

**МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ**

**ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І
СИСТЕМ**

**КВАЛІФІКАЦІЙНА РОБОТА
на тему: СИСТЕМА ЗАХИСТУ ВЕБ-РЕСУРСІВ УСТАНОВИ**

Здобувача вищої освіти другого (магістерського) рівня,
II курсу, групи КМІз-21-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології в економіці»



(підпис)

Джосана Владислава Валерійовича

(прізвище, ім'я, по батькові)

Науковий керівник



(підпис)

Погореловська Ірина Дем'янівна,
к.е.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Керівник освітньо-
професійної програми



(підпис)

Горбовий Артур Іоліанович,
д.т.н, професор

(прізвище, ім'я, по батькові, науковий ступінь,
вчене звання)

Допущено до захисту за рішенням кафедри
протокол від «12» січня 2023 № 6

Завідувач кафедри



(підпис)

Філоненко Михайло Миколайович,
к.ф.-м.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене
звання)

Ірпінь – 2023

Реєстрація № 2

12.01.2023

(дата)



(підпис відповідальної особи)

Результати перевірки

до захисту
(до захисту, на доопрацювання)

12.01.2023

(дата)



(підпис)

Результат захисту

4 (добре)
(оцінка)

13.02.2023

(дата)



(підпис)

Голова комісії:



(підпис)

В.М. Франчук

Члени комісії:



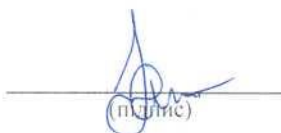
(підпис)

М.М. Філоненко



(підпис)

О.В. Редич



(підпис)

Є.В. Котух

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії, секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	27		13.02.2023 Франчук В.М. <i>В.М.</i> Філоненко М.М. <i>М.М.</i> Редич О.В. <i>О.В.</i> Котух С.В. <i>С.В.</i> Суховерхівка Л.П. <i>Л.П.</i>
2. Дотримання вимог щодо оформлення	від 0 до 10	7		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	4		
4. Рецензент	від 0 до 5	4		
РАЗОМ БАЛІВ	від 0 до 50	42		

Оцінювання кваліфікаційної роботи науковим керівником - 43 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підписи)
85	B	4 (добре)	13.02.2023	Франчук В.М. <i>В.М.</i> Філоненко М.М. <i>М.М.</i> Редич О.В. <i>О.В.</i> Котух С.В. <i>С.В.</i> Суховерхівка Л.П. <i>Л.П.</i>

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	
50-59	E	3 (задовільно)
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій

Кафедра комп'ютерних та інформаційних технологій і систем

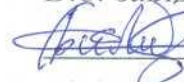
Галузь знань: 12 «Інформаційні технології»

Спеціальність: 122 «Комп'ютерні науки»

Освітня програма: «Інформаційні управляючі системи і технології в економіці»

Затверджую

В.о. завідувача кафедри

 І.М. Федотова-Півень

10.01.2022

**Завдання на кваліфікаційну (дипломну) роботу здобувача
вищої освіти**

Джосана Владислава Валерійовича

1. Тема кваліфікаційної роботи: Система захисту веб-ресурсів установи
затверджена наказом ректора від 05.01.2022 № 4

2. Строк здачі закінченої роботи: 11.01.2023

Цільова установка: застосування набутих знань, умінь для організації захисту
веб ресурсів з використанням сучасних підходів та технологій

3. Мета роботи: визначити способи та засоби захисту веб-ресурсів від кібератак

Об'єкт дослідження: методи та інструменти захисту веб-ресурсів

Предмет дослідження: програмно-апаратні засоби, технології, методи захисту
веб-ресурсів

4. Вихідні дані до роботи: правила та вимоги до захисту веб-ресурсів,
методика проведення пентесту, статистика кіберзагроз, спеціалізовані
інструменти захисту веб-ресурсів

5. Перелік графічного матеріалу: архітектура веб-ресурсу, схема інтеграції
систем IPS, WAF, види загроз, копії екранів, презентація роботи (.ppt)

6. Зміст кваліфікаційної (дипломної) роботи: Розділ 1. Вимоги щодо захисту
веб-ресурсів від несанкціонованого доступу. 1.1. Веб-ресурси та їх
вразливості. 1.2. Вимоги щодо захисту веб-ресурсів від несанкціонованого
доступу. Розділ 2. Аналітичний огляд засобів захисту веб-ресурсів. 2.1.
Міжмережні екрани як засіб захисту веб-додатків. 2.2. Використання систем
IPS та міжмережних екранів. 2.3. Пентест як спосіб перевірки захищеності
веб-ресурсів. Розділ 3. Захист веб-ресурсів установи. 3.1. Способи захисту
веб ресурсів від загроз. 3.2. Основні засоби і способи захисту Python веб-
додатків.

7. Календарний план виконання роботи

№ п/п	Назва етапів випускної кваліфікаційної (дипломної) роботи	Строк виконання етапів роботи	
		За планом	Фактично
1	Вибір теми, підбір літературних джерел, формування структури ДР	24.12.2021	«27» 12 2021 р.
2	1. Вступ та I розділ (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	26.05.2022	«26» 05 2022 р.
3	II розділ (аналітична частина проекту)	20.11.2022	«20» 11 2022 р.
4	III розділ (експериментальні дослідження та проектні рішення)	30.11.2022	«30» 11 2022 р.
5	Висновки	15.12.2022	«15» 12 2022 р.
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	10.01.2023	«10» 01 2023 р.
7	Перевірка на наявність схожості	31.01.2023	«31» 01 2023 р.
8	Отримання відгуку керівника	02.02.2023	«02» 02 2023 р.
9	Отримання зовнішньої рецензії	06.02.2023	«06» 02 2023 р.

8. Дата видачі завдання: 10.01.2022

9. Науковий керівник
кваліфікаційної роботи


(підпис)

Погореловська І. Д., к.е.н., доцент

10. Гарант
освітньо-професійної програми


(підпис)

Горбовий А. Ю., д.т.н., професор

11. Завдання до виконання
отримав здобувач вищої освіти


(підпис)

Джосан В. В.

АНОТАЦІЯ

Система захисту веб-ресурсів установи (назва випускної кваліфікаційної (дипломної) роботи)

Мета роботи: визначити способи та засоби захисту веб-ресурсів установи від кібератак.

Завдання роботи, які були вирішені у процесі виконання дипломної роботи:

- уточнено поняття «веб-ресурс»;
- визначено вимоги щодо захисту веб-ресурсів від несанкціонованого доступу;
- здійснено аналітичний огляд засобів захисту веб-ресурсів.
- визначено способи захисту веб-ресурсів установи від кіберзагроз.

Об'єкт дослідження: методи та інструменти захисту веб-ресурсів.

Предмет дослідження: програмно-апаратні засоби, технології, методи захисту веб-ресурсів.

Інформаційною базою дослідження є правила та вимоги до захисту веб-ресурсів, методика проведення пентесту, статистика кіберзагроз, спеціалізовані інструменти захисту веб-ресурсів.

За результатами дослідження було сформульовано наступні висновки: при формуванні системи захисту веб-ресурсів установи необхідно: опрацювати правила та вимоги захисту веб-ресурсів; ознайомитися зі статистикою кібератак, визначити які саме атаки є найпопулярнішими і, у яких сферах діяльності, які вразливості програмних рішень вони використовують для проникнення в систему безпеки та виходячи з цього, визначати способи захисту веб-ресурсів від цих загроз; сформувати комплексну систему захисту веб-ресурсів на всіх рівнях моделі OSI, в якій будуть поєднані Firewall, IPS та WAF; здійснити за допомогою пентесту оцінку захищеності веб-ресурсів та визначити найбільш критичні ризики для їх безпеки.

Одержані результати кваліфікаційної роботи (проєкту) можуть бути використані для зменшення кібервразливості веб-ресурсів установи, а також в освітньому процесі при викладанні навчальних дисциплін “Комп'ютерні мережі”, “Технології захисту інформації”.

Дипломна робота містить 47 сторінок, 2 таблиці, 9 рисунків, список використаних джерел з 49 найменувань

Ключові слова: веб-ресурси, вразливості, захист, кібератаки, кіберзагрози, Firewall, IPS, WAF, пентест.

Рік виконання дипломної роботи: 2022-2023

Рік захисту роботи: 2023.

SUMMARY

Purpose: to identify ways and means of protecting the institution's web resources from cyberattacks

Tasks of the work, which were solved in the process of completing the thesis: the concept of «web resource» is clarified;

- requirements for protecting web resources from unauthorized access were determined;

- an analytical review of the means of protecting web resources was carried out.

- ways to protect the institution's web resources from cyber threats are determined.

Object of research - methods and tools for protecting web resources.

Subject of research - software and hardware, technologies, methods of protecting web resources.

The information base of the study is the rules and requirements for the protection of web resources, the methodology of pentesting, statistics of cyber threats, specialized tools for protecting web resources.

Based on the results of the study, it was concluded that when forming a web resource protection system for an institution, it is necessary to: work out the rules and requirements for protecting web resources; familiarize yourself with the statistics of cyber attacks, determine which attacks are the most popular and in which areas of activity, which vulnerabilities of software solutions they use to penetrate the security system and, based on this, determine how to protect web resources from these threats; form a comprehensive web resource protection system at all levels of the OSI model, which will combine Firewall.

The obtained results of the qualification work (project) can be used to reduce the cyber vulnerability of the institution's web resources, as well as in the educational process in teaching the disciplines «Computer Networks», «Information Security Technologies».

The thesis contains 47 pages, 2 tables, 9 figures, a list of references of 49 titles.

Keywords: web resources, vulnerabilities, protection, cyber attacks, cyber threats, Firewall, IPS, WAF, pentest.

Year of completion of the thesis: 2022-2023

Year of thesis defense: 2023.

ЗМІСТ

ВСТУП.....	3
РОЗДІЛ 1. Вимоги щодо захисту веб-ресурсів від несанкціонованого доступу.	5
1.1. Веб-ресурси та їх вразливості.	5
1.2. Вимоги щодо захисту веб-ресурсів від несанкціонованого доступу	8
РОЗДІЛ 2. Аналітичний огляд засобів захисту веб-ресурсів	13
2.1. Міжмережеві екрани як засіб захисту веб-додатків.....	13
2.2. Використання систем IPS та міжмережєвих екранів	16
2.3. Пентест як спосіб перевірки захищеності веб-ресурсів	19
РОЗДІЛ 3. Захист веб-ресурсів установи	23
3.1. Способи захисту веб ресурсів від загроз.....	23
3.2. Основні засоби і способи захисту Python веб-додатків.....	32
ВИСНОВКИ.....	40
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	43

ВСТУП

Актуальність роботи. Захист веб-ресурсів залишається одним із важливих напрямів інформаційної безпеки. Щороку кількість веб-ресурсів збільшується, зростає також кількість конфіденційної інформації, яка локалізується на серверах віддаленого доступу (особливо із використанням хмарних технологій). У результаті цього зростають не тільки кількість атак на веб-ресурси, але й негативні економічні наслідки таких атак.

Останнім часом вразливість веб-ресурсів до атак отримала політичний вимір унаслідок як поширення гібридних війн у світі, так і зростання терористичних загроз.

Таким чином, удосконалення методів і систем захисту веб-ресурсів від атак залишається актуально науковою проблемою, особливо з урахуванням постійного вдосконалення методів та інструментів атак і появи нових методів та інструментів. Удосконалення методів захисту веб-ресурсів від атак є також важливою в практичному застосуванні задачею внаслідок зростаючих економічних, соціальних і політичних наслідків від зловмисних дій.

Орієнтуючись на актуальність обраної теми, було визначено **мету виконання** кваліфікаційної роботи – визначити способи та засоби захисту веб-ресурсів від кібератак.

Для досягнення визначеної мети були вирішені наступні завдання:

- уточнено поняття «веб-ресурс»;
- визначено вимоги щодо захисту веб-ресурсів від несанкціонованого доступу;
- здійснено аналітичний огляд засобів захисту веб-ресурсів.
- визначено способи захисту веб-ресурсів установи від кіберзагроз.

Об'єкт дослідження: методи та інструменти захисту веб-ресурсів.

Предмет дослідження: програмно-апаратні засоби, технології, методи

захисту веб-ресурсів.

Методи дослідження: при написанні дипломної роботи були використані наступні методи: системний аналіз, який застосовувався при роботі з інформаційними ресурсами за темою дослідження; методи порівняння, оцінювання та групування при аналізі можливостей міжмережевих екранів для веб-додатків; методи групування – при формуванні системи моніторингу стану комп'ютерної мережі; за допомогою методу узагальнення були сформувані висновки.

Інформаційною базою дослідження є правила та вимоги до захисту веб-ресурсів, методика проведення пентесту, статистика кіберзагроз, спеціалізовані інструменти захисту веб-ресурсів

Структура дипломної роботи містить вступ, основну частину, яка складається з трьох розділів, до яких входять 9 рисунків та 2 таблиці, висновки, перелік використаної літератури, що містить 49 джерел.

За результатами дослідження визначено способи та засоби захисту веб-ресурсів від кібератак, а саме, засоби контролю безпеки, наявності вразливостей та помилок конфігурації систем і сервісів, засоби запобігання кібератакам та способи захисту веб-додатків розроблених на Python.

Науково-практичне значення одержаних результатів: результати кваліфікаційної роботи (проєкту) можуть бути використані для зменшення кібервразливості веб-ресурсів установи, а також в освітньому процесі при викладанні навчальних дисциплін «Комп'ютерні мережі», «Технології захисту інформації».

РОЗДІЛ 1. ВИМОГИ ЩОДО ЗАХИСТУ ВЕБ-РЕСУРІВ ВІД НЕСАНКЦІОНОВАНОГО ДОСТУПУ

1.1. Веб-ресурси та їх вразливості

Веб-ресурс - це сукупність інтегрованих технічних і програмно-апаратних ресурсів та інформації, призначених для публікації у всесвітній мережі Інтернет [4].

Веб-ресурсом також можна вважати веб-сайт, веб-сторінку, віртуальний файл, що надається користувачеві у вигляді HTML-документа та має унікальну URL-адресу в мережі. Веб-ресурси установи розміщені на виділених веб-серверах, які забезпечують доступ до них за протоколом HTTP [15].

Веб-ресурси містять загальнодоступну та технологічну інформацію. У нормативному документі про системи технічного захисту інформації зазначено, що «до загальнодоступної інформації відноситься публічно оприлюднена інформація, яка може бути використана будь-якою фізичною або юридичною особою (користувачем інформаційних ресурсів), що має доступ до мережі Інтернет. До технологічної інформації ІТС належить технологічна інформація комп'ютерної системи захисту інформації та технологічна інформація, пов'язана з управлінням та адмініструванням ІТС і засобів обробки інформації, - дані про мережеві адреси, імена, персональні ідентифікатори та паролі користувачів, їх повноваження та права доступу до об'єктів, інформація журналу дій користувачів, інша інформація бази даних захисту» [1, с.3].

Типова архітектура веб-ресурсу, як правило, включає наступні основні елементи (рис.1.1):

- публічні веб-сервери, які забезпечують доступ користувачів мережі Інтернет до інформаційних ресурсів порталу;
- кеш-сервери, що забезпечують тимчасове зберігання копії ресурсу, до яких мали доступ користувачі Інтернету. При зверненні до веб-ресурсу здійснюється спроба витягти ресурс з кешу серверів, і лише якщо ресурс там відсутній, то запит перенаправляється на загальнодоступні веб-сервери. Використання кеш-серверів дозволяє зменшити навантаження на основні публічні

сервери, а також скоротити час доступу користувачів до кешованих ресурсів;

- DNS-сервери, які надають можливість конвертації символічних імен серверів веб-ресурсів у відповідні їм IP-адреси;

- сервери додатків, на яких встановлено спеціалізоване програмне забезпечення, призначене для управління інформаційним наповненням веб-ресурсу;

- сервери баз даних, що забезпечують централізоване зберігання інформаційних джерел веб-ресурсів;

- обладнання зв'язку, що забезпечує взаємодію між різними серверами веб-ресурсів.

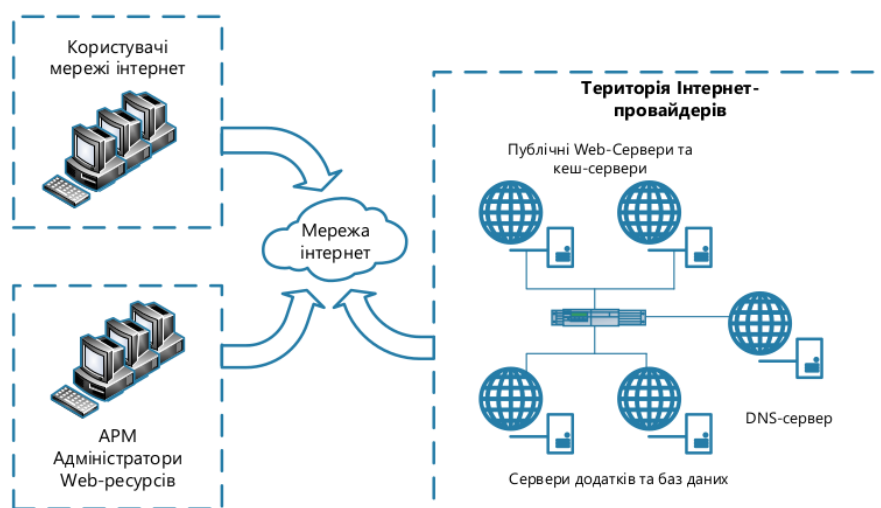


Рис. 1. 1. Типова архітектура Web-ресурсу [10]

Враховуючи той факт, що ресурси публічного веб-ресурсу за визначенням є відкритими для будь-якого користувача мережі Інтернет, вони стають потенційною мішенню для атак з боку зломисників. Слід зазначити, що останнім часом спостерігається значне збільшення кількості інформаційних атак, основна частина яких спрямована саме на загальнодоступні ресурси, у тому числі веб-портали. Атаки зломисників можуть бути спрямовані на порушення конфіденційності, цілісності або доступності інформаційних ресурсів, що зберігаються на серверах веб-ресурсу.

Щоб правильно організувати захист веб-ресурсів, написати безпечний

веб-додаток, необхідно розуміти, як виникають вразливості і як вони можуть бути використані. Як зазначено в роботі [7], «для веб-додатків та веб-сервісів, як і для інших видів програмного забезпечення, в принципі не існує стандартного алгоритму несанкціонованого вторгнення зловмисників. Кожен випадок має розглядатися індивідуально. Щоразу для здійснення атаки використовуються різні набори інструментів та методів проникнення в системи безпеки. Однак, всі випадки об'єднує одне - всі вони використовують інформацію про існуючі вразливості в програмних системах. Це можуть бути як відомі уразливості, які не були «закриті» через недогляд обслуговуючого персоналу ресурсу, так і уразливості нульового дня. Тобто ті вразливості, про які вперше стає відомо тільки після того, як вони були використані в атаці саме на цей тип програмного комплексу».

Одними з найбільш поширених та небезпечних веб-уразливостей, за даними консорціуму Open Web Application Security Project (OWASP), наразі є:

« - міжсайтовий скриптинг (Cross-site scripting, XSS) - дозволяє зловмиснику перехопити конфіденційну інформацію та виконати шкідливий код у веб-браузері;

- SQL-ін'єкції (SQL-injection) - механізми атак у веб-додатках, які використовують введені користувачем дані в SQL-запитах до бази даних, не вимагаючи попередньої обробки для видалення потенційно небезпечних символів і зарезервованих слів;

- підробка міжсайтових запитів (cross-site request forgery, CSRF) - один із способів використання вразливих міжсайтових скриптів, який полягає у підробці HTTP-запитів, за допомогою якого зловмисник має можливість виконувати запити від імені легітимного користувача; ін'єкція команд (command injection) - має місце, коли дані, що надсилаються на сервер у HTTP-запиті, використовуються для генерації команд операційної системи без перевірки на наявність спеціальних символів» [43].

За даними [45] «критично небезпечна уразливість «SQL-injection» виявляється майже в половині веб-додатків (48%). Саме ця уразливість може приз-

вести до отримання несанкціонованого доступу до інформації, що зберігається в базах даних додатків, або навіть отримання повного контролю над сервером. Більше половини веб-сайтів містять уразливості, пов'язані з використанням передбачуваних значень ідентифікаторів користувачів і сесій (Credential/Session Prediction)».

Поширеною помилкою адміністрування веб-серверів є «Ідентифікація програмного забезпечення» (Fingerprinting). Зокрема, дана уразливість зустрічається на 8 з 10 веб-ресурсів під управлінням Apache. Це пов'язано з тим, що стандартна конфігурація досліджуваних серверів передбачає розкриття інформації про версії веб-сервера в повідомленнях про помилки. Лідером за кількістю систем з уразливостями високого рівня ризику опинилася банківська галузь (89%). Також високий відсоток веб-додатків, схильних до критично небезпечної уразливості, зазначається для телекомунікаційної галузі (80%). Далі йдуть промисловість (71%), інформаційні технології (67%) та електронна комерція (42%) [43].

Звичайно, найкращий захист - це відсутність вразливостей, тому ще на етапі початкового налаштування політики безпеки веб-сервера або компіляції веб-додатку необхідно опрацювати існуючі фреймоврки з кібербезпеки, які містять вимоги та пропозиції щодо захисту веб-ресурсів установи, а також продумати, до яких дій може вдатися зловмисник та заздалегідь вжити заходів.

1.2. Вимоги щодо захисту веб-ресурсів від несанкціонованого доступу

За результатами проведеного аналізу було визначено, що рекомендації і правила, методології, концепції, інструменти, вимоги організації захисту веб-ресурсів установи містяться у фреймоврках з кібербезпеки, зокрема:

- **ACSC Essential 8** (Australian Cyber Security Center - Австралійський центр з кібербезпеки)[44] — це документ в якому визначені критерії контролю кібербезпеки, які спрямовані на підвищення стійкості бізнесу, а саме:

- «Оцінка вразливостей безпеки та застосування патчів (оновлення);

- Впровадження контролю програм і додатків;
 - Конфігурація програм і додатків;
 - Безпека платформи Microsoft Office (зокрема, макросів);
 - Рекомендації по захисту операційної системи;
 - Обмеження адміністративних привілеїв (налаштування прав доступу);
 - Запровадження багатофакторної авторизації (MFA/2FA);
- Регулярне резервне копіювання» .

Відповідність цим критеріям оцінюється трьохрівневою шкалою «зрілості»:

- перший рівень зрілості – часткового відповідає цілям стратегії;
- другий рівень зрілості – в основному відповідає цілям стратегії;
- третій рівень зрілості – повністю відповідає цілям стратегії.

Такий поділ дозволяє організаціям визначити свій поточний стан відповідності вимогам з кібербезпеки та розуміти, які конкретно зусилля необхідні для просування далі відповідно цього фреймворку.

- **CIS Critical Security Controls v8** – збірка практичних порад і рекомендацій з комп'ютерної безпеки, що об'єднує та консолідує 18 правил з кібербезпеки [47].

- **NIST** (National Institute of Standards and Technology) Cybersecurity Framework – один з найбільш впливових та визнаних світовою спільнотою фреймворків безпеки інформаційних систем, розроблений на державному рівні в США [47]. Містить перелік кращих практик і рекомендацій з кібербезпеки, зокрема серія NIST 800-53 включає близько 20-ти розділів управління кібербезпекою.

- **ISA/IEC 62443** – серія міжнародних стандартів безпеки для промислових систем автоматизованого управління, розроблена комітетом ISA (International Society of Automation) і прийнята Міжнародною електротехнічною комісією (IEC). Забезпечує гнучку структуру для усунення та пом'якшення поточних і майбутніх вразливостей, ризиків безпеки в системах промислової автоматизації та контролю. IEC 62443 містить усього сім вимог

безпеки, а саме: контроль ідентифікації та автентифікації; застосування контролю; цілісність системи; конфіденційність даних; обмежений потік даних; доступність ресурсів [42].

- **Міжнародний стандарт інформаційної безпеки ISO/IEC 27005:2018** (Information technology - Security techniques - Information security risk management) [9] - один із основних документів, які містять вказівки щодо управління ризиками інформаційної безпеки різних типів підприємств та установ, що мають намір керувати ризиками, які можуть поставити під загрозу їх інформаційну безпеку. Вимоги щодо інформаційної безпеки, які визначені у цьому стандарті, сформовані на основі аналізу та узагальнення найкращих методів, апробованих на практиці та визначають найкращі варіанти дій при виникненні інцидентів в інформаційній безпеці.

- **Вимоги до захисту інформації WEB-сторінки** від несанкціонованого доступу [1], які визначають, що “технологія оброблення інформації повинна бути здатною реалізовувати можливість виявлення спроб несанкціонованого доступу до інформації WEB-сторінки та процесів, які з цією інформацією пов’язані, а також забезпечити реєстрацію в системному журналі визначених політикою відповідної послуги безпеки подій”. Крім того, технологічними процесами повинна бути реалізована можливість створення резервних копій інформації WEB-сторінки та процедури їх відновлення з використанням резервних копій.

Програмно-апаратні засоби захисту повинні мати належним чином оформлені документи (експертні висновки, сертифікати), які засвідчують відповідність цих засобів вимогам нормативних документів системи захисту. Апаратура, в плані розподілу привілеїв, повинна мати вищий рівень безпеки. Для запобігання атакам веб-сервер повинен бути налаштований на режим «тільки читання», режиму запису на зовнішні жорсткі диски, магнітооптичні диски тощо повинен бути відсутнім.

Встановлення на ОС нових (додаткових) компонентів, ПЗ (системного та/або функціонального), сервісів та розміщення будь-яких інших мережеских

ресурсів, які не належать до категорії веб-ресурсів установи, не повинно порушувати політику безпеки інформації в комп'ютерній системі, що забезпечує функціонування веб-сторінки.

Забезпечення безпеки інформації вимагає виділення окремого ресурсу (комп'ютера) під кожне завдання. Інакше помилка в системі безпеки може порушити роботу відразу кількох сервісів. Зокрема, сервер електронної пошти, веб-сервер і сервер баз даних повинні бути розміщені на різних комп'ютерах. При цьому кожен сервер має бути оснащений системою захисту, інакше він може стати легкою мішенню для зловмисника.

Отже, комп'ютерна система захисту інформації повинна забезпечувати захист цілісності та доступності розміщеної на WEB-сторінці загальнодоступної інформації, а також конфіденційності та цілісності технологічної інформації WEB-сторінки. Крім того, «має бути реалізовано контрольований і санкціонований доступ до інформації та заборона неконтрольованої й несанкціонованої її модифікації» [1].

Незалежно від серйозності заходів, спрямованих на забезпечення безпеки Web-сервера, вірогідність проникнення повністю виключити неможливо. Тому, якщо це все ж таки сталося, важливо мінімізувати наслідки атаки. Розподіл привілеїв є ефективним способом досягнення цієї мети, при цьому кожен користувач зможе запускати тільки певні програми. Важливо також, щоб для користувачів, які порушили встановлені правила розмежування доступу до WEB-сторінки, система захисту на період сеансу роботи блокувала доступ до веб-сторінки.

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

- веб-сервер і робочі станції розміщуються на території установи-власника веб-сторінки або на території оператора (технологія T1), мінімально

необхідний функціональний профіль визначається:

КА–2, ЦА–1, ЦО–1, ДВ–1, ДР–1, НР–2, НИ–2, НК–1, НО–1, НЦ–1, НТ–1;

– 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 [1].

Позначення послуг безпеки згідно з НД ТЗІ 2.5-004: «ДВ-1 - ручне відновлення після збоїв; ДР-1 - квоти; КА-2 - базова адміністративна конфіденційність; КВ-1 - мінімальна конфіденційність при обміні; НВ-1 - автентифікація вузла; НИ-2 - одиночна ідентифікація і автентифікація; НК-1 - однонаправлений достовірний канал; НО-1 - розподіл обов'язків; НР-2 - захищений журнал; НТ-1 - самотестування за запитом; НЦ-1 - КЗЗ з контролем цілісності; ЦА-1 - мінімальна адміністративна цілісність; ЦВ-1 - мінімальна цілісність при обміні; ЦО-1 - обмежений відкат» [7] .

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

РОЗДІЛ 2. АНАЛІТИЧНИЙ ОГЛЯД ЗАСОБІВ ЗАХИСТУ ВЕБ-РЕСУРСІВ

2.1. Міжмережеві екрани як засіб захисту веб-додатків

Специфіка веб-додатків полягає в тому, що під час сеансу роботи користувача з веб-сервером може здійснюватися велика кількість різних TCP-з'єднань, які відкриваються з різних адрес, але мають (можливо, динамічний) ідентифікатор сеансу. Атаки на веб-ресурси, які маніпулюють програмним забезпеченням для досягнення зловмисних цілей, як правило, виконуються одночасно з легальними сеансами користувачів і зазвичай використовують стандартні HTTP-порти. Не рекомендується блокувати весь трафік на рівні цих портів, оскільки доступ до веб-ресурсів буде повністю заблокований ззовні або зсередини. Ця ситуація мережевої безпеки є ідеальною можливістю для хакерів знайти вразливості на рівні веб-додатків. Виходом з цієї ситуації може бути використання міжмережних екранів (англ. Firewall, нім. Brandmauer) для веб-додатків.

Міжмережні екрани (Firewall) - це «комплекс апаратно-програмних чи програмних засобів, що здійснює контроль та фільтрацію інформаційних потоків відповідно до заданих правил політики безпеки — за IP-адресами, портами відправника та отримувача. Типовими функціями брандмауера є контроль доступу до вузлів мережі, фільтрація доступу до незахищених сервісів, контроль порядку доступу до мережі, запобігання спробам доступу з зовнішньої чи внутрішньої мережі, запобігання доступу до секретної інформації з внутрішньої захищеної мережі. Міжмережні екрани забезпечують також антивірусний захист, створення каналів VPN, виявлення та захист від вторгнень» [37].

Firewall перешкоджає зловмисникам чи деяким типам шкідливих програм отримувати доступ до комп'ютера по мережі або через Інтернет. Для цього Firewall перевіряє дані, що надходять з Інтернету або по мережі, і блокує їх або дозволяє передачу на комп'ютер (рис. 2.1).

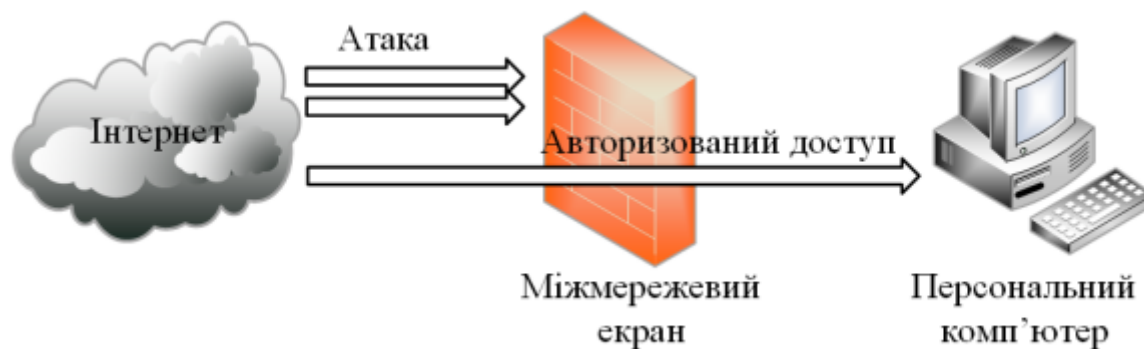


Рис. 2.1. Схема роботи міжмережевого екрану [31]

Традиційні мережні брандмауери працюють на рівнях 3 і 4 моделі OSI захищаючи мережевий трафік. З цієї причини, традиційний мережний брандмауер сам по собі не захистить від атак на веб-сторінки підприємства чи установи.

Для захисту від веб-атак класичним пристроєм міжмережний екран веб-додатків (Web Application Firewall, WAF). WAF - це програмний або апаратний брандмауер для захисту веб-додатків. Брандмауер WAF це вузькоспеціалізований пристрій, який застосовує набір правил захисту до протоколів високого (прикладного) рівня HTTP/HTTPS, FTP/FTPS. Такі пристрої поєднують в собі безліч сервісів для захисту від різних мережних атак - за принципом «все в одному» - і мають потенціал для швидкого розгортання, легкого масштабування і адаптації до потреб бізнесу. WAF дозволяє захищати клієнтів від SQL і PHP-ін'єкції, cross-site scripting (XSS), підбору паролів (bruteforce та інші), від експлуатації вразливостей zero-day в веб-аплікаціях, DDoS-атаки рівня додатків і атаки на інші ресурсомісткі сторінки. WAF здатний контролювати всі об'єкти, до яких можуть отримати доступ користувачі - URL-адреси, які вводяться, а також параметри GET і POST-запитів. Також брандмауер не дозволяє користувачам запускати об'єкти, які не належать веб-ресурсу» [6].

Реалізація міжмережного екрану в якості зворотного проксі-серверу є

найбільш популярною та найбільш вживаною (рис.2.2).

WAF знаходиться між зовнішніми користувачами і веб-додатками і аналізує весь HTTP/HTTPS-трафік, виявляючи і блокуючи шкідливі запити до того, як вони зможуть вплинути на користувачів або на веб-додаток.

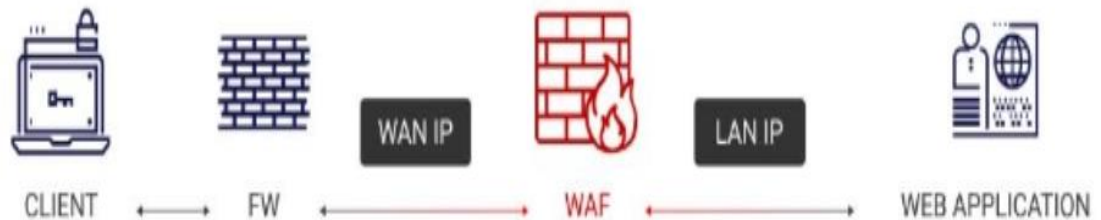


Рис. 2.2. Реалізація WAF в якості зворотнього проксі серверу [33]

WAF працює на основі набору правил, що називаються політиками, які використовуються для фільтрації кібератак. WAF захищають критично важливі для бізнесу веб-додатки і веб-сервери від більшості відомих на сьогоднішній день атак 7 рівня моделі OSI – рівні додатків, у тому числі, від загроз, що предсталені в OWASP Top 10.

Основними загрозами цього рівня є атаки на різного роду фреймворки, маніпуляція з файлами cookie, експлуатація SQL-ін'єкцій, атаки з використанням міжсайтових сценаріїв.

Основними функціями WAF є: «запобігання витоку даних - перевірка відповідей серверів на наявність критичних даних; перевірка всього контенту веб-сайту, включаючи HTML, DHTML і CSS, а також протоколів доставки контенту (HTTP/HTTPS); перевірка будь-якого протоколу або структури даних, що використовується для передачі даних веб-додатків; перевірка будь-якого протоколу або структури даних, що використовується для передачі даних веб-додатків; підтримка розриву SSL/TLS з'єднання; запобігання або виявлення підробки сесійних посвідчень; автоматичне оновлення сигнатур атак та їх реалізація; пристрій підтримує клієнтські сертифікати SSL; підтримка

апаратного зберігання ключів (FIPS)» [48].

Але цих загальних функцій WAF недостатньо для повноцінного захисту інтернет-ресурсів, зокрема від низькорівневих атак, таких як, DDoS, SYN-flood, TCP-flood, UDP-flood. Види атак, яким неспроможний протистояти WAF зведені у табл. 2.1.

Таблиця 2.1

Види атак, яким неспроможний протистояти Web Application Firewall

Типи атак	Дії WAF
Атаки направленні на викраденні даних	Блокує
Ін'єкції	Блокує
Цілеспрямована атака	Може блокувати, а може і ні в залежності від атаки
DDoS	Може бачити активність в мережі і зменшувати кількість запитів до ресурсу, але фактично не блокує
Ураження шкідливим ПЗ	Може блокувати, а може і ні в залежності від атаки
Атаки на компоненти з відомими вразливостями	Може блокувати, а може і ні в залежності від атаки
XSS	Блокує
Syn-flood	Не блокує
Атаки спрямовані на викрадання сесії викрадача	Блокує
TCP-flood	Не блокує
UDP-flood	Не блокує
CSRF	Блокує

[Складено з джерела 38]

Таким чином, виникає потреба у створенні комплексної системи безпеки захисту веб-ресурсів.

2.2. Використання систем IPS та міжмережевих екранів

Для підвищення ефективності системи захисту веб-ресурсів необхідно створити захист від **низькорівневих атак**. Для цього може бути застосований Firewall з системою запобігання вторгнення (англ. Intrusion Prevention

System, IPS). IPS - це програмна або апаратна система забезпечення безпеки, яка розпізнає ознаки вторгнення, виявляє атаки і активно блокує вторгнення (рис.2.3).

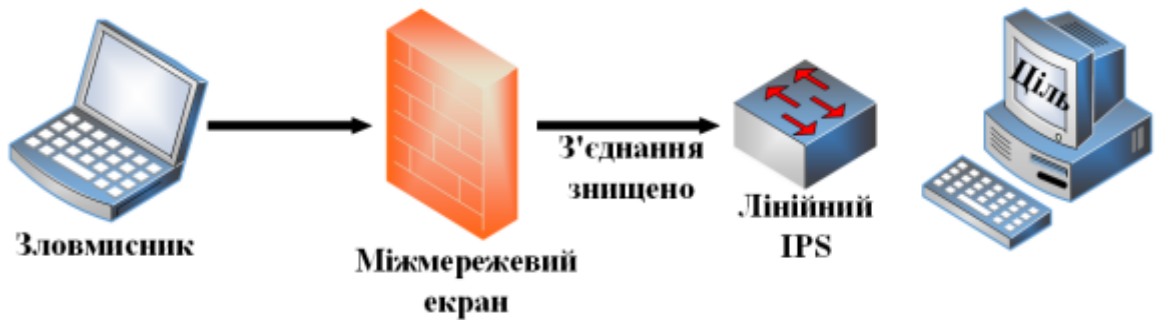


Рис. 2.3. Архітектура системи запобігання вторгнень IPS [31]

Системи IPS виконують наступні функції: «IPS сама зупиняє атаку - блокує шкідливу частину, дозволяючи неуразеній частині входити в систему та повідомляє адміністраторів безпеки про критичні події, що спостерігаються в системах; реагує на інциденти шляхом модифікації середовища безпеки для запобігання атакам; формує звіти» [5].

Отже, міжмережевий екран буде формувати доступ до портів і забезпечувати мінімальний захист, а IPS буде відслідковувати низькорівневі атаки і реагувати на зміну потоків трафіку.

Доповненням до інфраструктури мережевої безпеки можуть бути системи виявлення вторгнень (англ. Intrusion Detection System, IDS).

IDS – програмний або апаратний засіб, призначений для виявлення фактів несанкціонованого доступу в комп'ютерну систему (мережу), або несанкціонованого управління такою системою (рис. 2.4).

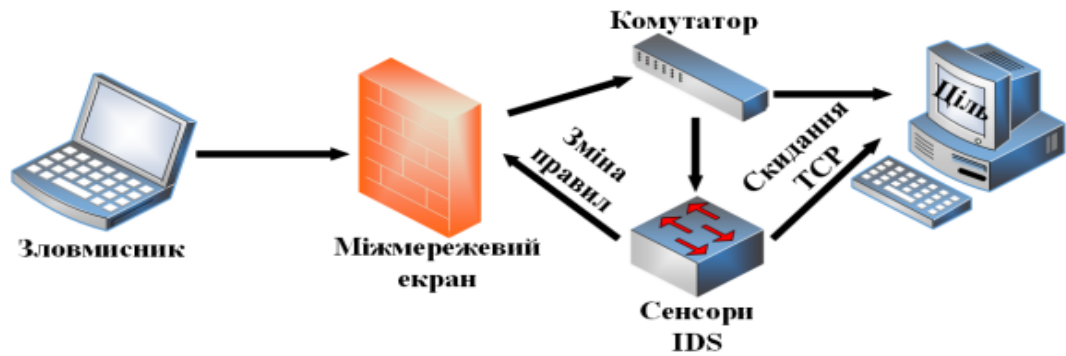


Рис. 2.4. Архітектура системи виявлення вторгнень IDS [31]

IDS здійснюють моніторинг та спостереження підозрілої активності. Вони можуть виявити атакуючих, які змогли обійти Firewall, і видати звіт про це адміністратору, який, у свою чергу, зробить подальші кроки щодо запобігання атаки [16].

У табл. 2.2 відображено рівні моделі OSI, на яких працює кожна з розглянутих систем -Firewall, IPS, WAF/

Таблиця 2.2

Робота систем комплексу на моделі OSI

Рівень моделі OSI	Firewall	Intrusion Prevention System	Web Application Firewall
2	+		
3	+	+	
4	+	+	
5		+	+
6		+	+
7			+

[Складено з джерела 38]

Таким чином, у комплексі системи захищатимуть інтернет-ресурси на всіх рівнях моделі OSI.

На рис. 2.5 показано схему запропонованої інтеграції системи. WAF реалізований в системі як зворотний проксі перед захищеними веб-серверами. Отримуючи дозволені міжмережвим екраном запити, IPS аналізує протоколи та зупиняє певні типи атак, після чого дані надсилаються до WAF, де

відбувається обробка атак функціоналу WAF, а також їх блокування.

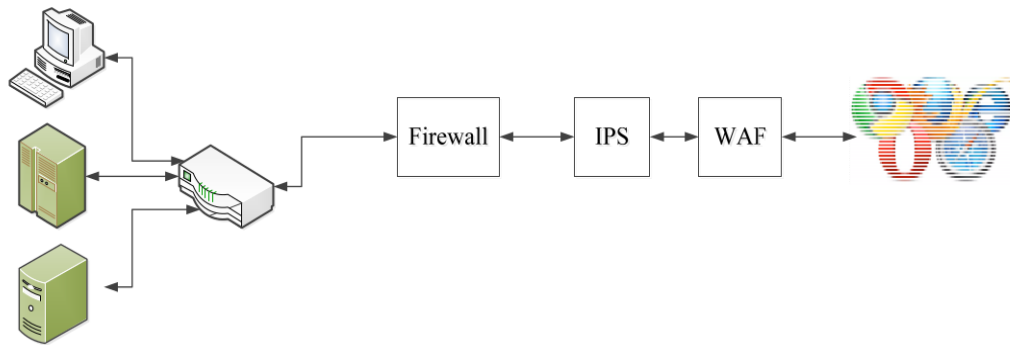


Рис. 2.5. Схема інтеграції систем з Web Application Firewall [38]

Відповіді з веб-сервера надсилаються назад до WAF, де вони перевіряються на предмет витоку даних. Після перевірки дані надходять до користувача. Використання цієї інтегрованої системи захисту інтернет-ресурсів у складі Firewall, IPS і WAF забезпечує на 30% більшу продуктивність, ніж використання звичайного WAF. Використання сучасних високопродуктивних міжмережевих екранів та IPS дозволить запобігти потраплянню шкідливих файлів у внутрішню мережу, що захищається, забезпечить додаткову безпеку та знизить ризики цілеспрямованих атак на ІТ-ресурси.

Цей комплекс дозволить підвищити безпеку будь-якого інтернет-ресурсу, зменшити навантаження на адміністраторів ІТ-систем та забезпечити більш ефективну обробку звернень легітимних користувачів інтернет-ресурсів.

2.3. Пентест як спосіб перевірки захищеності веб-ресурсів

При побудові системи захисту веб-додатків важливим є організаційний момент – необхідно знати, які вразливості існують у веб-додатках та політиці безпеки веб-сервера. «Налагодження та оцінку захищеності веб-ресурсів, а також отримання комплексу рекомендацій щодо посилення захисту дозволяє аудит. Найбільш поширеними підходами до аудиту технологій є тест на проникнення (пентест - демонстрація дій порушника) та «традиційний» аудит (аналіз конфігурації веб-сервера та використання сканера вразливостей)»

[11].

Тест на проникнення є оптимальним способом перевірки захищеності інформаційної системи за допомогою імітації спрямованих атак. Пентест дає можливість оцінити захищеність інформаційної системи від несанкціонованого впливу, використовуючи різні моделі вторгнень. Тест на проникнення для веб-додатків фокусує увагу виключно на оцінці рівня захисту веб-додатків. Процес складається з активного аналізу додатків і пошуку в них вразливостей, технічних помилок або інших проблем. Інформацію про всі слабкі місця відображається в підсумковому звіті.

Penetration testing - частина комплексного аудиту інформаційної безпеки. Під час аудиту аналізується велика частина організаційних і технічних заходів щодо забезпечення інформаційної безпеки.

Перевіряють налаштування систем захисту, виявляють наявність вразливостей у прошивках обладнання, системному ПЗ і призначеному для користувача софті, а також вивчають реакцію працівників на традиційні хитрощі, включно з таргетованим фішингом, а іноді й фізичним доступом неавторизованого персоналу. Разом ці перевірки дають змогу виявити слабкі місця в системі інформаційної безпеки підприємства.

Зовнішнє тестування комп'ютерних систем на проникнення вбиває відразу двох зайців. По-перше, така незалежна оцінка дає змогу уникнути фінансових і репутаційних втрат компаніям. Ця послуга особливо затребувана в організаціях, які вже зіткнулися з інцидентами у сфері ІБ. По-друге, окремі види діяльності вимагають підтвердження відповідності певним стандартам безпеки та законодавчим актам.

Процес тестування на проникнення імітує реальну хакерську атаку і включає в себе кілька етапів:

- збір інформації про мету;
- застосування соціального інжинірингу;
- визначення точок входу в мережу;
- виявлення і використання вразливостей;

- підвищення привілеїв в атакованій системі;
- складання звіту та рекомендацій.

Зазвичай тестування на наявність вразливостей починається із зовнішньої мережі, а потім тестуються внутрішні сервіси.

Найвідомішими методиками тестування на проникнення є:

OSSTMM (Open Source Security Testing Methodology Manual) - універсальний стандарт, бо може бути базовим керівництвом під час проведення тестування на проникнення [14];

NIST SP800-115 - NIST Special Publications 800 Series - стандарт інформаційної безпеки, розроблений National Institute of Standards and Technology. Це технічне керівництво, яке можна застосовувати для перевірки рівня інформаційної безпеки організацій різних сфер, це одна з обов'язкових методологій проведення пентесту, яку використовують професійні аудиторські компанії [14];.

OWASP (Open Web Application Security Project) - відкрита інтернет-спільнота, що пропонує найбільш вичерпну методологію для тестування додатків, сайтів та API. Одна з головних переваг OWASP у тому, що методологія описує тестування на кожній стадії життєвого циклу розроблення застосунків: визначення вимог, проєктування, розроблення, впровадження та підтримка [14];

ISSAF (Information System Security Assessment Framework) — документ, в якому наведені докладні рекомендації з тестування на проникнення. Містить утиліти, якими можна провести пентест, вказівки щодо їхнього використання, а також детально роз'яснено, які результати та за яких параметрів можна отримати в результаті тестування [14];

PTES (Penetration Testing Methodologies and Standards) - це рекомендації для проведення базового пентесту, а також кілька розширених варіантів тестування, для організацій з підвищеними вимогами до інформаційної безпеки. Одна з переваг PTES у тому, що він дає докладний опис цілей і очікувань від пентесту [14].

Для проведення пентестів можна використовувати одну з методологій, проте досвідчені компанії-аудитори застосовують одразу кілька. Вибір залежить від конкретної організації, її бізнес-процесів і процесів інформаційної безпеки.

Сканування необхідно проводити регулярно, оскільки в операційних системах, на яких розміщені веб-сервери, постійно виявляються нові вразливості. Існують нові методи атак на веб-додатки. Удосконалюються сканери вразливостей, трапляються збої в політиці безпеки.

За результатами аудиту надаються рекомендації щодо модернізації системи захисту мережі, які дозволяють усунути небезпечні вразливості і таким чином підвищити рівень захисту інформаційної системи від дій зовнішнього злоумисника.

Звичайно, найкращим методом боротьби з уразливостями є виправлення коду веб-додатку та правильне налаштування параметрів політики безпеки.

РОЗДІЛ 3. ЗАХИСТ ВЕБ-РЕСУРСІВ УСТАНОВИ

3.1. Способи захисту веб ресурсів від загроз

Наявність у веб-ресурсах уразливостей, які попередньо були розглянуті у п.1.1, створюють певні загрози, які зумовлюють пошук способів їх уникнення та заходів щодо забезпечення безпеки веб-ресурсів. Зокрема, веб-ресурси потребують захисту від SQL-ін'єкцій, XSS-атак, CSRF-атак, Broken authentication атаки, атак, що використовують неправильні конфігурації, атак щодо витоку критичних даних, атак, що зумовлені відсутності контролю рівня доступу на функціональному рівні.

Проведені дослідження свідчить, що:

- *для захисту від SQL-ін'єкцій, потрібно* використовувати останні доступні оновлення для систем, а також оновлювати всі компоненти, які використовуються для створення програмних продуктів, включаючи плагіни, бібліотеки, платформи на яких розробляється додаток, програмне забезпечення веб-сервера і баз даних. Наприклад, якщо веб-ресурсу потрібно тільки зчитати дані з бази даних за допомогою операторів SELECT, недопустимим є надання даному обліковому запису інших привілеїв на такі операції як UPDATE, INSERT або DELETE.

Потрібно налаштувати необхідні звіти про помилки та їх обробку в коді веб-додатку і на сервері так, щоб повідомлення про помилки баз даних ніколи не були відправлені клієнту в веб-браузер. Такі дії потрібні, щоб злоумисники не мали можливості використати технічні деталі повідомлень про помилки, щоб скорегувати свої запити для успішної експлуатації ресурсу у власних цілях [36].

Щоб запобігти **SQL-ін'єкціям**, потрібно використовувати параметризовані запити, які мають підтримку в більшості мов програмування, що використовуються при розробці веб-додатків.

Для прикладу:

*SELECT * FROM tableName WHERE column ='parameterName'.*

Якщо зловмисник змінить значення “parameterName” на “OR '1'='1'”, то такий запит набере вигляду:

*SELECT * FROM table WHERE column = " OR '1'='1'”.*

Оскільки результатом даного виразу '1' = '1' завжди буде “true”, зловмисник отримає доступ до даних з усіх таблиць бази даних. Цей прийом дозволяє виконати будь який довільний запит до БД, додавши в кінець даний вираз SQL. за допомогою параметризації можна легко усунути вразливість цього запиту. Наприклад, для програми, що написана з використанням PHP і MySQL, він має наступний вигляд:

```
$stmt = $pdo ->prepare ('SELECT * FROM tableName WHERE columnName
=:value');
```

```
$stmt->execute (array ('value' => $parameterName)).
```

Також рекомендовано використовувати «Escape-символи» – вони дозволяють ігнорувати спеціальні символи. «Escape-символи» – це засіб вказати ядру MySQL, що одиночні лапки – це не символ, який завершує рядок, а це частина самого рядка. Для того, щоб MySQL знав, що даний запит безпечний, потрібно додати у відповідному місці символ зворотної косої межі.

Також можна використовувати брандмауер (WAF) для веб-додатків, які роблять запити до бази даних. WAF допомагає ідентифікувати спроби SQL інжекції, а також запобігти спробам додавання інжекції SQL у додаток [39].

- для захисту від XSS-атаки, потрібно використання атрибуту HttpOnly. Мета цього атрибуту в тому, що в ситуації, коли сервер налаштовує файли куки, цей атрибут є гарантом надійності, адже можуть додаватися деякі інші атрибути, які можуть гарантувати, що файли куки будуть надійно захищені від різного роду шкідливих запитів.

Наприклад:

JS:Set-Cookie: [ім'я] =[значення].

Атрибут *HttpOnly* є гарантом того, що відповідь, в якій будуть зберігатися файли куки, буде відправлена на ту ж адресу, де їх було створено. Кодування вихідних змінних. Іноді бувають ситуації, при яких веб-додатку треба бути впевненим, що всі результуючі дані на веб-сторінці захищені, наприклад, при атаці XSS.

Шифрування вихідних даних змінюється розміткою так званих «HTML альтернативних уявлень», які спеціалісти називають «entities». Під час використання «entities», веб-браузер показує результуючі об'єкти та має можливість не запускати їх відразу.

Наприклад, `<hello>` перетворюється в `& lt; hello & gt;`. В ситуації, коли браузер розпізнає об'єкти, вони мають змогу конвертуватись в HTML-код і можуть бути показані, проте не запущені. Наприклад, якщо хакер вставить в код `<hello>alert(«Hello»)</hello>` в HTML-код сервер повинен повернути `<hello>alert(«Hello «)</hello>` [36].

Під час виконання браузером завантаження зашифрованого скрипта, веб-браузер конвертує цей скрипт в вище зазначений код `<hello>alert(«Hello «)</hello>` та показує результат скрипта, як деяку частину сторінки, проте найцікавіше те, що скрипт запущеним не буде.

Якщо у юзера, який пройшов аутентифікацію, навіть є куки файл, який надає змогу користувачу автоматично здійснювати вхід в систему, всеодно він буде обмежений на певних сторінках до тих пір, поки він не введе логін та пароль. Цей підхід є корисним в боротьбі з атаками XSS, адже він не дає можливості зловмиснику XSS перехопити або захопити сеанс [36].

- для захисту від атаки CSRF (Cross-site request forgery, міжсайтова підробка запиту) потрібно забезпечення того, щоб GET-запити не мали побічних ефектів, і забезпечення того, щоб не-GET-запити могли виходити тільки з клієнтського коду.

Щоб це забезпечити необхідно запровадити передачу стану уявлення (REST). REST - це набір принципів проектування, які привласнюють певні

типи дій (перегляд, створення, видалення, оновлення) різним HTTP-verbs. REST-фул (RESTful) дизайни будуть підтримувати код в чистому, підтримуваному стані і допоможуть масштабувати сайт. Більш того, REST наполягає на тому, щоб запити GET використовувалися тільки для перегляду ресурсів. Відсутність побічних ефектів в GET-запитах обмежить можливу атаку, яка може бути завдана створеними URL-адресами - зловмиснику доведеться багато і старанно працювати для створення шкідливих POST-запитів.

Використання Anti-Forgery Tokens дають можливість визначати відправника запиту. Навіть якщо дії по редагуванню обмежені не-GET-запитами, ви не повністю захищені. POST-запити як і раніше можна відправляти на ваш сайт із сценаріїв і сторінок, розміщених на інших доменах. Для гарантії оброблення тільки дійсних HTTP-запитів, необхідно включати секретний і унікальний токен в кожен HTTP- відповідь, і щоб сервер перевіряв токен при його поверненні в повторних запитів, що використовують метод POST (або будь-який інший метод). крім GET, насправді.

Кожен раз, коли сервер відображає сторінку, яка виконує конфіденційні дії, він повинен записати токен захисту від підробки в приховане поле форми HTML. Цей токен потрібно включити в POST-запит на сервер (через форму або AJAX-виклики). Сервер повинен перевіряти токен, коли він повертається в повторних запитах, і відхиляти будь-які виклики з недійсним токеном[2]. Токенами проти підробки зазвичай є випадкові числа, які зберігаються в файлах cookies або на сервері. Сервер порівнює токен з вхідного запиту, зі значенням, збереженим у файлі cookie. Якщо значення ідентичні, сервер прийме HTTP-запит дійсний[36].

Перевіряти, що файли Cookies відправлені з атрибутом Cookie SameSite. Команда Google Chrome розширила заголовок Set-Cookie новим атрибутом «Same- Site», щоб запобігти CSRF. Атрибут cookie з одним сайтом дозволяє розробникам інструктувати веб-переглядачі здійснювати контроль: чи

надсилаються файли cookie разом із запитом, ініційованим доменами сторонніх виробників.

Встановлення атрибута «Same-Site» у файлі cookie досить просто:

Set-Cookie: CookieName = CookieValue;

SameSite = Lax;

Set-Cookie:CookieName = CookieValue;

SameSite = Strict;

Значення «Strict» означає, що будь-який запит, ініційований доменом третіх сторін у вашому домені, матиме cookie, видалене веб-переглядачем.

Це найбезпечніше налаштування, оскільки запобігає спробам сайтів хакерів виконувати шкідливі дії з-під сеансу користувача.. Значення Lax дозволяє GET-запиту (і лише їм) від стороннього домену до вашого домену приєднати файли cookie. При цьому налаштуванні користувачеві не потрібно буде входити знову на ваш сайт, якщо за посиланням з іншого сайту (скажімо, результатів пошуку Google).

- **для захисту від *Broken authentication* - атаки, потрібно** використання безпечно вбудованого менеджера сеансів на стороні сервера, який генерує новий випадковий «session ID» з високою ентропією після входу в систему.

Для уникнення ненадійності та незахищеності від небезпек, ідентифікатори сеансів не повинні розташовуватись в URL-адресі. Також вони повинні ставати недійсними після виходу з системи, бездіяльності впродовж деякого часу або під час абсолютних тайм-аутів. Існують спеціальні налаштування для заборони на сесійну передачу через URL.

Наприклад:

js.insession.use_trans_sid=0;

session.use_only_cookies=1;

js_flag

session.use_trans_sid

Offjs_flag

`session.use_only_cookies=2.`

Зазвичай, одною з найбільш складних проблем в розробці веб-додатку є надійність пароля. Правила надійних паролів робить практично неможливим вгадування пароля за допомогою різного роду засобів. Щоб пароль був складним, його треба створювати з урахуванням верхнього та нижнього регістрів. Мінімальна кількість символів в паролі повинна бути не менше 10 символів. Основні правила формування пароля, згідно [39]:

- „ – мінімум один рядковий символ (a-z);
- мінімум один титульний символ (A-Z);
- мінімум один спецсимвол (НЕ пунктуація);
- мінімум одна цифра (0-9);
- не більше 2 однакових символів підряд(изменено);
- пароль повинен відповідати (як мінімум) 3 з 4 правилам складності”.

Серверний механізм зміни пароля треба створювати таким чином, щоб він був мінімального рівня складності та, щоб він мав сенс, як для програми, так і для користувачів. Зазвичай, при зміні паролю на серверній частині додатку співвідносять старий та новий паролі та вимагають хоча б мінімальної зміни топології. Передача паролів тільки через TLS або інший надійний канал. Систему аутентифікації потрібно будувати так, щоб сторінка входу та, безпосередньо, аутентифікація були доступні лише через TLS.

Аутентифікаційна сторінка зі входом повинна працювати через TLS-канал або, в деяких випадках, через інший надійний канал. Потреба повторної аутентифікації для важливих функцій. Іноді виникають ризики перехоплення сеансів. Для зниження цих ризиків та ризиків CSRF, гарною практикою є вимагання від користувача повторного введення даних облікового запису перед тим, як оновлювати конфіденційні дані облікового запису, такі, як електронна пошта, телефон, пароль або якісь важливі операції [38].

- Для захисту від атак невірно налаштованої конфігурації, потрібно

виконати наступні кроки:

- зменшувати кількість вразливих місць за допомогою циклічних процесів;
 - оновлювати програмне забезпечення, і тримати його в актуальному стані;
 - відключати облікові записи за замовчуванням і регулярно оновлювати паролі;
 - створювати захищену архітектуру додатків з шифруванням даних, що мають будь-яку конфіденційну інформацію;
 - переконатися, що параметри безпеки в бібліотеках налаштовано вірно;
 - виконувати регулярні перевірки та використовуйте інструменти для ідентифікації вразливостей в системі;
 - використовувати ту ж конфігурацію для створення, розробки і розгортання програми, оскільки відмінності це ворота для неправильних конфігурацій;
 - максимально автоматизувати систему, де це можливо, щоб уникнути помилок, що пов'язані з людським фактором;
 - переконатися, що додаток використовує останні оновлення, включаючи операційні системи, програми, системи керування базами даних та веб- сервери додатків;
 - перевіряти, чи немає облікових записів, які можуть використовуватися за замовчуванням і чи не були змінені їхні паролі. Облікові записи за замовчуванням зазвичай є джерелом проблем;
 - перевірити, чи існують непотрібні або потенційно небезпечні функції. Наприклад, програма може мати функцію налагодження, яка дозволить зломисникам обійти автентифікацію для доступу до секретної інформації;
 - проводити планові перевірки безпеки додатку та сканування вразливостей, щоб своєчасно виявляти неправильні конфігурації [38].
- Для захисту від атаки витоку критичних даних, потрібно:**
- Забезпечення шифрування даних та впровадження перевіреної техніки

шифрування. Однією з найважливіших аспектів будування додатку є безпека конфіденційних даних. Саме для цього існує окрема галузь - шифрування. Проте з шифруванням треба бути обережним, обов'язково дотримуватись деяких правил. Також потрібно шифрувати дані та визначати їх доступність, шифрувати інформацію в формі, що зберігається або в транзиті.

Важливо, насамперед, зберігати конфіденційну інформацію закодованою або зашифрованою увесь час, адже такі важливі дані не повинні бути в загальному доступі і в ніякому випадку не повинні передаватися в чистому текстовому форматі. Зловмисникам дуже легко буде дістати конфіденційну інформацію, якщо вона буде у вигляді відкритого тексту.

Визначення даних, потребуючих додаткового захисту та обмеження доступу лише для групи підтверджених користувачів шляхом шифрування на основі ключів.

Ситуація 1: Шифрування кредитної карти. Додаток шифрує номери кредитних карт у базі даних за допомогою автоматичного шифрування баз даних. Проте, є вірогідність, що додаток автоматично буде розшифровувати інформацію після її отримання, яке може призвести до отримання номерів кредитних карток в прозорому тексті за допомогою SQL-ін'єкцій. Гарною практикою є розшифровування інформації за допомогою закритого ключа, тобто, в системі є зашифровані номери кредитних карток, шифрування яких здійснюється за допомогою відкритого ключа. Використання шлюзів безпечної аутентифікації. Взагалі, на даний момент, використовують в додатках протокол HTTPS (SSL / TLS) - це захищений протокол, за допомогою якого конфіденційні дані, що передаються між веб-браузером та сервером залишаються приватними. За допомогою SSL можна створювати пару публічних або приватних ключів для того, щоб передавати інформацію між браузером та сервером.

Ситуація 2: SSL може не використовуватись для сторінок, які пройшли підтвердження та аутентифікацію. Шахрай просто займається тим, що спостерігає за мережевим трафіком, як приклад у відкритій бездротовій

мережі та забирає звідти cookie сесії користувача. Уже з отриманими cookie можна здійснювати злочинні дії, а саме повторити файл з даними та перехопити сеанс користувача які допомагають отримати доступ до особистих даних. Впровадження хорошого алгоритму хешування паролів. Зловмисники можуть скористатися не надійним алгоритмом для хешування паролів, щоб отримати конфіденціальну інформацію, яка збережена на сервері додатків чи веб-сервері. Для того, щоб пароль був добре захищеним є необхідність у використанні криптографічних хеш-функцій.

Ситуація 3: використання несолідованих хешів для зберігання паролів у базі даних для кожного користувача. Якщо файл пошкоджено, то зловмисник може отримати доступ до файлу паролів. Виставлення можна зробити за допомогою веселки, для хешів які не солодкі з попереднім розрахуванням. Також необхідно зазначити, що шифрування також може бути не ідеальним, використання застарілих чи слабких алгоритмів шифрування не є гарантією безпеки, а лише надає ілюзію захисту. Така ж ситуація і з простими хешами, що можна повернути. Важливо постійно оновлювати алгоритми шифрування, оскільки сучасні та сильні алгоритми, протоколи, та ключі, забезпечують високий рівень безпеки.

- **Захист від атаки відсутності контролю рівня доступу на функціональному рівні.** Одна з необхідностей програм, це потреба у послідовному та простому модулі авторизації, який необхідно викликати з усіх бізнес-функцій. Як правило такий захист виконується за допомогою одного або декількох компонентів, зовнішніми до коду програми.

Для безпеки рекомендовано за замовчуванням застосовувати правило заборони, а також заборона доступу до всіх функцій програми, а вже потім надати доступ користувачам чи програмам, що його потребують. Але хоч доступ і надано, необхідно постійно його перевіряти під час звертання до ресурсу, що являє собою перевірку від дійсних користувачів, які авторизовані. Необхідно використовувати списки контролю доступу та перевірку на наявність доступу та аутентичності.

Необхідно використовувати по мінімуму привілегій та надавати доступ тільки при необхідності. Загальний доступ не повинен бути доступним, необхідно зазначити ролі для зменшення кількості прав у користувачів. Приховувати деякі елементи через неясність не є гарним вирішення проблеми безпеки, роблячи невидимими або ж якимось іншим чином, ховаючи посилання на функції інтерфейса чи кнопки, бо випадково хтось може натрапити на прихований функціонал. Окрім того, треба брати до уваги той факт, що зломисники не будуть користуватися юзер-інтерфейсом як звичайний користувач. Скоріш за все вони будуть шукати інші шляхи відправлення запитів для того щоб надсилати їх безпосередньо, щоб отримати респонс від самої бази даних. Потрібно перевірити увесь наявний функціонал веб-додатку та URL адреси, використовуючи акаунт користувача з найнижчим рівнем доступу.

Пересвідчіться що ці акаунти не можуть отримати доступ до тих частин програми, до яких вони не мають прав доступу. Зараз є певні застосунки, що надають функціонал перевірки. Вони можуть порівнювати користувачів із правами адміністратора і звичайних користувачів. Після цієї перевірки надається інформація щодо частин програми, до яких мають доступ звичайні користувачі, хоча і не повинні.

Майже всі веб-застосунки не відображають кнопки чи URL посилання для тих користувачів, які не мають прав доступу, проте варто все ж таки перевірити детальніше, бо такий метод контролю не дає стовідсоткового результату. Найкращим рішенням буде перевірка методів контролерів і методів бізнес-логіки.

3.2. Основні засоби і способи захисту Python веб-додатків

Python – одна з найбільш популярних мов програмування, яку цінують за лаконічність, універсальність і стабільність, на якій за короткий термін можна створити веб-додатки для бізнесу, розробити серверну частину сайтів.

Python використовується для вирішення завдань на платформах iOS, Android,

Windows, серверних ОС. Тому розглянемо можливі варіанти захисту веб-додатків, створених на Python.

Існує два можливі варіанти захисту Python веб-ресурсів:

- Захист через шифрування даних (Sentinel Envelope за допомогою Sentinel Data File Protection (DFP))
- Експорт модулів через Cython (<https://cython.org/>) з подальшим захистом через Sentinel Envelope

Перший варіант захищає Python веб ресурс, комбінуючи Sentinel Envelope із Sentinel Data File Protection (DFP). Таким чином, захищається безпосередньо інтерпретатор Python, якому на рівні захисту вказується з якими файлами він повинен працювати як із захищеними (зашифрованими). Такі файли інтерпретатор попередньо автоматично розшифровуватиме і лише потім виконуватиме. Незашифровані файли працюватимуть так само, як і раніше. Самі *.com. модулі, що потребують захисту, шифруються за допомогою інструмента Sentinel Data File Protection (DFP).

Другий варіант має на увазі захист Python додатків, комбінуючи Cython (<https://cython.org/>) із Sentinel Envelope. Спочатку необхідно модулі Python, що вимагають захисту, перевести в C-Code за допомогою Cython і потім скомпілювати їх у модулі *.pyd/*.so, які згодом захищаються за допомогою Sentinel Envelope.

Другий метод забезпечує більш високий рівень безпеки, оскільки додатковий етап компіляції знижує рівень абстракції коду і дозволяє Sentinel Envelope захищати веб ресурс як код, а не лише дані, що дозволяє застосовувати складніші механізми захисту.

З іншого боку, другий метод є складнішим у налаштуванні, так як вимагає додатково «прошарку» у вигляді Cython і Скомпілятор. Обидва методи не можуть захистити стартовий скрипт програми, а лише його модулі Python. Тому рекомендується помістити фактичну точку входу розроблюваної програми в Python і використовувати тільки стартовий скрипт для виклику модуля [41].

Захист програмного забезпечення Python за допомогою Sentinel Data File Protection складається з трьох етапів [41]:

- збереження програмного забезпечення у модулі байт-коду.
- шифрування файлів за допомогою утиліти Sentinel dfcrypt.

Захист Python веб ресурсу складається з таких етапів:

- Захисту інтерпретатора Python, увімкнувши в налаштуваннях захисту опцію «Enable data file protection (Data Protection Utility) = Version 2» для захищеного інтерпретатора із зашифрованими файлами даних. Захищена програма може розповсюджуватися шляхом упаковки захищеного за допомогою Envelope інтерпретатора разом із зашифрованими файлами (наприклад, з використанням pyinstaller).

Приклад є простим Python додатком командного рядка, який складається зі сценарію запуску (main.com) і 3х модулів (moduleA.com, moduleB.com, moduleC.com). Цей приклад був написаний так, щоб працювати однаково в обох версіях Python2 та Python3. Директорія data_file_protection містить:

- build_dfp_protected_python2_app.bat Цей сценарій захищає та пакує приклад програми для Python2 для Windows, з використанням: коду розробника DEMOMA, Sentinel Envelope, утиліти шифрування файлів даних для Sentinel LDK (dfcrypt) та pyinstaller.

- envelope_python2.prjx Файл проекту Sentinel Envelope, в якому вказані параметри Envelope для захисту інтерпретатора Python2 (python27.dll) для наведеного вище сценарію.

- build_dfp_protected_python3_app.bat Цей сценарій захищає та пакує приклад програми для Python3 для Windows, з використанням: коду розробника DEMOMA, Sentinel Envelope, утиліти шифрування файлів даних для Sentinel LDK (dfcrypt) та pyinstaller.

Ключ «--dfp» активує Data File Protection, що дозволяє захищеному інтерпретатору Python виконувати зашифровані модулі Python. У Windows потрібно використовувати Sentinel Envelope та активувати Data File Protection Version 2.

Захист Python програм проводиться, шляхом увімкнення Data Protection Version 2, копіюються вихідні файли Envelope у центр: dist/. Щоб розгорнути захищену програму, просто потрібно скопіювати папку dist/ на цільовий комп'ютер. Захищений веб ресурс можна запустити, виконавши команду: dist//.

Наш захист веб ресурсу буде створюватися через модуль NGIUX, який дозволяє невеликим config, перехватувати деякий трафік, який йде на сервер і його деяким чином змінювати:

Модуль ngx_http_limit_req_module модуль (0.7.21) використовується для обмеження швидкості обробки запиту за визначений ключ, зокрема, швидкість обробки запитів, що надходять з однієї IP-адреси. Обмеження здійснюється за допомогою методу «дірявого відра».

Працює це все наступним чином: Наприклад сайт працює в нас на 5000-м порту, зазвичай сайти, зазвичай сайти слухають 80-ті чи 443-ті, в нас буде запускатися на 5000-му, що ми можемо зробити рис. 3.1

```
(base) nlevashov@MacBook-Pro-Nick ~ % cd /usr/local/etc/nginx/servers
(base) nlevashov@MacBook-Pro-Nick servers % cat server.conf
limit_req_zone $binary_remote_addr zone=mylimit:10m rate=20r/s;

server {
    listen 4000;
    server_name 127.0.0.1;

    location / {
        proxy_pass http://127.0.0.1:5000;
    }

    location /auth {
        limit_req zone=mylimit burst=100;
        proxy_pass http://127.0.0.1:5000;
    }
}

(base) nlevashov@MacBook-Pro-Nick servers %
```

Рис. 3.1.Код захисту для веб-ресурсу

Ми можемо не виставляти до загального доступу вид на 5000-й порт, а тільки на 4000-й, тобто таким чином замаскувати його, і скористатися ngx, для того щоб всі запити приймав сайт, на 4000-му порті, далі проводив деяку

валідацію, і після цього пересилав весь трафік уже до нашого справжнього сайту. Тобто ми за допомогою програми створюємо сайт дублер, для того щоб всі атаки він прийняв на себе, а потім відфільтровував інформацію, та пересилав на справжній. Тобто це такий реверс проксі, який приймає всі запити, валідує їх, і в залежності від того куди вони йдуть посилає його на справжній сайт, чи якщо він створює небезпеку відхиляє її.

Тобто, таким чином, утворює firewall. Ця конфігурація не дає прямого доступу до сайту, при умові, що ми 5000-й порт не відкриваємо на загальний доступ.

Розберемо детально, що дозволяє захистити сайт від втручання. Першим рядком, а саме :

```
[(base) nlevashov@MacBook-Pro-Nick~ % cd /usr/local/etc/nginx/servers
(base) nlevashov@MacBook-Pro-Nick servers % cat server.conf
limit_req_zone $binary_remote_addr zone=mylimit: 10m rate=20r/s;

server {
    listen 4000;
    server_name 127.0.0.1;
    location / {
        proxy_pass http://127.0.0.1:5000;
    }
    location /auth {
        limit_req zone=mylimit burst=100;
        proxy_pass http://127.0.0.1:5000;
    }
}
(base) nlevashov@MacBook-Pro-Nick servers %
```

За допомогою виділенного рядка, ми виділяємо деяку зону, тобто всіх користувачів, ми розділяємо на декілька зон, при чому у якості ідентифікатора, ми використовуємо \$binary_remote_addr, це такий ір адреса, з якого приходить користувач, якщо користувач приходить з різних ір, тоді ми вважаємо, що це різні користувачі, тому розподіляємо їх по різним зонам. Якщо навпаки з одної IP-адреси, ми вважаємо, що це той самий користувач, який буде знаходитись в одній зоні. І кількість запитів, які можуть приходити з однієї ір-

адреси, ми обмежуємо 20-ма запитами в секунду, за допомогою `rate=20r/s`.

`Mylimit: 10m` – обмежує пам'ять для IP-адрес, після їх заповнення їх потрібно очищати. Таким чином, конфігурується зона.

Потім на певному `location`, ми цю зону застосовуємо:

```
[(base) nlevashov@MacBook-Pro-Nick~ % cd /usr/local/etc/nginx/servers
(base) nlevashov@MacBook-Pro-Nick servers % cat server.conf
limit_req_zone $binary_remote_addr zone-mylimit: 10m rate=20r/s;

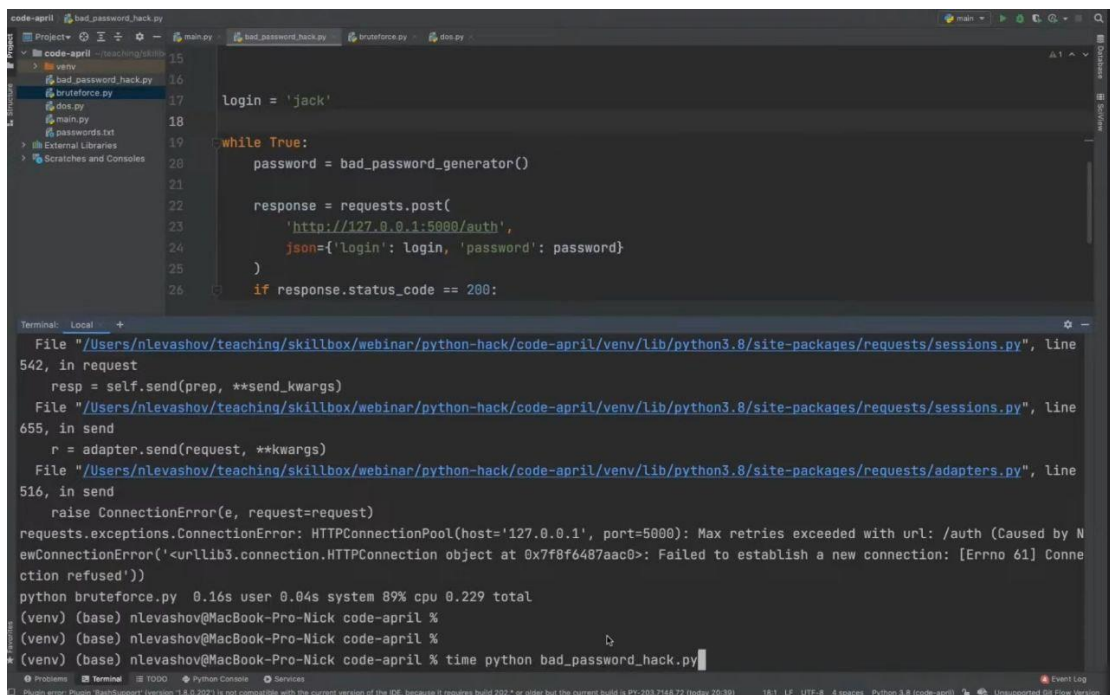
server {
    listen 4000;
    server_name 127.0.0.1;
    location / {
        proxy_pass http://127.0.0.1:5000;
    }
    location /auth {
        limit_req zone=mylimit burst=100;
        proxy_pass http://127.0.0.1:5000;
    }
}
(base) nlevashov@MacBook-Pro-Nick servers %\
```

Параметр `burst=100` визначає довжину черги повідомлень, які можуть прийти від одного користувача. Він слугує певним фільтром інформації. Тобто це працює, якщо ми відправили 19 запитів, всі вони пройшли, далі йде 20-й - все добре, так як ми задовільнили виходящу вимогу `rate=20r/s`. Далі ми переходимо на сайт чи сервер `{ proxy_pass http://127.0.0.1:5000 }`, без зауважень та проблем, тобто ми проходимо систему безпеки.

Якщо ми відсилаємо 21-й запит в ту ж секунду, програма розуміє, що задана вимога `rate=20r/s` – порушена, і він ставить її в чергу, таким чином, в черзі утворюється перший запит. Надсилати 22-й запит не має сенсу так як ми недочекались відповіді на 21-й. Ми чекаємо секунду, на яку нам знов відкривається вікно на 20 запитів, і поки ці запити не будуть відпрацьовані, програма чекає секунду, потім час минає, цей запит йде на сервер, і надсилає відповідь користувачу, який створює перебор.

Користувач надсилає 22-й запит, так як після 2-го секунда обмеження $rate=20r/s$ – задовольняється. Можна ще зробити 18 запитів, вони всі проходять, і 41-й – стає знов в чергу. Якщо ми надсилаємо за допомогою мультитрейдингу, чи якогось синхронного коду, мультипроцесорного коду, з одної ір-адреси декілька запитів, то ми недочекаємось поки один запит відпрацює, ми відсилаємо наступний. Таким чином, якщо ми відправимо більше 100 запитів одночасно, 101-й вже в чергу не поміститься і отримає 503-ю помилку. А всі інші сто, будуть чекати, коли закінчиться чергова секунда, і почнеться наступна. Таким чином, ми можемо захиститися від потоку запитів, і його призупиняти.

Щоб продемонструвати ефективність даного методу, можна провести перезапуск певної атаки рис.3.2.



```

code-april  bad_password_hack.py
Project  main.py  bad_password_hack.py  bruteforce.py  dos.py
code-april  ~teaching/skillbox
> venv
  bad_password_hack.py
  bruteforce.py
  dos.py
  main.py
  passwords.txt
  External Libraries
  Scratches and Consoles
15
16
17 login = 'jack'
18
19 while True:
20     password = bad_password_generator()
21
22     response = requests.post(
23         'http://127.0.0.1:5000/auth',
24         json={'login': login, 'password': password}
25     )
26     if response.status_code == 200:

```

```

Terminal: Local
File "/Users/nlevashov/teaching/skillbox/webinar/python-hack/code-april/venv/lib/python3.8/site-packages/requests/sessions.py", line 542, in request
    resp = self.send(prepared_request, **send_kwargs)
File "/Users/nlevashov/teaching/skillbox/webinar/python-hack/code-april/venv/lib/python3.8/site-packages/requests/sessions.py", line 655, in send
    r = adapter.send(request, **kwargs)
File "/Users/nlevashov/teaching/skillbox/webinar/python-hack/code-april/venv/lib/python3.8/site-packages/requests/adapters.py", line 516, in send
    raise ConnectionError(e, request=request)
requests.exceptions.ConnectionError: HTTPConnectionPool(host='127.0.0.1', port=5000): Max retries exceeded with url: /auth (Caused by NewConnectionError(<urllib3.connection.HTTPConnection object at 0x7f8f6487aac0>: Failed to establish a new connection: [Errno 61] Connection refused))
python bruteforce.py 0.16s user 0.04s system 89% cpu 0.229 total
(venv) (base) nlevashov@MacBook-Pro-Nick code-april %
(venv) (base) nlevashov@MacBook-Pro-Nick code-april %
(venv) (base) nlevashov@MacBook-Pro-Nick code-april % time python bad_password_hack.py

```

Рис. 3.2. Демонстрація серверу на який буде застосовуватись атака

Якщо маємо певного користувача адміна, будемо проводити атаку з метою отримання його паролю.

Якщо раніше на 4000-й наша атака займала менше секунди, коли ми підемо на захищений порт, тоді програма буде працювати в 10 разів довше, так

як до цього можна було 200-300 відсилати запитів в секунду, зараз ми обмежені $rate=20r/s$.

Цей метод демонструє збільшення часу на отримання коду, шляхом перебору, тому навіть взлом 3-х символного коду, займе досить тривалий час (рис.3.3).

The screenshot shows a code editor with a file named `bad_password_hack.py`. The script contains the following code:

```

15
16
17 login = 'jack'
18
19 while True:
20     password = bad_password_generator()
21
22     response = requests.post(
23         'http://127.0.0.1:4000/auth',
24         json={'login': login, 'password': password}
25     )
26     if response.status_code == 200:
27         while True:

```

The terminal window below the code editor shows the following output:

```

requests.exceptions.ConnectionError: HTTPConnectionPool(host='127.0.0.1', port=5000): Max retries exceeded with url: /auth (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x7f8f6487aac0>: Failed to establish a new connection: [Errno 61] Connection refused'))
python bruteforce.py 0.16s user 0.04s system 89% cpu 0.229 total
(venv) (base) nlevashov@MacBook-Pro-Nick code-april %
(venv) (base) nlevashov@MacBook-Pro-Nick code-april %
(venv) (base) nlevashov@MacBook-Pro-Nick code-april % time python bad_password_hack.py
SUCCESS jack black
python bad_password_hack.py 0.51s user 0.10s system 67% cpu 0.903 total
(venv) (base) nlevashov@MacBook-Pro-Nick code-april % time python bad_password_hack.py
SUCCESS jack black
python bad_password_hack.py 0.50s user 0.08s system 6% cpu 9.138 total
(venv) (base) nlevashov@MacBook-Pro-Nick code-april % time python bad_password_hack.py
SUCCESS jack black
python bad_password_hack.py 0.53s user 0.08s system 6% cpu 8.970 total
(venv) (base) nlevashov@MacBook-Pro-Nick code-april %

```

Рис.3.3.Демонстрація потоку запитів на сервер, який займає довше часу

Цей ефективний метод захисту, дозволяє зменшувати потоки від поганих користувачів, але, при цьому, не заважати звичайним. Якщо хакер вирішив “зломати” якийсь веб-ресурс, але на його провайдері є хороший користувач – програма захисту хорошому користувачу не зашкодить. А хакеру не вдасться здійснити повний перебір, через брак часу.

ВИСНОВКИ

Безпека веб-ресурсів установи є важливою складовою її інформаційної безпеки. оскільки ризики в он-лайн середовищі пов'язані не лише з виведенням із ладу окремих чи усіх компонентів системи, а скоріше із несанкціонованим доступом до персональних даних користувачів та комерційної інформації, наслідками якого можуть бути як матеріальні так і репутаційні втрати.

Проведені дослідження за темою кваліфікаційної роботи свідчать про те, що для здійснення атаки на веб-ресурси використовуються різноманітні набори інструментів та методи проникнення в системи безпеки у залежності від існуючих помилок та вразливостей у програмних рішеннях. Необхідно зазначити, що відсутність вразливостей є найкращим способом захисту веб-ресурсів. Тому ще на початковому етапі розробки політики безпеки веб-ресурсів установи доцільно опрацювати (було опрацьовано) існуючі правила та вимоги захисту веб-ресурсів, які дозволяють визначити мінімально необхідні рівні послуг безпеки для забезпечення захисту веб-ресурсів. Вказівки щодо управління ризиками інформаційної безпеки, критерії контролю кібербезпеки, практичні поради і рекомендації з комп'ютерної безпеки містяться у стандартах інформаційної безпеки (наприклад, ISO/IEC 27005:2018).

За результатами проведеного аналізу актуальних атак на веб-ресурси, за даними консорціуму OWASP, було визначено, що статистика кібератак кожного року змінюється, але найпопулярнішими є атаки «Cross-site scripting» (міжсайтовий скриптинг), «Information leakage» (витік інформації), «Content spoofing» (підмінна даних через заміну контенту сторінок), SQL-injection, Broken authentication.

Для кожного типу проаналізованих атак (SQL-ін'єкції, XSS-атаки, атаки CSRF, Broken authentication, атаки за разунок невірно налаштованої конфігурації, атаки щодо витоку критичних даних, атаки за рахунок відсутності контролю рівня доступу на функціональному рівні) запропоновано можливі способи захисту веб-ресурсів від загроз, що

зумовлені цими атаками, або способи їх уникнення.

Атаки на веб-ресурси, виконуються, як правило, одночасно з сеансами роботи легальних користувачів і, зазвичай, використовуються стандартні HTTP-порти, що унеможлиблює блокування трафіку на рівні цих портів, оскільки доступ до веб-ресурсів буде повністю заблокований ззовні або зсередини. Виходом з цієї ситуації може бути використання сучасних високопродуктивних міжмережних екранів веб-додатків Web Application Firewall, які розміщуються між зовнішніми користувачами і веб-додатками та аналізують весь HTTP / HTTPS-трафік, виявляючи та блокуючи шкідливі запити до того, як вони зможуть вплинути на користувачів або на веб-додаток.

Web Application Firewall дозволяє захищати клієнтів від SQL-injection, PHP-injection, Brute force, Cross-site scripting, а також від експлуатації вразливостей Zero-day у веб-аплікаціях. Але самотійно Web Application Firewall не захищає веб-ресурси від низькорівневих атак. Тому для підвищення ефективності захисту інтернет-ресурсів необхідно застосовувати Firewall разом з системою запобігання вторгнення IPS, яка розпізнає ознаки вторгнення, виявляє атаки та блокує їх. Тобто, лише комплексна система, в якій поєднані Firewall, IPS та WAF захищатиме веб-ресурси на всіх рівнях моделі OSI.

При виконанні кваліфікаційної роботи було також досліджено один із способів перевірки захищеності веб-ресурсів - пентест. Було детально проаналізовано методики тестування на проникнення, такі як є: OSSTMM, NIST SP800-115, OWASP, ISSAF, PTES. Саме у цих документах зібрані докладні рекомендації з організації та проведення тестування на проникнення. При виборі конкретної методики для проведення пентестів повинні враховуватися особливості діяльності конкретної установи, її бізнес-процеси та процеси інформаційної безпеки.

Пентестування, за допомогою імітації спрямованих кібератак, дозволяє оцінити захищеність веб-ресурсів від несанкціонованого впливу, використовуючи різні моделі вторгнень, знайти вразливості та помилки у веб-додатках. Результати пентестування можуть зумовити модернізацію системи захисту

веб-ресурсів, зокрема переналаштування параметрів політики безпеки, виведення програмного коду веб-додатків, з метою усунення небезпечних вразливостей, і тим самим підвищити рівень захисту веб-ресурсів від дій зовнішнього злоумисника.

Одним із можливих засобів захисту веб додатків, написаних на Python є застосування модуля NGIUX, що дозволяє виявити хакерську атаку на веб ресурс, та мінімізувати ризик витоку інформації.

Отже, при формуванні системи захисту веб-ресурсів установи необхідно:

- опрацювати правила та вимоги захисту веб-ресурсів;
- ознайомитися зі статистикою кібератак, визначити які саме атаки є найпопулярнішими і, у яких сферах діяльності, які вразливості програмних рішень вони використовують для проникнення в систему безпеки та виходячи з цього, визначати способи захисту веб-ресурсів від цих загроз;
- сформувати комплексну систему захисту веб-ресурсів на всіх рівнях моделі OSI, в якій будуть поєднані Firewall, IPS та WAF;
- здійснити за допомогою аудиту, одним із способів проведення якого є пентест, оцінку захищеності веб-ресурсів та визначити найбільш критичні ризики для їх безпеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу/НОРМАТИВНИЙ ДОКУМЕНТ СИСТЕМИ ТЕХНІЧНОГО ЗАХИСТУ ІНФОРМАЦІЇ. - 2003.
2. Замула О.А. Аналіз міжнародних стандартів в галузі оцінювання ризиків інформаційної безпеки /О.А. Замула, В.І. Черниш //Системи обробки інформації: зб. наук. пр. – Х.: ХУПС, 2011. – Вип. 2 (92). – С.53-56. 77.
3. Захист мережі: комплексний підхід – 2017. URL: <https://goo.su/16VM> (дата звернення 17.11.2022 р).
4. Комп'ютерні мережі. Типологія. URL:http://comp-net.at.ua/index/topologija_komp_39_juternikh_merezh/0-6. –Назва з титул. екрана.] (дата звернення 7.11.2022 р).
5. Комп'ютерні мережі. URL: <http://moodle.udc.ntu-kpi.kiev.ua/moodle/mod/resource/view.php?id=5873>. (дата звернення 7.11.2022).
6. Колченко Г. Ф. Щодо сертифікації систем управління сучасними телекомунікаційними мережами/ Г. Ф. Колченко // Матер.IV міжнар.наук.-техн. Симпозіуму «Нові технології в телекомунікаціях». К.:ДУІКТ, 2011р.
7. Критерії оцінки захищеності інформації в комп'ютерних системах від несанкціонованого доступу/ НД ТЗІ 2.5-004-99. URL: <https://tzi.ua/assets/files/%D0%9D%D0%94-%D0%A2%D0%97%D0%86-2.5-004-99.pdf> (дата звернення 11.12.2022 р).
8. Методи підвищення показників якості системи управління телекомунікаційними мережами/ [В.В. Хиленко, Л.Н. Беркман, Г.Ф. Колченко, О.Г. Варфоломеева]. –К.: Норіта-плюс, 2007.–236 с
9. Міжнародний стандарт інформаційної безпеки ISO/IEC 27005:2018 URL: https://www.iso.org/standard/75281.html?browse=tc_ (дата звернення 22.12.2022 р).

10. Петров К. Е., Кобзев І. В., Онищенко Ю. М. ПОЛІТИКА БЕЗПЕКИ WEB-ЗАСТОСУВАНЬ ТА СЕРВЕРІВ /Вісник ХНУВС. 2015. № 2 (69). - с. 256-263.
- 11.Пентест. Тестування на проникнення. URL: <https://kr-labs.com.ua/service/cybersecurity/testuvannya-na-pronyknennya-ta-vrazlyvosti-pentest> (дата звернення 22.12.2022 р).
- 12.Про мережеві технології URL :<http://lotocka.blogspot.com/> . – Назва з титул. екрана].(дата звернення 09.01.2023р).
- 13.Рекомендації інформаційної безпеки для малого та середнього бізнесу – 2018. URL: <https://habr.com/ru/post348892> (дата звернення 9.01.2023 р).
- 14.Тертичний В.О. Мережна безпека, система виявлення та протидії атакам, відмовостійкість мереж / В.О. Тертичний. //Харків, ХНУРЕ, Матеріали XXIV міжнародного молодіжного форуму «Радіoeлектроніка та молодь в XXI столітті». – 2020. С. 281-283.
- 15.Чому потрібен план реагування на кіберінциденти – 2018. URL:<https://nv.ua/ukr/biz/experts/chomu-potriben-plan-reaguvannya-na-kiberincidenti-2472078.html>. (дата звернення 09.01.2023 р).
- 16.Alisha Cecil, “A summary of Network Traffic Monitoring and Analysis Techniques Whitepaper”,2006, URL: http://www.cs.wustl.edu/~jain/cse567-06/ftp/net_monitoring/index.html; (дата звернення 10.01.2023 р).
- 17.Bhavani A.B. Cross-site Scripting Attacks on Android WebView / A.B. Bhavani // International Journal of Computer Science and Network. — 2013. — Vol. 2, Issue 2. — 5 і. URL: <http://ijcsn.org/IJCSN-2013/2-2/IJCSN-2013-2-2-03.pdf>. (дата звернення 12.01.2023 р).
- 18.Cuff P. Distributed channel synthesis / P. Cuff // IEEE. Trans. Inf. Theory. — 2013. — Vol. 59(11). — P. 7071 — 7096.
- 19.D'Hoinne J. Magic Quadrant for Web Application Firewalls. / Jeremy D'Hoinne, Adam Hils, Ayal Tirosh, Claudio Neiva. – 2018. URL: <https://gtmr.it/2IVLx72>. (дата звернення 14.01.2023 р).

20.Ed Wilson, “Network Monitoring and Analysis: A Protocol Approach to Troubleshooting”, Prentice Hall PTR, 2013.

21.Ganore P. What Is A Web Server And How Does It Function?
Pravin Ganore. – 2017. URL:<https://www.milesweb.com/blog/hosting/web-server-function/> (дата звернення 16.01.2023 р).

22.Handbook on Ontologies / eds. S. Staab and R. Studer. — International Handbooks on Information Systems. — Berlin: Springer, 2009. — 832 p. 29.

23.Jan Engelhardt and Nicolas Bouliane. Writing Netfilter modules. Revised, February 07, 2011.

24.Juniper Official Web-site, URL:
<http://www.juniper.net/techpubs/software/junos-es/junos-es93/junos-es-swconfig-security/understanding-icmp-flood-attacks.html>. (дата звернення 18.01.2023 р).

25.Laura Chappell. «Wireshark (R) 101: Essential Skills for Network Analysis».
Laura Chappell University, 2013.

26.Lysenko S. Botnet detection technique for corporate area network / Lysenko S., Savenko O., Kryshchuk A., Kljots Y. Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), Berlin, DE, IEEE. – 2013. – P. 315-320.

27.Open Source Intrusion Detection System using Snort: conf. paper from the 4th International Symposium on Sustainable Development. International Burch University, Faculty of Engineering and Information Technologies. Sarajevo: SOC-1441, 2014. 8 p.

28.Pomorova O. Multi-agent based approach for botnet detection in a corporate area network using fuzzy logic / Pomorova O., Savenko O., Lysenko S., Kryshchuk A. // Computer Networks 20th International Conference, CN 2013, Lwówek Śląski, Poland, June 17–21. – 2013. – P. 243–254.

29.Ricca F. / F. Ricca, P. Tonella. – 2001. – URL:
<https://dl.acm.org/citation.cfm?id=381476>. (дата звернення 11.12.2022 р).

- 30.Sahin C.S. General Framework for Evaluating Password Complexity and Strength / C.S. Sahin, R. Lychev, N. Wagner. — 11 і. К. ГусПей, М. Марпер. — 11 р. —URL: <http://arxiv.org/abs/1512.05814> (дата звернення 17.11.2022 р).
- 31.Sankar R. Burpsuite – A Beginner’s Guide For Web Application Security or Penetration Testing / Ravi Sankar. – 2018. URL: <https://kalilinuxtutorials.com/burpsuite/> (дата звернення 23.01.2023 р).
- 32.Schieler C. Rate-distortion theory for secrecy systems / C. Schieler, P. Cuff // IEEE Trans. on Inf. Theory. — 2014. — Vol. 66(12). — Q.7584–7605.
- 33.Shaker A. Gore S. Intrusion Detection System (IDS): Case Study. International Conference on Advanced Materials Engineering IPCSIT. 2011. Vol. 15. P. 6 – 9.
- 34.Shui Yu, “Distributed Denial of Service Attack and Defense” (Springer-Briefs in Computer Science), Springer–Verlag New York, 2014
- 35.Terri W. O. Firewalls. Практичне застосування міжмережевих екранів /William Olgtri Terri. – Київ: ДМК Пресс, 2001. – 400 с. – (Захист та адміністрування).
- 36.Tee Huu,“Evaluation of a Multi-Agent System for Simulation and Analysis of Distributed Denial-of-Service Attacks.New edition”, 2013, URL: <http://www.dtic.mil/dtic/tr/fulltext/u2/a420448.pdf> (дата звернення 26.12.2022 р).
- 37.Top Free Network-Based Intrusion Detection Systems (IDS) for the Enterprise. URL: <https://www.upguard.com/articles/top-free-network-based-intrusion-detection-systems-ids-for-the-enterprise> (дата звернення 23.11.2022 р).
- 38.Web Application Firewall
URL:https://www.owasp.org/index.php/Web_Application_Firewall (дата звернення 11.11.2022 р).
- 39.Website Security Statistics Report: 2018. — WhiteHat Security, 2018. — 30

p. URL: <https://info.whitehatsec.com/Website-Stats-Report-2018.html> (дата звернення 16.01.2023 p).

40.Website Security Statistics Report: 2015. — WhiteHat Security, 2015. — 30 p. URL: <https://info.whitehatsec.com/Website-Stats-Report-2015.html> (дата звернення 23.12.2022 p).

41.Yaworski P. Web Hacking 101. How to Make Money Hack- ing Ethically / Peter Yaworski // Lean Publishing. — 2017. URL: http://leanpub.com/web-hacking-101_ (дата звернення 11.11.2022 p).

42.Vijayarani S., Sylviaa M. Intrusion Detection System – A Study. International Journal of Security, Privacy and Trust Management. 2015. Vol. 4, N 1. P. 31 – 44.

43. Owasp URL: <https://owasp.org/www-project-top-ten/>(дата звернення 18.01.2022 p).

44. Content essential URL: <https://www.cyber.gov.au/acsc/view-all-content/essential-eight> (дата звернення 11.11.2022 p).

45.Controls v8 Gvuide URL: https://kr-labs.com.ua/books/CIS_Controls_v8_Guide.pdf (дата звернення 11.11.2022 p).

46.CIP Security and IEC 62443-4-2 URL:[https://kr-labs.com.ua/books/2020-ODVAConference_CIP_Security_and_IEC_62443_Visoky_Wiberg_Final.p](https://kr-labs.com.ua/books/2020-ODVAConference_CIP_Security_and_IEC_62443_Visoky_Wiberg_Final.pdf)df (дата звернення 17.11.2022 p).

47.CIS Controls Version 8 URL:https://krlabs.com.ua/books/CIS_Controls_v8_Guide.pdf (дата звернення 22.01.2023 p).

48.HACKMAGEDDON – статистика інформаційної безпеки URL: <http://www.hackmageddon.com> (дата звернення 17.11.2022 p).

49.Cybersecurity Framework. URL: <https://www.nist.gov/cyberframework>. (дата звернення 15.09.2022 p).