінімізована.

1.7 Узагальнена класифікація вразливостей веб-додатків

Створення класифікації загроз безпеки веб-додатків є важливою для фахівців у галузі інформаційної безпеки, виробників програмних продуктів та інших сторін, які займаються безпекою в мережі Інтернет. На основі класифікації в подальшому можуть бути створені методики досліджень на предмет наявності вразливостей, рекомендації по розробці додатків з урахуванням загроз та вимог до продуктів. Схематично класифікація вразливостей зображена на рисунку 1.4.

- Класифікація вразливостей за рівнем в інфраструктурі АС:
    1 вразливості мережевого рівня (протоколи стека TCP/IP, протоколи NetBEUI, IPX/SPX);
    2 вразливості рівня операційної системи (вразливості Windows, Linux, MAC OS);
    3 вразливості рівня бази даних (вразливості конкретних СУБД – Oracle, MS SQL, Sybase);
    4 вразливості серверного рівня (вразливості WEB, SMTP, FTP, DNS серверів);
    5 вразливості прикладного рівня (рівень роботи програмного забезпечення веб-додатків – код PHP, ASP.Net, Java та інші).
- За ступенем ризику:
    1 високий. Вразливості, що дозволяють несанкціоновано отримати безпосередній доступ до веб-додатка з правами адміністратора (суперкористувача), або в обхід мережевих екранів, чи інших засобів захисту;
    2 середній. Вразливості, що дозволяють атакуючому отримати інформацію, яка з високою ймовірністю дозволить отримати несанкціонований доступ до інформації, що обробляється у веб-додатку;

3 низький. Вразливості, що дозволяють зловмисникові здійснювати несанкціонований збір критичної інформації про систему.

- За джерелом виникнення:
  1 технічні (вразливості програмного забезпечення);
  2 технологічні (помилки проектування веб-додатків);
  3 організаційні (помилки бізнес-логіки роботи ПЗ).
- За класом атак:
  1 на механізм автентифікації. Підбір паролю або логіну (Brute Force), недостатня автентифікація (Insufficient Authentication), небезпечне відновлення паролів (Weak Password Recovery Validation);
  2 на механізм авторизації. Передбачуване значення ідентифікатора сесії (Credential / Session Prediction), недостатня авторизація (Insufficient Authorization), відсутність таймаута сесії (Insufficient Session Expiration), Фіксація сесії (Session Fixation);
  3 на клієнтів. Підміна вмісту (Content Spoofing), міжсайтове виконання сценаріїв (Cross-site Scripting, XSS), розщеплення HTTP-запиту (HTTP Response Splitting);
  4 направлені на виконання зловмисного коду. Переповнення буфера (Buffer Overflow), атака на функції форматування рядків (Format String Attack), впровадження операторів LDAP (LDAP Injection), виконання команд ОС (OS Commanding), впровадження операторів SQL (SQL Injection), впровадження серверних розширень (SSI Injection), впровадження операторів XPath (XPath Injection);
  5 на розголошення інформації. Індексування директорій (Directory Indexing), ідентифікація додатків (WEB-Server/Application Fingerprinting), витік інформації (Information Leakage), зворотний шлях в директоріях (Path Traversal), передбачуване розташування ресурсів (Predictable Resource Location);

6 логічні. Зловживання функціональними можливостями (Abuse of Functionality), відмова в обслуговуванні (Denial of Service), недостатня протидія автоматизації (Insufficient Anti-automation).

1.8 Аналіз поширеності вразливостей веб-додатків

В розділі наведена статистика знайдених вразливостей за даними дослідницького центру Positive Research, це один з найбільших в Європі дослідницьких центрів в області інформаційної безпеки. З 2004 року при сприянні Positive Research лідери IT– галузі, серед яких Microsoft, Cisco, Google, Avaya, Citrix, VMware, Trend Micro, усунули кілька сотень вразливостей і недоліків систем безпеки.

Оцінка захищеності проводилась ручним способом за методами «чорного» і «білого» ящиків з використанням допоміжних автоматизованих засобів. Метод «чорного» ящика полягає в проведенні робіт з оцінки захищеності інформаційної системи без попереднього отримання будь-якої інформації про неї з боку власника. Метод «білого» ящика заключається в тому, що для оцінки захищеності інформаційної системи використовуються всі необхідні дані про неї, включаючи вихідний код веб-додатків.

Виявлені вразливості класифіковані згідно системі Web Application Security Consortium Threat Classification (WASC TC v. 2 [11]), за винятком вразливостей Improper Input Handling, Improper Output Handling і Denial of Service, оскільки вони реалізуються при експлуатації безлічі інших вразливостей. Проект WASC TC являє собою спробу класифікувати всі загрози безпеки веб-додатків. Члени Web Application Security Consortium створили його для розробки та популяризації стандартної термінології опису проблем безпеки веб-додатків. Цей документ дає можливість розробникам, фахівцям в області безпеки, виробникам програмних продуктів і аудиторам використовувати для взаємодій єдину мову.

Рисунок 1.4 - Узагальнена класифікація вразливостей

Найкращі умови для несанкціонованих проникнень в сегменті веб-додатків надають системи управління вмістом (CMS) – 18%. Хакери постійно шукають уразливості в CMS, і знаходять їх у великій кількості. Адміністраторам сайтів, побудованих на популярних платформах, необхідно не тільки контролювати активність підозрілу, але і оперативно встановлювати оновлення для CMS. Не слід забувати також про форуми, які знаходяться на другому місці за кількістю вразливостей (7%). Третє місце і 3% складають Інтернет-магазини, на останньому місці – фреймворки, інші не класифіковані веб-додатки складають 70%[12].

За результатами аналізу захищеності в 2010 – 2012 роках, всі досліджені ресурси містили хоча б одну уразливість. Десять вразливостей, виявлених на найбільшій кількості сайтів, представлені в таблиці 2. Лідер у цьому списку – Cross-Site Request Forgery, до якого схильні 61% всіх перевірених сайтів. Далі

знаходяться Information Leakage і Brute Force – 54% і 52% сайтів, а також SQL Injection з критичним рівнем ризику, виявлена в 47% ресурсів. На більш низьких позиціях в список входять ще дві вразливості з високим ступенем ризику – OS Commanding і Path Traversal (обидві з часткою вразливих сайтів в 28%). У список десяти найбільш поширених вразливостей увійшли також недоліки середнього рівня ризику: Insufficient Anti-automation (42% досліджених сайтів), Cross-Site Scripting (40%), Predictable Resource Location (36%) і Insufficient Transport Layer Protection (22%), що зображено в таблиці 1.4.

Таблиця 1.4 – Найбільш поширені вразливості ПЗ

Вразливість	Доля веб-сайтів, %
Міжсайтове виконання запитів (Cross-Site Request Forgery)	61
Витік інформації (Information Leakage)	54
Неспроможність протистояти атаці прямого перебору (Brute Force)	52
Впровадження SQL-операторів (SQL Injection)	47
Недостатня протидія автоматизації (Insufficient Anti-automation)	42
Міжсайтове виконання сценаріїв (Cross-Site Scripting)	40
Передбачуване розташування ресурсів (Predictable Resource Location)	36
Виконання команд операційної системи (OS Commanding)	28
Зворотний шлях в директоріях (Path Traversal)	28
Недостатній захист транспортного рівня (Insufficient Transport Layer Protection)	22

На рисунку 1.5 наведені дані по долям веб-сайтів, на яких виявлені вразливості різного рівня ризику. Визначено, що частка сайтів з вразливостями високого ступеня ризику в 2020 році нижче в порівнянні з 2019 роком, а середнього ступеня – представлені на рисунку 1.6.

Найбільш поширеними вразливостями високого ступеня ризику є SQL Injection, OS Commanding і Path Traversal, які зустрічаються практично в кожному третьому веб-додатку.

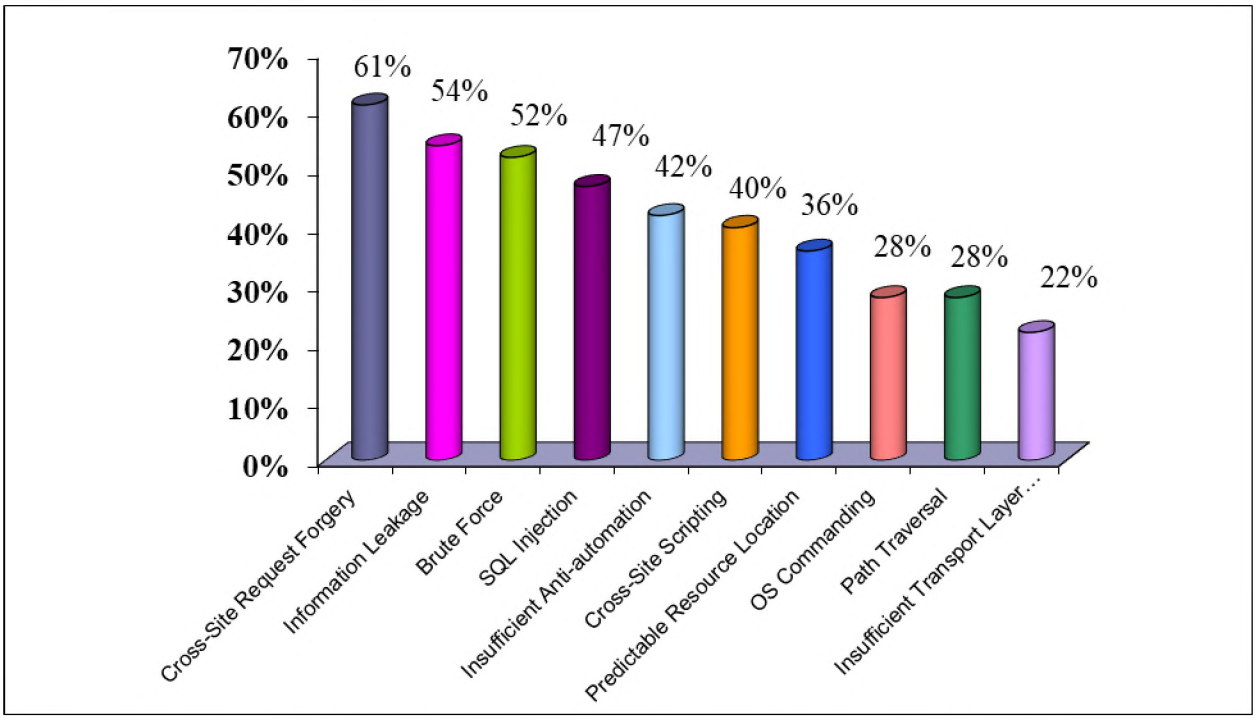

Рисунок 1.5 – Найбільш поширені вразливості веб-додатків

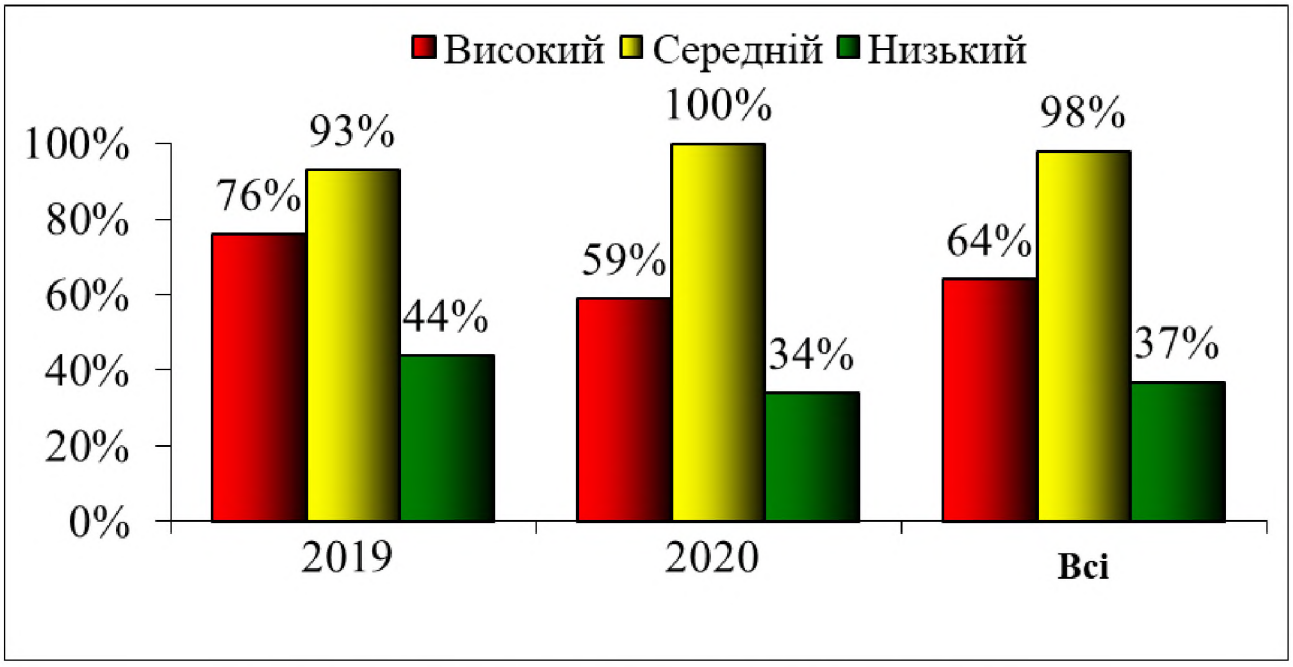

Рисунок 1.6 - Доля веб-сайтів з вразливостями

Серед протестованих ресурсів зустрічаються веб-додатки, написані на різних мовах програмування. При цьому для кожної мови характерний свій набір найбільш значущих вразливостей. Більшість розробників використовують мову програмування PHP: на ній написані 63% сайтів, що досліджувались. Значні частки ASP.NET (19%) і Java (14%). Решта мов програмування зустрічається набагато рідше. Розподіл сайтів – учасників тестування за мовою програмування візуально представлено на рисунку 1.7.

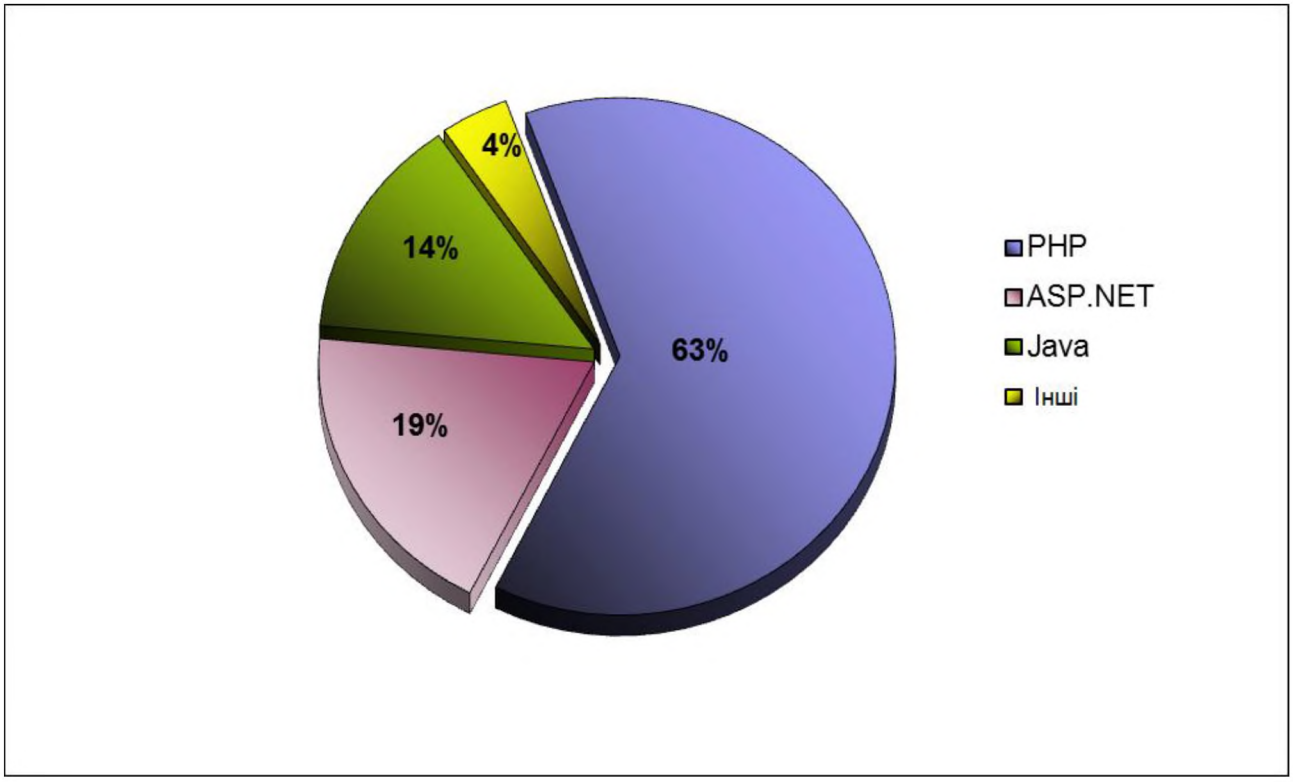

Рисунок 1.7 – Розподіл сайтів-учасників тестування за мовою програмування

1.9 Дослідження обраних вразливостей за методикою НД ТЗІ 2.7-009-09

Веб-додаток складається з веб-сторінок, тому доцільно використати нормативний документ системи технічного захисту інформації, що встановлює вимоги до технічних та організаційних заходів захисту інформації веб-сторінки в мережі Інтернет (НД ТЗІ 2.5-010-03 "Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу"). Цей документ призначений для

суб'єктів відносин (власників або розпорядників веб-сторінки, операторів (провайдерів), користувачів), діяльність яких пов'язана з розробкою та експлуатацією веб-сторінки, розробників комплексної системи захисту інформації та постачальників окремих її компонентів, а також для фізичних та юридичних осіб, які здійснюють оцінку захищеності веб-сторінки на відповідність вимогам ТЗІ.

З урахуванням особливостей надання доступу до інформації веб-сторінки, типових характеристик середовищ функціонування та особливостей технологічних процесів оброблення інформації, визначаються наступні мінімально необхідні рівні послуг безпеки для забезпечення захисту інформації від загроз:

– за умови, коли веб-сервер розміщується у оператора, а робочі станції – на території власника веб-сторінки, взаємодія яких з веб-сервером здійснюється з використанням мереж передачі даних (технологія Т2), мінімально необхідний функціональний профіль визначається: КА-2, КВ-1, ЦА-1, ЦО-1, ЦВ-1, ДВ-1, ДР-1, НР-2, НИ-2, НК-1, НО-1,НЦ-1, НТ-1, НВ-1.

Технологія Т1 різниться від технології Т2 способом передачі інформації від робочої станції до веб-сервера, а саме: наявністю у другому випадку незахищеного середовища, яке не контролюється, і додатковими вимогами щодо ідентифікації та автентифікації між КЗЗ робочої станції й КЗЗ веб-сервера під час спроби розпочати обмін інформацією та забезпечення цілісності інформації при обміні[13].

Згідно проведеного дослідження за методикою НД ТЗІ 2.7-009-09 "Методичні вказівки з оцінювання функціональних послуг безпеки в засобах захисту інформації від несанкціонованого доступу"[14], для більш детального аналізу та адаптації методів автоматизованого пошуку вразливостей обрано вразливості слабкості механізму обробки SQL-запитів (атака "SQL-ін'єкція") та слабкості механізму обробки введених користувачем даних (атака "міжсайтове виконання сценаріїв" - Cross-Site Scripting), а також недостатній захист

транспортного рівня (Insufficient Transport Layer Protection). Саме ці загрози найбільш актуальні для досліджування веб-додатків згідно списків міжнародних аудиторських компаній[15]. Вплив обраних вразливостей на критерії безпеки представлений в таблиці 1.5, де зазначено:

- КА - адміністративна конфіденційність;
- КВ - конфіденційність при обміні;
- ЦА - адміністративна цілісність;
- ЦВ - цілісність при обміні;
- ДР - використання ресурсів;
- НР - реєстрація;
- НИ - ідентифікація і автентифікація;
- НК - достовірний канал;
- НО - розподіл обов'язків;
- НЦ - цілісність комплексу засобів захисту.

Таблиця 1.5 - Вплив обраних вразливостей на критерії безпеки

Вразливість	Атака	Критерії безпеки
Слабкість механізму обробки SQL-запитів	SQL-ін'єкція	КА-2, ЦА-1, ЦВ-1, ДР-1, НР-2, НИ-2, НО-1, НЦ-1
Слабкість механізму обробки введених користувачем даних	Міжсайтове виконання сценаріїв (XSS)	КА-1, ЦВ-1, НИ-1,
Архітектурна слабкість (незахищеність) протоколу HTTP	Перехоплення мережевого трафіку	КА-1, КВ-1, НК-1

Функціональна послуга безпеки "Ідентифікація та автентифікація" рівня НИ-1 – "Зовнішня ідентифікація та автентифікація" має одну з обов'язкових вимог, що не виконується за наявності вразливостей "Слабкість механізму

обробки SQL-запитів" або "Слабкість механізму обробки введених користувачем даних". Так як перш, ніж дозволити будь-якому користувачу будь-якого типу виконувати будь-які інші, контрольовані КЗЗ ОЕ дії, КЗЗ ОЕ здатний з використанням засобів і захищених механізмів одержати від зовнішнього джерела автентифікований ідентифікатор цього користувача, але за наявності вразливості отримати ідентифікатор користувача неможливо.

Функціональна послуга безпеки "Адміністративна конфіденційність" рівня КА-2 – "Базова адміністративна конфіденційність" не виконується в разі наявності вразливостей, так як має залежність від послуги НИ-1.

Функціональна послуга безпеки "Адміністративна цілісність" рівня ЦА-1 – "Мінімальна адміністративна цілісність" не виконується в разі наявності вразливостей, так як має залежність від послуги НИ-1.

Функціональна послуга безпеки "Розмежування обов'язків" рівня НО-1 – "Виділення адміністратора" не виконується в разі наявності вразливостей, так як має залежність від послуги НИ-1.

Функціональна послуга безпеки "Реєстрація" рівня НР-1 – "Зовнішній аналіз" не виконується в разі наявності вразливостей, так як має залежність від послуги НИ-1.

Функціональна послуга безпеки "Цілісність комплексу засобів захисту" рівня НЦ-1 – "КЗЗ з контролем цілісності" не виконується в разі наявності вразливостей, так як має залежність від послуг НР1 та НО-1.

Функціональна послуга безпеки "Використання ресурсів" рівня ДР-1 – "Квоти" не виконується в разі наявності вразливостей, так як має залежність від послуги НО-1.

Функціональна послуга безпеки "Достовірний канал" рівня НК-1 – "Однонаправлений достовірний канал" не виконується при використанні незахищеного протоколу HTTP. Так як у складі функціональних модулів ОЕ, що входять до складу КЗЗ, не реалізовані засоби створення достовірного каналу, використовуваного для початкової ідентифікації та автентифікації.

Функціональна послуга безпеки “Конфіденційність при обміні” рівня КВ-1 – “Мінімальна конфіденційність при обміні” має одну з обов’язкових вимог, що не виконується при використанні незахищеного протоколу HTTP. Так як КЗЗ ОЕ не забезпечує захист від безпосереднього ознайомлення з інформацією, що міститься в переданих об'єктах різного типу.

Функціональна послуга безпеки “Цілісність при обміні” рівня ЦВ-1 – “Мінімальна цілісність при обміні” має одну з обов’язкових вимог, що не виконується за наявності вразливостей “Слабкість механізму обробки SQL-запитів” або “Слабкість механізму обробки введених користувачем даних”. Так як КЗЗ ОЕ не забезпечує можливість виявлення порушення цілісності інформації, що міститься в переданих об'єктах різного типу.

1.10 Дослідження сканерів безпеки веб-додатків

Найголовніша причина, за якою команди розробників ПЗ прагнуть до автоматизації, полягає в тому, що ручне тестування забирає дуже багато часу. У міру зростання веб-додатка час на його повне тестування також зростає, в залежності від складності програми. Крім того, існує велика кількість різних версій браузерів, які активно розвиваються і стають перешкодою для підтримки кросбраузерності веб-додатків.

Автоматизація тестів – основна практика гнучкої методології розробки ПЗ (Agile, Scrum). Вона дозволяє розробникам часто писати високоякісний код, забезпечує каркас, завдяки якому команда нарощує швидкість при дотриманні високих стандартів якості. Система керування версіями вихідного коду, автоматизовані збірки і комплекти тестів, розгортання, моніторинг і широке розмаїття сценаріїв та інструментів позбавляють від рутини, гарантують надійність і дозволяють команді постійно виконувати свою роботу найкращим чином [16].

Отже, на сьогоднішній день актуальним завданням є пошук методів автоматизації тестування веб-додатків з метою спрощення ручних перевірок та зменшення матеріальних витрат на розробку програмного забезпечення.

Автоматизація пошуку вразливостей досить складне завдання, особливо, якщо намагатися створити універсальний метод пошуку вразливостей, тому що необхідно “навчити комп'ютер” бути хакером, і створити програму так, щоб вона могла знаходити вразливості. Комп'ютеру дуже складно аналізувати не структуровані дані, а веб-сторінки можна вважати не структурованими. Так, є стандарти, яких повинні дотримуватися розробники при створенні веб-ресурсів, але на практиці ці стандарти досить часто ігноруються. Існує безліч технологій створення сайтів і програмісти можуть використовувати будь-які інструменти.

Для пошуку та аналізу вразливостей безпеки потрібні глибокі знання та досвід. Забезпечення безпеки сайту – це комплексна міра. Безпека багато в чому залежить від вибору програмного забезпечення веб-сервера, хостинг-провайдера та якості програмування самого веб-додатку.

На поточний момент існує не велика кількість професійних сканерів вразливостей, серед яких можна виділити найбільш якісні, такі як: Nessus, XSpider, Acunetix, IBM Internet Scanner, Retina, Shadow Security Scanner, N–Stealth. Але всі вони мають свої особливості та недоліки, серед яких найсуттєвішими є:

– надмірна витрата часу на аналіз захищеності (від 10 хвилин до декількох діб);

– висока ціна (сканери коштують від 600 до 5000 доларів);

– існуючі сканери не здатні підтримувати механізми двокрокової (двохфакторної) авторизації в веб-додатку та проводити сканування «з середини»;

– ймовірний великий поріг помилкового спрацювання та невиявлених вразливостей (наявні помилки першого та другого роду);

– недоступність алгоритмів та методів, за якими працює сканер (закритий програмний код).

1.11 Висновки

Дослідження веб-додатків свідчить про велику кількість вразливостей та загроз для банківського програмного забезпечення, тому для перевірки наявності вразливостей та оцінки захищеності необхідна велика кількість часу. Постає питання ефективного розподілу ресурсів, яке вирішується впровадженням автоматизації тестування вразливостей. Але існуючі інструменти мають суттєві недоліки, а саме: вони мають велику ціну; недостатньо ефективні; мають закриті алгоритми аналізу (недоступність програмного коду); не здатні підтримувати механізми двокрокової автентифікації; мають ймовірний поріг помилкового спрацювання.

Саме тому в магістерській дипломній роботі потрібно вирішити наступні задачі:

– розробити методику комплексної оцінки захищеності веб-додатків;

– проаналізувати сучасні методи пошуку вразливостей;

– згідно частини методики розробити підсистему пошуку вразливостей веб-додатків;

– довести ефективність розробленої підсистеми.

Прийнято рішення розробити і впровадити простий, гнучкий до змін, сканер вразливостей веб-додатків, що буде мати можливість авторизації та проводити сканування “з середини” веб-додатку. Отже, вимоги до створення програмного засобу сканування вразливостей з метою підвищення ефективності дослідження та економії ресурсів:

– підсистема має не допускати помилок першого та другого роду;

– час на тестування автоматизованим шляхом має бути не більшим за час, при аналізі ручним способом;

– підсистема має дозволяти використовувати всі існуючі механізми автентифікації у веб-додатку;

– підсистема має бути модульною, розширюваною, гнучкою до додавання нових методів пошуку та детектування вразливостей;

– робота підсистеми повинна бути наглядною.

РОЗДІЛ 2. СПЕЦІАЛЬНА ЧАСТИНА

Дослідження методів пошуку вразливостей та методики, що запропонована НД ТЗІ 2.7.-009-09 “Методичні вказівки з оцінювання функціональних послуг безпеки в засобах захисту інформації від несанкціонованого доступу”, вказують на те, що для створення ефективної підсистеми пошуку вразливостей необхідно об’єднати рекомендації методики з методами отримання ідентифікуючої інформації про веб-додаток та методами моделювання атак на веб-додатки.

2.1 Аналіз методів пошуку вразливостей веб-додатків

Одним із завдань забезпечення безпеки веб-додатків є виявлення вразливостей веб-додатків з метою подальшого їх усунення. В роботі розглянуті сучасні методи виявлення вразливостей в веб-додатках і проведено їх аналіз.

В залежності від використаних технологій при розробці програмного забезпечення, або використаної мови програмування, наявність певних вразливостей може істотно розрізняються, обмежуючи застосування тих чи інших методів аналізу.

Згідно досліджень OWASP [15] найбільш ефективним способом виявлення вразливостей веб-додатків є експертний аналіз вихідного коду додатка (англ. “code review”). Але цей спосіб досить трудомісткий, вимагає високої кваліфікації експерта і не захищений від помилок самого експерта. Тому активно розвиваються методи автоматичного виявлення (автоматизованого сканування) вразливостей веб-додатків.

Існуючі методи можливо класифікувати за принципом пошуку на ручні та автоматизовані, за знаннями про систему на принцип “білого ящика”, принцип “чорного ящика” та принцип “сірого ящика”. Принцип “білого ящика” (англ. “white-box”) – передбачає передачу виконавцю всього програмного забезпечення з його подальшим розгортанням у виконавця, що виконує роботу

з його аналізу, або організацію аналогічної копії додатку у власній інформаційній системі з наданням виконавцю повного доступу до цього ресурсу. У даному випадку мається можливість відстежити, яким чином додаток реагує на будь-який переданий до нього запит. Це найбільш продуктивний метод проведення аналізу захищеності веб-додатків, що дозволяє виявити максимальну кількість вразливостей. Проте варто зауважити, що даний метод позбавлений можливості поглянути на додаток з позицій атакуючого.

Принцип “сірого ящика” (англ. “gray-box”). Проведення робіт з наданням всієї необхідної інформації про програму, окрім забезпечення безпосереднього доступу до самого веб-серверу, на якому функціонує досліджуваний веб-додаток. Зазвичай виконавцю надаються наступні дані: структура каталогів додатка, дані для авторизованого підключення у веб-додатку (наприклад, ім'я користувача, пароль і набір одноразових паролів для проведення транзакцій), вихідний код деяких файлів або функцій.

Принцип “чорного ящика” (англ. “black-box”). Проведення робіт з оцінки захищеності програми без попереднього отримання будь-якої інформації про неї. Використовується, коли необхідно оцінити захищеність з позицій зловмисника, що зазвичай володіє мінімальними знаннями про досліджувану систему. В основному подібні оцінки здійснюються в рамках «тестування на проникнення» (англ. “penetration testing”). Всі дослідження можуть проходити як з попередженням обслуговуючого персоналу про плановані роботах, так і без нього. У другому випадку існує можливість оцінити, за який час після початку дослідження персонал зафіксує інцидент, а також яка адекватність зроблених дій з мінімізації його впливу або запобігання.

2.1.1 Метод отримання ідентифікуючої інформації про веб-додаток

Метод отримання ідентифікуючої інформації про веб-додаток може бути використаним за принципом “білого” або “сірого” ящика та ґрунтується на аналізі виконаних наборів HTTP-запитів до веб-додатка, відповіді яких

дозволять зробити висновок про технології, за допомогою яких розроблено веб-додаток, які версії програмного забезпечення використовуються та інше.

Для визначення типу і версії веб-сервера використовується техніка “fingerprint” [17], яка заснована на тому, що кожен веб-сервер по-своєму обробляє HTTP-протокол, в результаті чого можна з високим ступенем імовірності визначити тип і навіть версію веб-сервера шляхом відправлення серверу набору певних запитів і аналізу відповідних їм відповідей.

Для визначення решти параметрів використовується аналіз HTML-сторінок засобами пошуку регулярних виразів. Шаблони для пошуку задаються експертом. Наприклад, використання розширення aspx в URL може свідчити про те, що веб-додаток було розроблено з використанням технології “Microsoft .Net”. Можливість отримання ідентифікуючої інформації користувачем додатка є потенційною вразливістю, так як у мережі Інтернет накопичені величезні обсяги даних вразливостей програмних продуктів із зазначенням версій програм, методів реалізації атак, бюлетнів безпеки. Ця інформація призначена для адміністраторів і фахівців у галузі інформаційної безпеки, в той же час вона доступна через мережу Інтернет всім бажаючим.

Метод отримання ідентифікуючої інформації про веб-додаток і виявлення його вразливостей за допомогою відкритих баз даних вразливостей отримав широке практичне застосування через свою простоту і доступність, але сам метод не дозволяє знайти нові вразливості.

2.1.2 Метод тестування на проникнення

Метод тестування на проникнення (англ. penetration testing) розглядає веб-додаток з точки зору зовнішнього користувача, тобто потенційного зловмисника. При цьому вважається, що зловмисник володіє такими ж можливостями, як і звичайний користувач, тобто не має доступу до вихідних кодів веб-додатка, конфігураційним налаштуванням, тобто проводиться за принципом “чорного ящика”. Метод передбачає моделювання проведення атак та тестування веб-додатка шляхом відправлення запитів, які імітують

активність користувача, включаючи, в тому числі, і некоректні запити, що відповідають діям зловмисника.

При пошуку вразливостей в веб-додатку методом тестування на проникнення виникають такі основні задачі: отримання та аналіз структури веб-додатка; побудова набору тестових HTTP-запитів та їх "прогін" з аналізом відповідей веб-додатка для виявлення вразливостей.

Перевіряються типові вразливості, пов'язані з неправильним налаштуванням веб-додатка та веб-сервера. Виконується перевірка можливості автоматичної побудови індексу каталогу, виконання HTTP методів PUT і DELETE та інше.

При виконанні тестування основна проблема полягає у визначенні критеріїв наявності вразливості. На сьогоднішній день для вирішення цього завдання застосовується метод, в якому розпізнавання відповідей сервера здійснюється регулярними виразами, що задаються експертами. Таким чином, даний метод також вимагає детального налаштування експертом, при цьому метод працює в рамках протоколу HTTP і не залежить від технології, за допомогою якої розроблено веб-додаток.

Даний метод дозволяє накопичувати знання експерта у вигляді набору правил побудови запитів і набору шаблонів для аналізу HTTP-відповідей. Цей метод отримав широке поширення для виявлення вразливостей, коли необхідно оцінити наявність хоча б типових помилок при розробці та налаштуванні веб-додатків. Перевагою даного методу є те, що він дозволяє оцінювати розгорнутий і налаштований веб-додаток, тобто "бойовий варіант" працюючого програмного забезпечення,виявляючи не тільки помилки в програмуванні, але й помилки конфігурацій.

2.1.3 Метод статичного аналізу програмного коду

Метод статичного аналізу програмного коду веб-додатків виконується за принципом "білого" ящика та не потребує розгорнутого веб-додатка. Замість

цього проводиться побудова графів управління і залежностей по даним, а виявлення вразливостей за допомогою аналізу цих графів.

Для виявлення вразливостей використовується два основних підходи: аналіз типів безпеки і аналіз потоків даних [18,19]. У кожному з підходів вразливість визначається, як порушення в програмі властивості “невтручання” [20].

При аналізі типів безпеки створюється набір типів безпеки (t1, t2, ..., tn), над яким вводиться відношення часткового порядку. Кожній змінній програми ставиться в відповідність її тип безпеки. Існують два способи визначення типів безпеки всіх змінних – ручний і автоматичний. Ручний спосіб передбачає, що при кожному оголошенні змінної програміст повинен явно задати її тип безпеки. Автоматичний спосіб передбачає, що дано: розмітка з типами безпеки змінних, що містить вхідні дані, розмітка функцій, що здійснюють введення вхідних даних користувача, правила виведення типів безпеки ще нерозмічених змінних. Таким чином, при аналізі програми послідовно від початку до кінця всі нерозмічені змінні отримують свій тип безпеки автоматично. Тип безпеки змінної постійний на весь час її існування.

Для виявлення вразливостей потрібно визначити необхідні рівні безпеки параметрів функцій. У реалізаціях методу така специфікація часто знаходиться в окремому конфігураційному файлі і не залежить від конкретної програми.

Аналіз типів безпеки має істотний недолік – постійну прив'язку типу безпеки до змінної. В результаті тип безпеки визначається без урахування того, з якого джерела дані потрапили в змінну, що призводить до великого числа помилок першого роду. Для того щоб уникнути великої кількості помилок першого роду, було запропоновано прив'язувати тип безпеки не до змінної, а до її значення. У результаті кожна змінна теоретично може отримувати будь-який тип безпеки протягом свого існування. Цей підхід отримав назву аналізу потоків даних[21].

У рамках аналізу потоків даних кожної конструкції мови програмування зіставляються формальні правила виводу результуючого типу безпеки змінних, що беруть участь в цій конструкції. А для бібліотечних функцій створюється розмітка, що описує типи безпеки параметрів і результату функцій. На основі розмітки і формальних правил виводу, аналізатор визначає в кожній ділянці коду тип безпеки конкретних даних.

Виявлення вразливостей веб-додатків методом статичного аналізу передбачає вирішення таких основних завдань:

– визначення конструкцій мови і бібліотечних функцій, які повертають дані, потенційно контрольовані зловмисником. Наприклад, це параметри HTTP запитів POST та GET, cookie-файли користувача. Вся інформація, отримана з таких конструкцій, позначається певною міткою;

– розробка аксіом поширення мітки між змінними програми. Дані аксіоми повинні підтримувати не тільки тривіальні присвоювання, але й більш складні конструкції мови, такі, наприклад, як присвоювання через масиви, роботу через посилання та інше. Дана система аксіом повинна містити правила, за якими інформація, позначена міткою, може знову стати безпечною. Повнота і точність методу виявлення вразливостей переважно залежить від якості моделі поширення мітки за додатком;

– визначення конструкцій мови, які не повинні приймати дані з типом безпеки мітки. У цей список повинні входити функції для роботи з системним оточенням, з СУБД, з поштовими сервісами і т.д.

Метод статичного аналізу програмного коду веб-додатків дозволяє виявляти вразливості, пов'язані з нестійкістю веб-додатків до некоректних вхідних даних. Інші класи вразливостей даний метод виявляти не дозволяє. Метод специфічний для кожної технології створення веб-додатків і вимагає від розробника анотування функцій перевірки коректності вхідних даних.

2.1.4 Метод динамічного аналізу програмного коду

Метод динамічного аналізу програмного коду веб-додатків за своєю суттю та принципом дії аналогічний методу статичного аналізу за винятком того, що аналіз проводиться в процесі роботи веб-додатка без побудови графів програми. Замість цього в процесі виконання програми для кожної змінної виробляється обчислення типу безпеки, а для кожної розміченої функції порівнюються типи безпеки параметрів зі специфікацією, зазначеної в розмітці.

Метод динамічного аналізу дозволяє здійснювати виявлення вразливостей для динамічних мов програмування, як, наприклад, PHP, Python, Perl, Ruby. На сьогоднішній день для різних мов програмування метод динамічного аналізу часто інтегрується в інтерпретатор мови, наприклад, PHPrevent для мови PHP.

Основною проблемою методу динамічного аналізу є проблема зняття міток “tainted”. Існує кілька способів вирішення: додаткова розмітка програмного коду, застосування стандартних «чистячих» бібліотек функцій зі складу технології, за допомогою якої розроблено веб-додаток.

Метод динамічного аналізу, на відміну від статичної аналізу, дозволяє здійснювати виявлення вразливостей в генерованому на льоту коді. У той же час, метод динамічного аналізу аналізує лише один з можливих графів виконання програми та не дозволяє зробити висновок про відсутність вразливостей у всій програмі. Метод динамічного аналізу, аналогічно статичному аналізу, специфічний для кожної технології створення веб-додатків[22].

2.2 Розробка комплексної методики оцінки захищеності

В роботі пропонується застосувати наступний підхід. Розділити об’єкт дослідження (веб-додаток) умовно на 3 частини:

– сукупність документації, бізнес-схем, технічних завдань, специфікацій ПЗ, користувацьких інструкцій;

– серверна частина (Back-end), системне ПЗ операційної системи, веб-сервер, SQL– сервер та інші;

– інтерфейсна (Ftront– end) та прикладна частина, тобто ПЗ самого сайту, наприклад: клієнт-банкінг, блог, форум, система управління контентом (CMS).

Тестування послуг безпеки потрібно починати проводити якомога раніше, на самому початковому етапі розробки веб-додатків. Робота тестувальників, та фахівців з безпеки повинна починатись з аналізу вимог до ПЗ, перевірки технічного завдання на однозначність, протиріччя, повноту, коректність. Також аналізуються фактори, що можуть вплинути або завадити заявленій роботі веб-додатка.

Для перевірки серверної частини пропонується використання автоматизованих сканерів безпеки та утиліт, таких як Nesus, Xspider, Wikto, Metasploit Framework та інших. А також поєднати методи тестування, що базуються на знаннях про отриману систему, її програмного коду, та методи, що дозволяють взаємодіяти з системою тільки через інтерфейсний рівень. Проаналізувати ПЗ, що використовується сервером за допомогою Nmap. Перевірити модуль адміністрування сервера (Cpanel, WebMin, ISPmanager) та наявність вразливостей в базах даних, таких як OSVDB (The Open Source Vulnerability Database), NVD (National Vulnerability Database) і корпорації MITRE.

Необхідно визначити схильність веб-додатка до DDoS-атак, що спрямовані

на вразливість в коді ПЗ. Для цього виконується навантажливе тестування серверу за допомогою спеціальних інструментів, таких як Apache Jmeter або HP Load Runner. Слід зауважити, що для DDoS-атак, спрямованих на перевантаження пропускного каналу сервера, не існує жодного ефективного засобу захисту. Прикладом може слугувати атака на ПАТ «Укртелеком» у 2010

році та нездатність найбільшої телекомунікаційної компанії захистити своїх користувачів [23].

Для перевірки прикладної частини пропонується поєднання тестування “білого” та “чорного” ящиків, використати дві команди дослідників. Перша повинна займатись тестуванням на проникнення, а друга – перевіряти програмний код та аналізувати систему з середини. Потрібно скласти список перевірок, що буде базуватись на класифікації уразливостей згідно системи Web Application Security Consortium Threat Classification (WASC TC v. 2 [11]), за винятком вразливостей Improper Input Handling, Improper Output Handling, оскільки вони реалізуються при експлуатації інших вразливостей. Проект WASC являє собою спробу класифікувати всі загрози безпеки веб-додатків. Члени цього консорціуму створили його для розробки та популяризації стандартної термінології опису проблем безпеки веб-додатків. Цей документ дає можливість розробникам, фахівцям в області безпеки, виробникам програмних продуктів і аудиторам використовувати для взаємодій єдину мову.

При дослідженні використовуються утиліти та методи тестування ПЗ, які пропонує організація, орієнтована на підвищення безпеки програмного забезпечення The Open Web Application Security Project (OWASP) [24].

Однією з вимог міжнародних платіжних систем VISA та MasterCard до банківських продуктів може слугувати дотримання стандарту безпеки PA-DSS та PCI-DSS v 2.0 [25,26]. В цьому випадку спеціалісти з інформаційної безпеки, окрім списку власних вимог та перевірок, мусять доповнити методику згідно вимог стандарту, та провести відповідне тестування, що пропонується стандартом безпеки платіжних карток.

Розроблена методика оцінки захищеності веб-додатків містить багато перевірок та включає в себе тестування веб-додатків на велику кількість вразливостей, а також включає в себе різні методи їх пошуку. Тому, в рамках дипломної роботи буде вирішена задача автоматизації пошуку певних видів вразливостей з використанням декількох методів тестування.

2.3 Моделювання атаки міжсайтового виконання сценаріїв (XSS)

XSS (англ. “Cross Site Scripting” – “міжсайтове виконання сценаріїв”) - тип вразливості інтерактивних інформаційних систем у веб-додатках. Прийнято вживати термін “XSS”, щоб не було плутанини з каскадним таблицями стилів, що використовують скорочення “CSS”.

XSS виникає, коли в згенеруються сервером сторінку з якоїсь причини потрапляють скрипти користувача. Специфіка подібних атак полягає в тому, що замість безпосередньої атаки сервера вони використовують вразливий сервер в якості засобу атаки на клієнта.

Атака, за принципом дії ділиться на “Stored XSS” та “Reflected XSS”. Reflected XSS – скрипт не зберігається на сервері вразливого сайту, або він не може автоматично виконатися в браузері клієнта. Для виконання даного типу атаки потрібна додаткова дія, яку повинен виконати браузер клієнта (наприклад, клік за спеціально сформованої посиланням).

Stored XSS - шкідливий скрипт зберігається на сервері, і спрацьовує в браузері при відкритті зараженої сторінки сайту, не вимагає додаткових дій і є непомітним для клієнта.

Суть вразливості в відсутності перевірки вхідних даних, які розраховані на введення користувачем, що призводить до можливості довільного виконання коду з використанням мови JavaScript. JavaScript - об'єктно-орієнтована мова програмування. Є діалектом мови ECMAScript. Зазвичай використовується як вбудовувана мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності веб-сторінкам.

Найпростішим способом виявлення даної вразливості є передача в точки входу інформації веб-додатку конструкції виду: “><script> Alert(); </script>, де теги <script></script> вказують браузеру на виконання JavaScript коду, розташованого між цими тегами, функція Alert() виводить повідомлення “OK”. Таким чином, якщо виконається хоча б даний елементарний код на мові

JavaScript, то веб-сторінка, що входить до складу веб-додатка, є потенційно вразливою і фактично дозволяє виконати будь-який довільний код на мові JavaScript.

Найбільш поширеною метою експлуатації зловмисником вразливості XSS є крадіжка cookie-файлів.

Cookie-файли (куки) – невеликий фрагмент даних, створений веб-сервером або веб-додатком, що зберігається на комп'ютері користувача у вигляді файлу, який веб-клієнт (звичайно веб-браузер) щоразу пересилає веб-серверу в HTTP-запиті при спробі відкрити сторінку відповідного сайту. Застосовується для збереження даних на стороні користувача, на практиці зазвичай використовується для:

- автентифікації користувача;
- зберігання персональних налаштувань користувача;
- відстеження стану сесії доступу користувача;
- ведення статистики про користувачів.

Як і атака типу SQL-ін'єкція, XSS також заснована на відсутності фільтрації даних веб-додатку. Відповідно захист від подібного роду атак повинен будуватися із застосування фільтруючої обробки всіх вхідних даних.

Існують такі основні способи фільтрації даних:

- обробка всіх HTML тегів і заміна їх аналогами у вхідних параметрах;
- обробка всіх користувальницьких URI;
- прогнозування дій, які слід очікувати від користувача.

Обробку всіх HTML тегів і заміна їх нешкідливими аналогами у вхідних параметрах здійснює, наприклад, PHP-функція htmlentities(). Існує майже аналогічна функція HTMLSpecialChars (строка), яка є більш обмеженою і її застосування не рекомендується.

Якщо дозволити користувачам вказувати їх власні URI (наприклад, завантажувати зображення), то необхідно бути впевненим, що вони не можуть використовувати URI, «заражений» JavaScript кодом. PHP-функція parse_url()

здійснює розбиття URI в асоціативний масив по частинах. Використовуючи дану функцію можна легко перевірити URI (наприклад, допустимим може бути http, ftp і неприпустимим JavaScript. Іншим способом застосування даної функції є перевірка довірених доменів, якщо необхідно обмежити можливість вказувати будь-які URI.

Як правило, з поточного запиту можна визначити обмежену кількість дій, яку користувач може здійснити на наступному етапі. Наприклад, користувач може редагувати документ, тоді очікуваними діями можуть бути або збереження результату, або скасування всіх занесених змін. Така система прогнозування дій користувача може допомогти при виявленні та запобіганні атак типу XSS. Дана система може бути побудована, наприклад, за наступним принципом: для кожного запиту згенерувати всі можливі URI, які очікуються від наступної дії користувача у вигляді строки або хешу у сесії. Далі порівняти наступний запит користувача зі значеннями в згенерованому масиві, при цьому, якщо немає збігів, необхідно використовувати деяку логіку (повністю залежить від структури та логіки самого додатка), яка зможе визначити, чи є запитуваний URI безпечним, можливо користувач просто перейшов з іншої частини сайту і не збирається проводити зловмисних дій.

Дані рішення у забезпеченні протидії атакам XSS дозволяють на досить ефективному рівні побудувати рубіж захисту програми від спроб зловмисника порушити логіку програми або вкрасти персональні дані.

2.4 Моделювання атаки впровадження коду SQL-ін'єкція

SQL-ін'єкція – один з поширених способів злому веб-додатків, що працюють з базами даних, заснований на впровадженні в запит довільного SQL-коду. Впровадження SQL, залежно від типу використовуваної СУБД (Системи управління базою даних) і умов впровадження, може дати можливість атакуючому виконати довільний запит до бази даних (наприклад, прочитати вміст будь-яких таблиць, видалити, змінити або додати нові дані), отримати можливість читання та / або запису локальних файлів і виконання довільних

команд на веб-сервері. Атака типу SQL-ін'єкція можлива за некоректної обробки вхідних даних, що використовуються в SQL-запитах та надходять від клієнтів при заповненні веб-форм.

За відсутності фільтрації переданих даних зловмисник може провести цю атаку та виявити вразливість, просто передавши параметр з лапками, тим самим порушив логіку SQL-запиту, і, якщо включено відображення помилок, на сторінці з'явиться повідомлення про помилку.

Наприклад в СУБД MySQL є спеціальний символ, що позначає коментар – це «-» який може використовуватися безпосередньо в SQL-запиті, таким чином наявність вразливості SQL-ін'єкція дозволяє зловмисникові впроваджувати свій довільний SQL запит, змінюючи оригінальний запит і отримувати дані з бази даних. Отже, зловмисник може потенційно отримати ідентифікаційні дані всіх користувачів, зареєстрованих в базі даних.

Коли зловмисник намагається виконати атаку SQL-ін'єкція, сервер повертає повідомлення про помилки, що синтаксис SQL-запиту є некоректним. Але існує випадок, коли при спробі провести SQL-ін'єкцію, зловмисник не отримує повідомлення про помилки, що досить сильно ускладнює проведення даної атаки, проте це не робить її проведення неможливим. Такий вид атаки називається Blind
SQL-ін'єкції (Сліпа SQL-ін'єкція). У даному випадку зловмисник може спробувати провести ряд SQL-запитів, заснованих на логічних умовах і також несанкціоновано отримати інформацію з бази даних MySQL.

У цьому випадку проведення такого роду атаки зводиться до маніпуляції логікою SQL-запитів, тобто передачу вразливому скрипту коректних і некоректних логічних конструкцій для отримання даних з таблиць MySQL.

Захист від вразливостей такого виду ґрунтується на фільтрації вхідних параметрів, а саме:

- фільтрація строкових параметрів;
- фільтрація числових параметрів.

Для фільтрації вхідних строкових параметрів необхідно, наприклад, застосувати PHP функцію mysql_real_escape_string(). Дана функція здійснює екранування спеціальних символів (x00, \n, \r, \, ', " та \x1a.) в рядку даних, екранування полягає в додавання символу "зворотного слеша" – "\". Таким чином, екрануючи спеціальні символи, тобто змінюючи їх спеціальне значення, дана функція забезпечує безпеку строкових даних.

Крім строкових параметрів веб-додатку можуть передаватися числові дані і, як правило, це цілі числа. Для захисту від порушення логіки SQL-запиту і проведення SQL-атаки через числовий параметр необхідно використовувати, наприклад, PHP-функцію INTVAL(). Сенс використання даної функції полягає в тому, що вона повертає ціле число з вхідної змінної і в тому випадку, якщо зловмисник спробує порушити логіку SQL-запиту, вводячи спеціальні символи або інші дані, дана функція все одно буде повертати ціле число і логіка запиту не порушиться.

Багато серверів баз даних підтримують можливість відправки параметризованих запитів (підготовлені вирази). При цьому параметри зовнішнього походження відправляються на сервер окремо від самого запиту або автоматично екрануються клієнтської бібліотекою. У ядро PHP, починаючи з версії 5.1, включено спеціальне розширення – PDO. PHP Data Objects (PDO) – розширення для PHP, що надає розробнику простий і універсальний інтерфейс для доступу до різних баз даних. PDO пропонує єдині методи для роботи з різними базами даних, хоча текст запитів може трохи відрізнятися. Так як багато СУБД реалізують свій діалект SQL, який в тій чи іншій мірі підтримує стандарти ANSI та ISO, то при використанні простих запитів можна домогтися сумісності між різними мовами. На практиці це означає, що можна досить легко перейти на іншу СУБД, при цьому не змінюючи або незначно змінюючи код програми [24].

PDO не використовує абстрактних шарів для підключення до БД, на зразок ODBC, а використовує для різних БД їх «рідні» драйвери, що дозволяє

добитися високої продуктивності. В даний час для PDO існують драйвери практично до всіх загальновідомим СУБД і інтерфейсів. Втім, є і драйвер для підключення до ODBC.

Параметризовані запити мають дві основні переваги:

– SQL-запит повинен бути оброблений (або підготовлений) тільки один раз, при цьому він може бути виконаний з тими ж або іншими параметрами. Коли запит підготовлено, СУБД буде аналізувати, узагальнювати і оптимізувати виконання запиту. Для складних запитів цей процес може бути дуже відчутний, особливо, якщо запит необхідно виконувати кілька разів з різними параметрами. Використання параметризованих запитів дозволяє уникнути необхідність виконання цих операцій для кожного запиту, тим самим збільшується продуктивність і швидкість виконання запитів.

– Параметри, які віддаються в параметризованих запитах, не потрібно обрамляти лапками. Драйвер автоматично виконує це. Якщо запити у веб-додатку будуються на основі параметризованих запитах, то розробник може мати більшу впевненість, що SQL-ін'єкцію провести буде неможливо (проте, якщо у веб-додатку є місця, де параметризрвані запити не використовуються або запити не фільтруються, то проведення SQL-ін'єкції залишається потенційно можливим).

2.5 Розробка підсистеми автоматизованого пошуку вразливостей

2.5.1 Використані технології

Розроблена підсистема автоматизованого пошуку вразливостей складається з двох частин, а саме модуля для збору інформації про сервер, що використовує метод отримання ідентифікуючої інформації про веб-додаток, та модуля пошуку вразливостей у веб-додатку, що використовує метод моделювання атак.

Програма написана з використанням сучасної мови програмування C# на платформі “Microsoft .Net 4.0” та призначена для роботи в усіх операційних

системах сімейства Windows розрядності x32 та x64, за умови попереднього встановлення фреймворку “Dot NET 4.0”.

В систему інтегрована бібліотека Nmap[27]. Це безкоштовне ПЗ, призначене для різноманітного сканування IP-мереж, визначення стану об'єктів мережі (портів і відповідних їм служб). Nmap використовує велику кількість різних методів сканування, таких як UDP, TCP (connect), TCP SYN (напіввідкрите), FTP-proxy (прорив через ftp), Reverse-ident, ICMP (ping), FIN, ACK, Xmas tree, SYN-сканування. Підтримує великий набір додаткових можливостей, а саме: визначення операційної системи віддаленого хоста з використанням відбитків стека TCP / IP, «непомітне» сканування, динамічне обчислення часу затримки і повтор передачі пакетів, паралельне сканування, визначення неактивних хостів методом паралельного ping-опитування, сканування з використанням підробних хостів, визначення наявності пакетних фільтрів, пряме (без використання portmapper) RPC-сканування, та інші види.

Nmap працює за допомогою командного рядка, але такий спосіб роботи не є зручним для спеціаліста. Тому було прийнято рішення розробити графічний інтерфейс для використання цього потужного інструмента, а також інтегрувати дане програмне забезпечення в підсистему пошуку вразливостей.

При розробці модуля пошуку вразливостей було використано бібліотеки Selenium (це проект, в рамках якого розробляється серія програмних продуктів з відкритим вихідним кодом (Open Source), призначених для автоматизованого функціонального тестування веб-додатків.

В підсистему пошуку вразливостей інтегровано Selenium WebDriver – це програмна бібліотека для управління браузерами. Часто вживається так само більш коротку назву WebDriver або “драйвер браузера”. Вона включає в себе сімейство драйверів для різних браузерів, а також набір бібліотек для різних мов програмування, що дозволяють виконувати керування браузером [28].

У рамках проекту Selenium підтримуються браузери Firefox, Internet Explorer і Safari, а також драйвери для мобільних браузерів для Android і iOS.

Драйвер для браузера Google Chrome розвивається в рамках проекту “Chrome”, а драйвер для браузера Opera (включаючи мобільні версії) розробляється компанією Opera Software. Тому вони формально не є частиною проекту Selenium, а поширюються і підтримуються незалежно.

Отже для використання опції “Пошук вразливостей веб-додатку” необхідно додатково встановити браузери, в яких буде проводитись тестування, або використовувати встановлений за замовчуванням в операційній системі Windows - Microsoft Internet Explorer.

Для виконання серії випробувань та апробації розробленої підсистеми пошуку вразливостей був розроблений веб-сайт, що містив досліджувані вразливості та дозволяв довести ефективність розробленого рішення.

2.5.2 Функції підсистеми пошуку вразливостей

При старті підсистема дозволяє обрати два режима роботи. Відповідно, перший режим роботи призначений для ідентифікації серверного програмного забезпечення, та використовує метод отримання ідентифікуючої інформації про веб-додаток та носить інформаційний характер, для спеціаліста, що його використовує. Другий режим роботи призначений для тестування сторінок веб-додатка на вразливості та використовує методи моделювання атак міжсайтового виконання сценаріїв (XSS), впровадження коду SQL (SQL-ін’єкція). Також підсистема дозволяє перевірити, чи використовує веб-додаток захищений канал передачі інформації між сервером та браузером клієнта (чи використовується безпечне з’єднання HTTPS). Головний інтерфейс підсистеми зображено на рисунку 2.1.

При виборі користувачем режиму “Сканування сервера та мережі” відкривається нове вікно та активується модуль аналізу сервера. Для того, щоб розпочати сканування необхідно ввести доменне ім’я або IP-адресу веб-додатка. Наступним кроком необхідно визначити, яке саме тестування потрібно провести, для цього необхідно обрати профіль роботи або ввести ключі сканування для Nmap самостійно та натиснути кнопку “Розпочати сканування”.

В залежності від обраних параметрів час тестування буде змінним та становитиме від 5 хвилин до декількох годин. По завершенню аналізу буде відображено спливаюче вікно “Сканування успішно завершено”. Результати сканування будуть відображені в інтерфейсі програми. В разі, якщо їх необхідно зберегти для подальшого аналізу або для друку, необхідно використати функцію “Зберегти результати” та записати їх в файл.

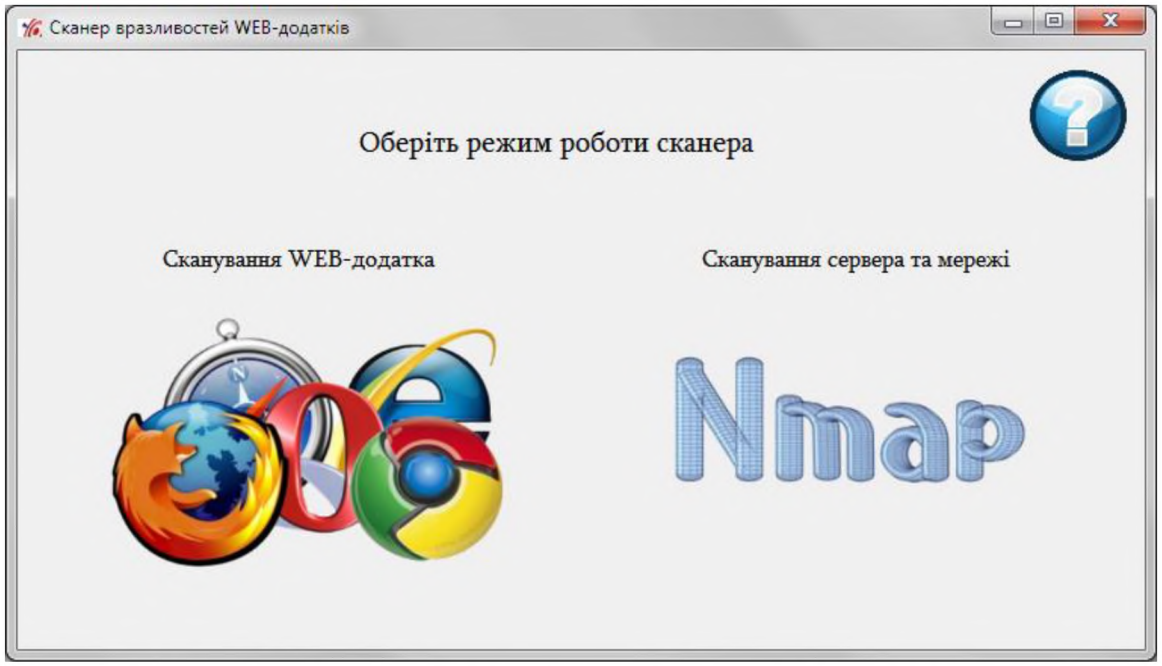

Рисунок 2.1 – Головний інтерфейс підсистеми пошуку вразливостей

Графічний інтерфейс підсистеми та модуля збору інформації про сервер представлено на рисунку 2.2.

Для пошуку вразливостей необхідно використати режим роботи – “сканування веб-додатка”, що зображений на рисунку 2.3. Наступним кроком необхідно визначити браузер в якому буде проведено тестування з наведеного переліку:

- Mozilla Firefox;
- Google Chrome;
- Opera Software ASA;
- Apple Safari;

– Microsoft Internet Explorer.

Після чого необхідно вказати URL-адресу веб-сторінки, сканування якої потрібно провести. В разі необхідності, можна використати функцію “автентифікація” для проведення сканування веб-додатка “зсередини”.

Для виконання поставлених вимог був розроблений модуль, що здатен підтримувати всі існуючі механізми автентифікації веб-додатків в полу-автоматичному режимі. В разі, якщо перед скануванням внутрішньої сторінки веб-додатку потрібно пройти автентифікацію, то в інтерфейсі необхідно заповнити відповідні поля “Логін” та “Пароль” користувача, або вказати допоміжний URL для базової автентифікації за принципом “login:password@URL”. Щоб ідентифікувати відповідні поля на веб-сторінці

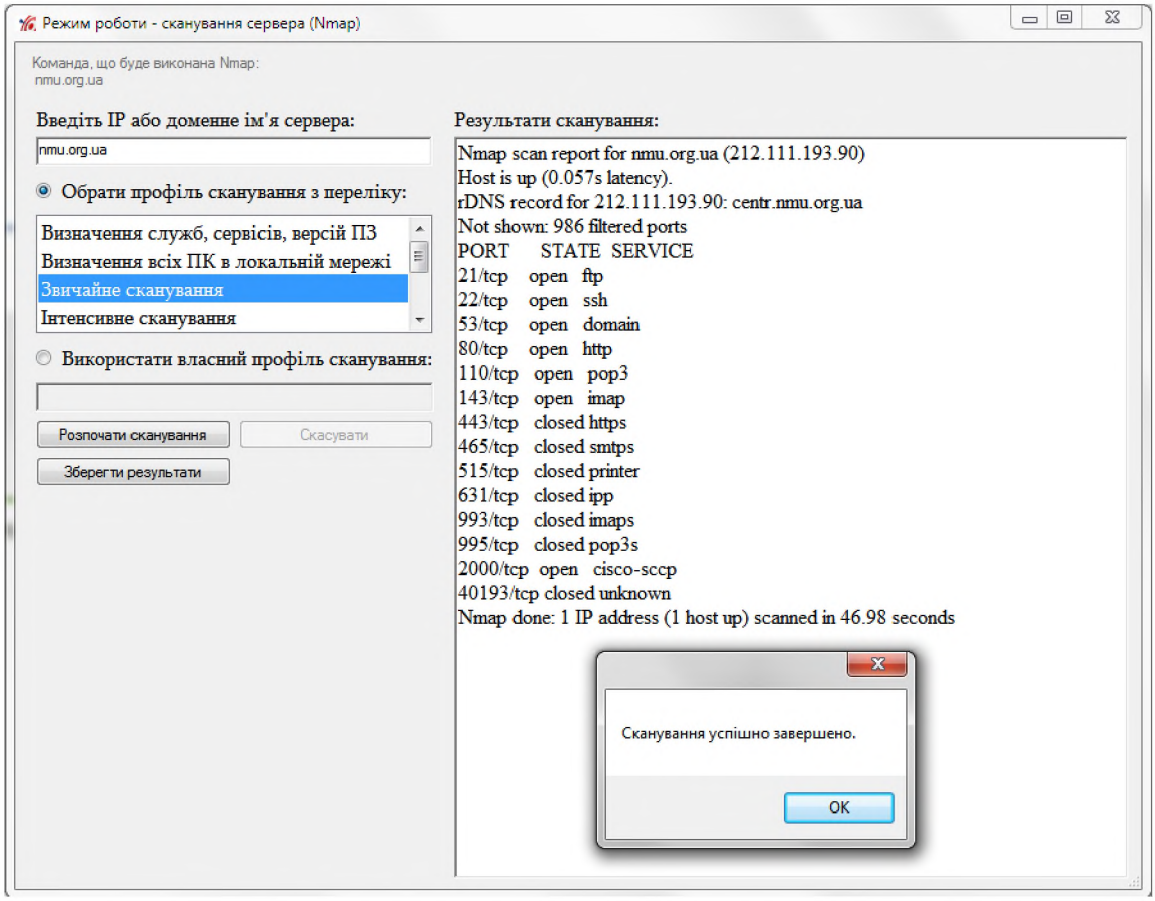

Рисунок 2.2 – Розроблений графічний інтерфейс

спеціаліст, використовуючи мову Xpath, вказує положення веб-елементів сторінки.

Якщо веб-додаток використовує інші технології автентифікації, такі як, наприклад, вхід по QR-коду, то для цього необхідно використати функцію “технологічна пауза” і виконати цю дію в ручному режимі, після чого підсистема автоматично розпочне сканування.

Використання технології керування браузером дозволяє підтримувати всі існуючі на сьогоднішній день механізми автентифікації, завдяки використання полуавтоматичного режиму введення даних.

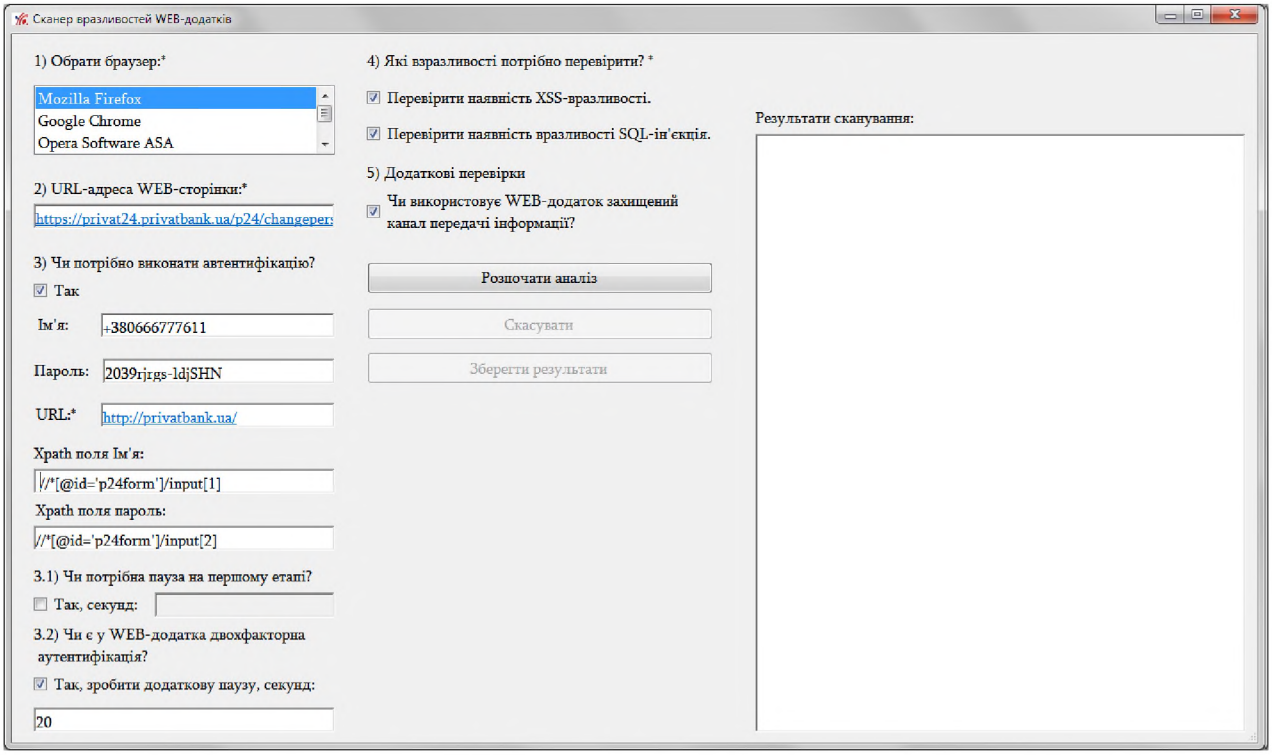

Рисунок 2.3 – Режим роботи “сканування веб-додатка”

Наступним кроком необхідно вказати, які типи вразливостей потрібно перевірити на вказаній веб-сторінці, відповідно відмітивши опції напроти назв вразливостей. Крім того, підсистема дозволяє перевірити, чи використовується веб-додатком захищене з’єднання при обміні інформацією між клієнтом та веб-сервером.

Після заповнення всіх необхідних даних можна розпочати сканування, яке розпочнеться з відкриття браузера, проходження механізму автентифікації, переходу на цільову сторінку. Далі виконаються набори тестів на проникнення

для вказаних вразливостей, в результаті роботи програми з'явиться повідомлення про успішність перевірки веб-сторінки. В разі, якщо буде знайдена вразливість, інформація про це відобразиться в інтерфейсі програми, а світлофор прийме червоний колір, разом із тим за допомогою Xpath буде вказано, де саме на сторінці знайдена вразливість (зображено на рисунку 2.4). Якщо ж жодної вразливості на вказаній сторінці не буде знайдено, то програма повідомить про це текстовим повідомленням та зеленим кольором світлофору (зображено на рисунку 2.5.).

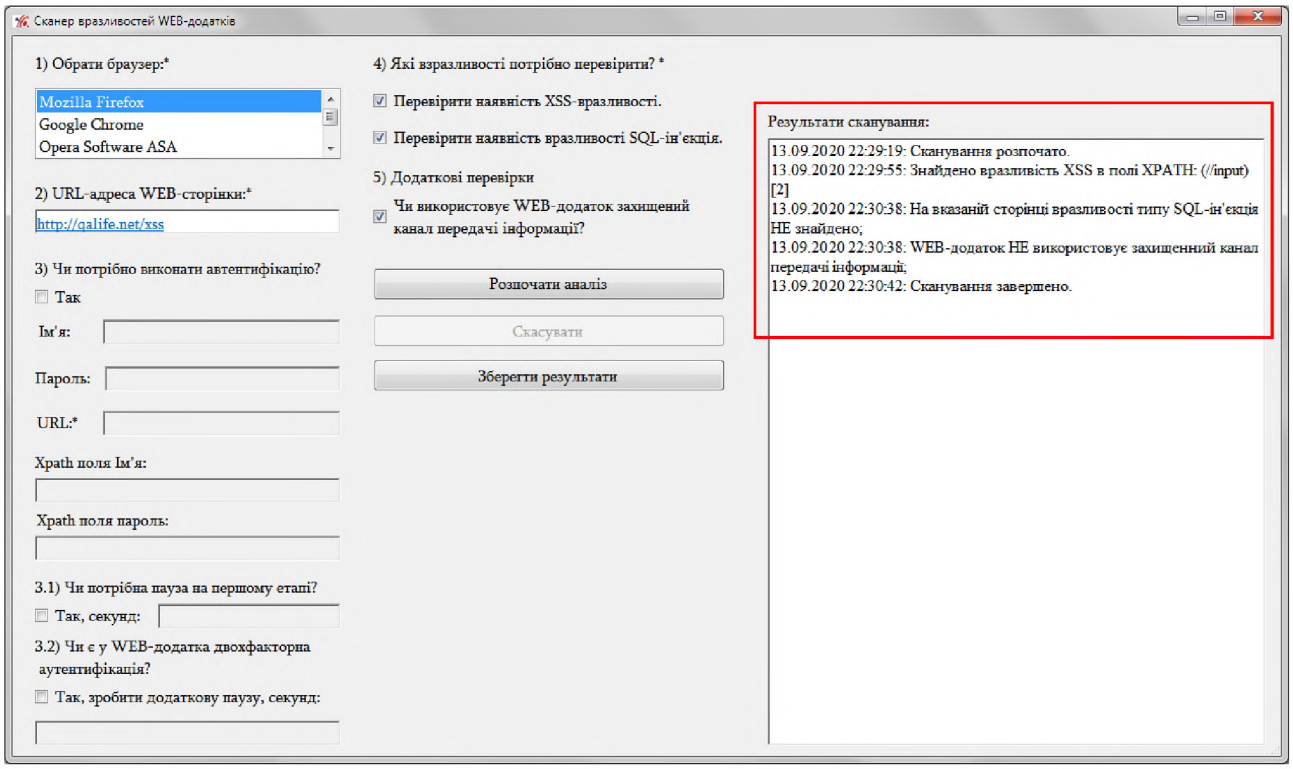

Рисунок 2.4 – Результати виявлення вразливостей

Після проведення сканування можна зберегти результати, використавши функцію "Зберегти результати" та виконати їх подальший аналіз з метою усунення недоліків.

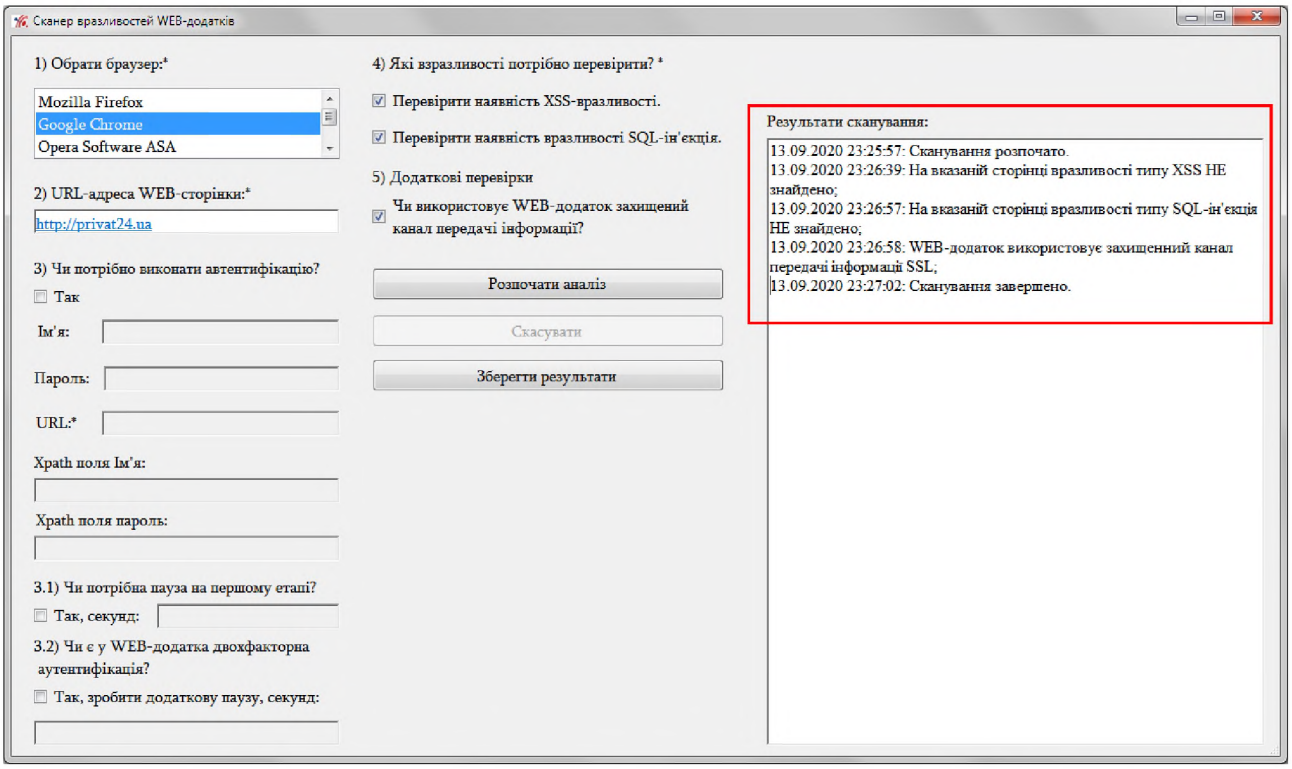

Рисунок 2.5 – Результати сканування (відсутність вразливостей)

2.6 Висновки

В результаті проведених досліджень було виконано поставлені задачі роботи:

– розроблена методику комплексної оцінки захищеності веб-додатків;

– проаналізовані сучасні методи пошуку вразливостей;

– згідно частини методики розроблена підсистему пошуку вразливостей веб-додатків;

– доведена ефективність розробленої підсистеми завдяки використання методу “моделювання атак”. Для перевірки наявності вразливостей використовувались змодельовані атаки, в разі якщо атака виконується успішно, робиться висновок про наявність вразливості, в іншому випадку результати свідчать про відсутність вразливостей.

В порівнянні з існуючими сканерами безпеки розроблена підсистема має ряд переваг та повністю реалізує поставлені до системи вимоги. А саме, результати випробувань свідчать про те, що вдалося повністю виключити

помилки першого роду, та мінімізувати прояв помилок другого роду. Підсистема здатна підтримувати всі існуючі механізми автентифікації у веб-додатках, на відміну від інших існуючих рішень. Сканування виконується наглядно, в кожний момент часу спеціаліст може контролювати процес та бачити, що саме перевіряється.

РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ

Мета цього розділу – обґрунтування економічної доцільності розробки методики виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку. Для вирішення цього завдання необхідно здійснити розрахунок капітальних витрат; експлуатаційних витрат, які визначають величину витрат на обслуговування системи на рік; величину можливого збитку; величину економічного ефекту; показники економічної ефективності.

3.1 Розрахунок (фіксованих) капітальних витрат

Капітальні (фіксовані) витрати є сумою витрат щодо створення і придбання основних фондів і нематеріальних активів, що підлягають амортизації.

Капітальні (фіксовані) витрати на проектування та впровадження проектного варіанта системи інформаційної безпеки складають:

$$К = К_{пр} + К_{зпз} + К_{пз} + К_{аз} + К_{навч} + К_{н}$$

де $К_{пр}$ – вартість розробки проекту інформаційної безпеки та залучення для цього зовнішніх консультантів;

$К_{зпз}$ – вартість закупівель ліцензійного основного й додаткового програмного забезпечення (ПЗ);

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

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

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

$К_{н}$ – витрати на встановлення обладнання та налагодження системи інформаційної безпеки.

3.1.1. Визначення витрат на підвищення рівня інформаційної безпеки підприємства шляхом розробки методики виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку

3.1.1.1 Визначення трудомісткості розробки методики виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку

Трудомісткість розробки визначається тривалістю кожної робочої операції:

$$t = t_{тз} + t_в + t_а + t_р + t_д, \quad \text{годин},$$

де $t_{тз}$ – тривалість складання технічного завдання на розробку методики виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку, $t_{тз}=112$;

$t_в$ – тривалість аналізу існуючих інформаційних потоків організації, вивчення ТЗ, літературних джерел за темою тощо, $t_в=80$;

$t_а$ – тривалість аналізу існуючих загроз безпеки інформації, $t_а=96$;

$t_р$ – тривалість розробки методики виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку, $t_м=240$;

$t_д$ – тривалість підготовки технічної документації, $t_д=40$.

Отже,

$$t = 112+80+96+240+40 = 568 \text{ годин}.$$

3.1.1.2. Розрахунок витрат на підвищення рівня інформаційної безпеки підприємства шляхом розробки методики виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку

Витрати на розробку заходів безпеки Кпз складаються з витрат на заробітну плату спеціаліста з інформаційної безпеки *Ззп* і вартості витрат машинного часу *Змч*:

$$К_{пз} = З_{зп} + З_{мч} = 97680 + 7480,56 = 105160,6 \text{ грн.}$$

$$З_{зп} = t\, З_{пр} = 568*346 = 196528 \text{ грн.}$$

де t – загальна тривалість операцій, годин;

Зпр – средньогодинна заробітна плата спеціаліста с інформаційної безпеки з нарахуваннями, грн/годину.

Вартість машинного часу для налагодження програми на ПК визначається за формулою:

$$З_{мч} = t \cdot С_{мч} = 296 \cdot 13,17 = 7480,56 \text{ грн.}$$

де t – трудомісткість операцій із побудови ефективної системи доступу персоналу, годин;

$С_{мч}$ – вартість 1 години машинного часу ПК, грн./година.

Вартість 1 години машинного часу ПК визначається за формулою:

$$С_{мч} = 0,8 \cdot 8 \cdot 1,55 + \frac{8400 \cdot 0,5}{1920} + \frac{5100 \cdot 0,4}{1920} = 13,17 \text{ грн.}$$

Розробка методики оцінки захищеності веб-додатків містить багато перевірок та включає в себе тестування веб-додатків на велику кількість вразливостей, а також включає в себе різні методи їх пошуку. При цьому планується придбання програмного забезпечення Oracle MySQL, вартістю 306887 грн., а також розробка сайта, вартістю 15000 грн.

Планується використання інтегрованої бібліотеки Nmap, яка є безкоштовним ПЗ, призначеним для різноманітного сканування IP-мереж, визначення стану об'єктів мережі (портів і відповідних їм служб).

Використання опції “Пошук вразливостей веб-додатку” можливо здійснити за допомогою використання браузерів, в яких буде проводитись

тестування, або використовувати встановлений за замовчуванням в операційній системі Windows - Microsoft Internet Explorer.

Для застосування методики із виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку необхідно здійснити навчання відповідних спеціалістів вартістю 15000 грн. ($К_н$=15000 грн.).

Таким чином, капітальні (фіксовані) витрати на підвищення рівня інформаційної безпеки підприємства шляхом розробки методики із виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку складуть:

$$К = 105160,6 + 306887 + 15000 = 427048 \text{ грн.}$$

3.1.1 Розрахунок поточних витрат

Річні поточні витрати на функціонування системи інформаційної безпеки складають:

$$С = С_в + С_к + С_{ак}, \quad \text{грн.}$$

де $С_в$ - вартість відновлення й модернізації системи;

$С_к$ - витрати на керування системою в цілому;

$С_{ак}$ - витрати, викликані активністю користувачів системи інформаційної безпеки).

При застосуванні методики із виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку, вартість модернізації системи складає 12000 грн. щорічно.

Витрати на керування системою інформаційної безпеки ($С_к$) складають:

$$С_к = С_н + С_а + С_з + С_{ел} + С_о + С_{тос}, \text{ грн.}$$

Річний фонд амортизаційних відрахувань ($С_а$) визначається прямолінійним методом, виходячи з вартості активів та строку їх корисного використання. Вартість Oracle MySQL складає 306887 грн.. Строк корисного використання – 4 роки. Таким чином,

$$С_а=306887/4=76721,75 \text{ грн.}$$

Відповідно до специфіки застосуванні методики із виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку спеціалістам з інформаційної безпеки необхідно проходити періодичне навчання протягом року, сумарна вартість якого складає 45000 грн. на рік.

Річний фонд заробітної плати інженерно-технічного персоналу, що обслуговує систему інформаційної безпеки ($С_з$), складає:

$$С_з = З_{осн} + З_{дод}, \text{ грн.}$$

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

Основна заробітна плата одного спеціаліста з інформаційної безпеки на місяць складає 22000 грн. Додаткова заробітна плата – 8% від основної заробітної плати. Отже,

$$С_з = 22000*12 + 22000*12*0,09= 287760 \text{ грн.}$$

Ставка ЄСВ для всіх категорій платників з 01.09.2020 р. складає 22%.

$$С_{єв} = 259200*0,22 = 63307,2 \text{ грн.}$$

Вартість електроенергії, що споживається апаратурою системою інформаційної безпеки протягом року ($С_{ел}$), визначається за формулою:

$$С_{ел} = Р \cdot F_р \cdot Ц_е, \text{ грн.},$$

де Р – встановлена потужність апаратури інформаційної безпеки, (Р=8,2 кВт);

$F_р$ – річний фонд робочого часу системи інформаційної безпеки ($F_р$ = 1920 год.);

$Ц_е$ – тариф на електроенергію, ($Ц_е$ = 1,55 грн./кВт за годину).

Вартість електроенергії, що споживається апаратурою системою інформаційної безпеки протягом року:

$$С_{ел} = 8,2*1920*1,55 = 24403,2 \text{ грн.}$$

Витрати на технічне й організаційне адміністрування та сервіс системи інформаційної безпеки визначаються у відсотках від вартості капітальних витрат - 2% (Стос = 427048 *0,02=8540,96 грн).

Витрати на керування системою інформаційної безпеки ($С_к$) визначаються:

$$С_к = 76721,75 +287760+63307,2 +24403,2 + 8540,96 =$$
$$= 460733,1 \text{ грн.}$$

Таким чином, річні поточні витрати на функціонування системи інформаційної безпеки складають:

$$С = 12000+ 460733,1 = 472733,1 \text{ грн.}$$

3.2 Оцінка можливого збитку

3.2.1 Оцінка величини збитку

Для розрахунку вартості такого збитку можна застосувати наступну спрощену модель оцінки.

Необхідні *вихідні дані* для розрахунку:

$t_п$ – час простою вузла або сегмента корпоративної мережі внаслідок атаки, 6 години;

$t_в$ – час відновлення після атаки персоналом, що обслуговує корпоративну мережу, 4 години;

$t_{ви}$ – час повторного введення загубленої інформації співробітниками атакованого вузла або сегмента корпоративної мережі, 7 годин;

$З_о$ – заробітна плата обслуговуючого персоналу (адміністраторів та ін.), 16000 грн./міс.;

$З_с$ – заробітна плата співробітників атакованого вузла або сегмента корпоративної мережі, 17000 грн./міс.;

$Ч_о$ – чисельність обслуговуючого персоналу (адміністраторів та ін.), 8 осіб.;

$Ч_с$ – чисельність співробітників атакованого вузла або сегмента корпоративної мережі, 40 осіб.;

О – обсяг прибутку атакованого вузла або сегмента корпоративної мережі, 1200 тис. грн. у рік;

$П_{зч}$ – вартість заміни встаткування або запасних частин, грн;

I – число атакованих сегментів корпоративної мережі, 5;

N – середнє число атак на рік, 60.

Упущена вигода від простою атакованого сегмента корпоративної мережі становить:

$$U = П_п + П_в + V,$$

де $П_п$ – оплачувані втрати робочого часу та простої співробітників атакованого вузла або сегмента корпоративної мережі, грн;

$П_в$ – вартість відновлення працездатності вузла або сегмента корпоративної мережі (переустановлення системи, зміна конфігурації та ін.), грн;

V – втрати від зниження обсягу продажів за час простою атакованого вузла або сегмента корпоративної мережі, грн.

Втрати від зниження продуктивності співробітників атакованого вузла або сегмента корпоративної мережі являють собою втрати їхньої заробітної плати (оплата непродуктивної праці) за час простою внаслідок атаки:

$$П_П = \frac{\sum Зс}{F} \cdot t_n = \frac{17000 \cdot 40}{176} \cdot 6 = 23181{,}81 \quad \text{грн},$$

де F – місячний фонд робочого часу (при 40-а годинному робочому тижні становить 176 ч).

Витрати на відновлення працездатності вузла або сегмента корпоративної мережі включають кілька складових:

$$П_в = П_{ви} + П_{пв} + П_{зч},$$

де $П_{ви}$ – витрати на повторне уведення інформації, грн.;

$П_{пв}$ – витрати на відновлення вузла або сегмента корпоративної мережі, грн;

$П_{зч}$ – вартість заміни устаткування або запасних частин, грн.

Витрати на повторне введення інформації $П_{ви}$ розраховуються виходячи з розміру заробітної плати співробітників атакованого вузла або сегмента корпоративної мережі $З_с$, які зайняті повторним введенням втраченої інформації, з урахуванням необхідного для цього часу $t_{ви}$:

$$П_{ви} = \frac{\sum Зс}{F} \cdot t_{ви} = \frac{17000 \cdot 40}{176} \cdot 7 = 27045{,}45 \text{ грн.}$$

Витрати на відновлення сегмента корпоративної мережі $П_{пв}$ визначаються часом відновлення після атаки $t_в$ і розміром середньогодинної заробітної плати обслуговуючого персоналу (адміністраторів):

$$П_{пв} = \frac{\sum Зо}{F} \cdot t_в = \frac{16000 \cdot 8}{176} \cdot 4 = 2909{,}09 \text{ грн.}$$

Таким чином, витрати на відновлення працездатності вузла або сегмента корпоративної мережі складають:

$$П_в = 27045{,}45+2909{,}09 = 29954{,}54 \text{ грн.}$$

Втрати від зниження очікуваного обсягу прибутків за час простою атакованого вузла або сегмента корпоративної мережі визначаються виходячи із средньогодинного обсягу прибутку і сумарного часу простою сегмента корпоративної мережі:

$$V = \frac{O}{Fг} \cdot (t_П + t_В + t_{ВИ})$$

$$V = \frac{1200000}{2080} \cdot (6 + 4 + 7) = 9807{,}69 \text{ грн.}$$

де $F_г$ – річний фонд часу роботи філії (52 робочих тижні, 5-ти денний робочий тиждень, 8-ми годинний робочий день) становить близько 2080 год.

$$U = 23181{,}81 + 29954{,}54 + 9807{,}69 = 62944 \text{ грн.}$$

Таким чином, загальний збиток від атаки на сегмент корпоративної мережі організації складе:

$$B = \sum_i \sum_n U = \sum_5 \sum_{60} 62944 = 18883200 \text{ грн.}$$

3.2.2 Загальний ефект від впровадження системи інформаційної безпеки

Загальний ефект від впровадження системи інформаційної безпеки визначається з урахуванням ризиків порушення інформаційної безпеки і становить:

$$E = B \cdot R - C \text{ грн.},$$

де B – загальний збиток від атаки у разі перехоплення інформації, тис. грн.;

R – вірогідність успішної реалізації атаки на сегмент мережі, частки одиниці (R=0,2);

C – щорічні витрати на експлуатацію системи інформаційної безпеки, тис. грн.

Загальний ефект від впровадження системи інформаційної безпеки визначається з урахуванням ризиків порушення інформаційної безпеки:

$$E = 18883200*0,2 - 472733,1 = 3303907 \text{ грн.}$$

3.3 Визначення та аналіз показників економічної ефективності системи інформаційної безпеки

Коефіцієнт повернення інвестицій ROSI показує, скільки гривень додаткового прибутку приносить одна гривня капітальних інвестицій на впровадження системи інформаційної безпеки:

$$ROSI = \frac{\text{E}}{K}, \quad \text{частки одиниці,}$$

де E – загальний ефект від впровадження системи інформаційної безпеки грн.;

К – капітальні інвестиції за варіантами, що забезпечили цей ефект, грн.

Коефіцієнт повернення інвестицій ROSI:

$$ROSI = \frac{3303907}{427048} = 7{,}74, \quad \text{частки одиниці,}$$

Проект визнається економічно доцільним, якщо розрахункове значення коефіцієнта повернення інвестицій перевищує величину річної депозитної ставки з урахуванням інфляції:

$$\text{ROSI} > (\text{N}_{\text{деп}} - \text{N}_{\text{інф}})/100),$$

де $\text{N}_{\text{деп}}$ – річна депозитна ставка, (5,5 %);

$\text{N}_{\text{інф}}$ – річний рівень інфляції, (5%).

Розрахункове значення коефіцієнта повернення інвестицій:

$$7{,}74 > (5{,}5 - 5)/100 = 7{,}74 > 0{,}005.$$

Термін окупності капітальних інвестицій $\text{T}_{\text{о}}$ показує, за скільки років капітальні інвестиції окупляться за рахунок загального ефекту від впровадження системи інформаційної безпеки. Відповідно термін окупності розробки методики із виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку:

$$T_O = \frac{K}{E} = \frac{1}{ROSI} = \frac{1}{7{,}74} = 0{,}13, \quad \text{років (1,6 місяців).}$$

3.4 Висновок

Згідно з наведеними розрахунками можна дійти висновку, що розробка методики із виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку, є економічно доцільною. Такі висновки зроблені, виходячи з коефіцієнту повернення інвестицій ROSI, який складає 7,74 (ROSI=7,74) та означає, що на 1 гривню капітальних витрат приходиметься 7,74 грн. економічного ефекту. Період окупності при цьому складе 0,13 років або 1,6 місяців. Капітальні інвестиції плануються на рівні 427048 грн., а експлуатаційні витрати в обсязі 472733,1 грн. на рік.

ВИСНОВКИ

Дослідження веб-додатків свідчать про велику кількість вразливостей та загроз для банківського програмного забезпечення, тому для перевірки наявності вразливостей та оцінки захищеності необхідна велика кількість часу. Постає питання ефективного розподілу ресурсів, яке вирішується впровадженням автоматизації тестування вразливостей. Але існуючі інструменти мають суттєві недоліки, а саме: вони мають велику ціну; недостатньо ефективні; мають закриті алгоритми аналізу (недоступність програмного коду); не здатні підтримувати механізми двокрокової автентифікації; мають ймовірний поріг помилкового спрацювання.

Саме тому в роботі були вирішити наступні задачі:

– проаналізовані та класифіковані вразливості веб-додатків;

– розроблена методика комплексної оцінки захищеності веб-додатків;

– проаналізовано сучасні методи пошуку вразливостей та автоматизовані сканери вразливостей;

– згідно частини методики розроблена підсистема пошуку вразливостей веб-додатків та доведена її ефективність завдяки використання методу “моделювання атак”. Для перевірки наявності вразливостей використовувались змодельовані атаки, в разі, якщо атака виконується успішно, робиться висновок про наявність вразливості, в іншому випадку результати свідчать про її відсутність.

В порівнянні з існуючими сканерами безпеки розроблена підсистема має ряд переваг та повністю реалізує поставлені до системи вимоги. А саме, результати випробувань свідчать про те, що вдалося повністю виключити помилки першого роду, та мінімізувати прояв помилок другого роду.

Підсистема здатна підтримувати всі існуючі механізми автентифікації у веб-додатках, на відміну від інших існуючих рішень. Сканування виконується наглядно, в кожний момент часу спеціаліст може контролювати процес та бачити, що саме перевіряється.

В подальшій роботі планується реалізувати розроблену комплексну методику пошуку вразливостей повністю та використовувати при виконанні оцінки захищеності веб-додатків комерційного банку.

ПЕРЛІК ПОСИЛАНЬ

1 Кількість і структура користувачів Інтернету в Україні (Електрон. ресурс) / Спосіб доступу: URL: http://www.gfk.ua/imperia/md/content/gfkukraine/presentations/20120517_imu_vyshlinsky.pdf. – Загол. з екрана.

2 December 2012 Web Server Survey (Електрон. ресурс) / Спосіб доступу: URL: http://news.netcraft.com/archives/2012/. – Загол. з екрана.

3 The Open Source Vulnerability Database (Електрон. ресурс) / Спосіб доступу: URL: http://osvdb.org/browse. – Загол. з екрана.

4 Закон України “Про банки і банківську діяльність”.

5 Закон України “Про підприємства в Україні”.

6 Закон України “Про інформацію”.

7 Закон України “Про захист інформації в інформаційно-телекомунікаційних системах”.

8 HTTP Response Splitting (Електрон. ресурс) / Спосіб доступу: URL: https://www.owasp.org/index.php/HTTP_Response_Splitting. – Загол. з екрана.

9 Вишняков В.М. Захист даних в інформаційних системах: навчальний посібник / В.М. Вишняков. - К.: КНУБА, 2010. - 128 с.

10 НД ТЗІ 1.1-003-99 “Термінологія в галузі захисту інформації в комп’ютерних системах від несанкціонованого доступу”.

11 The WASC Threat Classification v2.0 (Електрон. ресурс) / Спосіб доступу: URL: http://projects.webappsec.org/w/page/13246978/Threat%20Classification. – Загол. з екрана.

12 Статистика вразливостей веб-додатків (Електрон. ресурс) / Спосіб доступу: URL: http://ptsecurity.ru/download/статистика%20RU.pdf. – Загол. з екрана.

13 НД ТЗІ 2.5-010-03 “Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу”.

14 НД ТЗІ 2.7-009-09 “Методичні вказівки з оцінювання функціональних послуг безпеки в засобах захисту інформації від несанкціонованого доступу”.

15 OWASP TOP 10 2013 (Електрон. ресурс) / Спосіб доступу: URL: https://www.owasp.org/index.php/Top_10_2013-Top_10. – Загол. з екрана.

16 Лайза Криспин, Джанет Грегори. Гибкое тестирование. Практическое руководство для тестировщиков ПО и гибких команд, Москва "Вильямс" 2012. – 464 с.

17 An Introduction to HTTP fingerprinting 2013 (Електрон. ресурс) / Спосіб доступу: URL: http://netsquare.com/httprint/httprint_paper.html. – Загол. з екрана.

18 Language-Based Information-Flow Security (Електрон. ресурс) / Спосіб доступу: URL: http://www.cse.chalmers.se/~andrei/jsac.pdf. – Загол. з екрана.

19 Web Application Security Assessment by Fault Injection and Behavior Monitoring (Електрон. ресурс) / Спосіб доступу: URL: http://dl.acm.org/citation.cfm?id=775174. – Загол. з екрана.

20 Meseguer Security Policies and Security Models. (Електрон. ресурс) / Спосіб доступу: URL: http://spy.sci.univr.it/papers/Isa-orig/Sicurezza/NonInterferenza/noninter.pdf. – Загол. з екрана.

21 A type-based approach to program security (Електрон. ресурс) / Спосіб доступу: URL: http://users.cis.fiu.edu/~smithg/papers/tapsoft97.pdf. – Загол. з екрана.

22 Методы обнаружения уязвимостей в web-приложениях (Електрон. ресурс) / Спосіб доступу: URL: http://lvk.cs.msu.su/~ddk/pubs/methods_for_webapp_vuln_scanning.pdf. – Загол. з екрана.

23 “Укртелеком” заявил о DDos-атаке на свои серверы (Електрон. ресурс) / Спосіб доступу: URL: http://podrobnosti.ua/internet/2010/10/08/721600.html. – Загол. з екрана.

24 The Open Web Application Security Project (OWASP) (Електрон. ресурс) / Спосіб доступу: URL: https://www.owasp.org/. – Загол. з екрана.

25 Payment Card Indastry Data Security Standard (PCI DSS).

26 Payment Application Data Security Standard (PA DSS).

27 Nmap Security Scanner (Електрон. ресурс) / Спосіб доступу: URL: http://nmap.org/. – Загол. з екрана.

28 Selenium - Web Browser Automation (Електрон. ресурс) / Спосіб доступу: URL: http://docs.seleniumhq.org/. – Загол. з екрана.

29 Річний звіт, ПАО КБ "ПриватБанк", 2012. (Електрон. ресурс) / Спосіб доступу: URL: http://old.privatbank.ua/files/2_13_22 yearau.pdf. – Загол. з екрана.

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

№	Формат	Найменування	Кількість листів	Примітка
1	А4	Реферат	3	
2	А4	Список умовних скорочень	2	
3	А4	Зміст	2	
4	А4	Вступ	2	
5	А4	1 Розділ	42	
6	А4	2 Розділ	23	
7	А4	3 Розділ	12	
8	А4	Висновки	1	
9	А4	Перелік посилань	3	
10	А4	Додаток А	1	
11	А4	Додаток Б	1	
12	А4	Додаток В	1	
13	А4	Додаток Г	1	
14	А4	Додаток Д	1	

ДОДАТОК Б. Перелік документів на оптичному носії

1 Титульна сторінка.doc

2 Завдання.doc

3 Реферат.doc

4 Список умовних скорочень.doc

5 Зміст.doc

6 Вступ.doc

7 Розділ 1.doc

8 Розділ 2.doc

9 Розділ 3.doc

10 Висновки.doc

11 Перелік посилань.doc

12 Додаток А.doc

13 Додаток Б.doc

14 Додаток В.doc

15 Додаток Г.doc

16 Додаток Д.doc

Презентація.pptx

ДОДАТОК В. Відгуки керівників розділів

Відгук керівника економічного розділу:

Керівник розділу _____ _____

(підпис) (ініціали, прізвище)

ДОДАТОК Г. ВІДГУК

на кваліфікаційну роботу магістра на тему:
Обґрунтування методів виявлення вразливостей при забезпеченні безпеки веб-додатків комерційного банку
студента групи 125м-19-2
Бардака Ігоря Андрійовича

Пояснювальна записка складається з титульного аркуша, завдання, реферату, списку умовних скорочень, змісту, вступу, трьох розділів, висновків, переліку посилань та додатків, розташованих на __ сторінках та містить __ рисунків, __ таблиць, __ джерела та __ додатка.

Зміст та структура роботи дозволяють розкрити поставлену тему повністю.

Студент показав достатній рівень володіння теоретичними положеннями з обраної теми, показав здатність формувати власну точку зору (теоретичну позицію).

Робота виконана самостійно. У ході виконання роботи були вирішені наступні завдання: проведено аналіз загроз для веб-додатків, проаналізовані методи пошуку вразливостей, досліджено їх вплив на стан захищеності інформації, що оброблюється веб-сервером. Проведено аналіз сучасних автоматизованих сканерів безпеки, та розроблено власне програмне забезпечення для пошуку вразливостей, що продемонструвало новий підхід у вирішенні проблеми безпеки веб-додатків та дозволило підвищити ефективність захисту банківських ресурсів.

Це підтверджує самостійність обробки даних, практичні рекомендації та висновки.

Робота оформлена та написана грамотною мовою. Містить необхідний ілюстрований матеріал. Автор добре знає проблему, уміє формулювати наукові та практичні завдання і знаходить адекватні засоби для їх вирішення.

В цілому робота задовольняє усім вимогам і може бути допущена до захисту, а його автор Бардак Ігор Андрійович заслуговує на оцінку «_____».

Керівник роботи,

ДОДАТОК Д. Програмний код підсистеми пошуку вразливостей

Файл Main.cs

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Threading;
using System.Windows.Forms;
using General;
using Microsoft.Win32;

namespace ServerScaner
{
 static class Program
 {
  [STAThread]
  static void Main()
  {
   Application.EnableVisualStyles();
   Application.SetCompatibleTextRenderingDefault(false);
   if (!ChecIIsNmapInstalled())
   {
    MessageBox.Show("На вашому ПК не знайдено встановленного Nmap (Network mapper). Будь-ласка встановіть його для подальшої роботи з додатком.");
    Process.Start("http://nmap.org/download.html");
    Application.Exit();
   }
   else
   Application.Run(new MainAppForm());
  }

  // Функция проверят установлен ли Nmap на ПК
  static bool ChecIIsNmapInstalled()
  {
   RegistryKey Namp = Registry.CurrentUser.OpenSubKey("Software").OpenSubKey("Nmap");
   if (Namp != null)
   {
    return true;
   }
   else return false;
  }

  public static void KillProcess(string nameOfProcess)
  {
   Process[] processes = Process.GetProcessesByName(nameOfProcess);
   foreach (Process process in processes)
   {
    try
    {
     if (!process.HasExited)
```