НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПОГОДЖЕНО

Декан факультету
Інформаційних технологій
_____/ Болбот І.М., д.т.н, проф. /
підпис ПІБ, вчене звання і ступінь

«___»_____2025 р.

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ

Завідувач кафедри
Комп'ютерних систем і мереж
_____/ Касаткін Д.Ю., к.п.н., доцент. /
підпис ПІБ, вчене звання і ступінь

«___»_____2025 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

На тему: «Розробка засобів забезпечення резильєнтності мережі на прикладному рівні»

Спеціальність: 123 «Комп’ютерна інженерія»

Освітня програма: Комп’ютерні системи та мережі

Керівник дипломного проекту:_____/ Коваленко О.Є. /
підпис ПІБ

Виконав:_____/ Лукашенко Д.Ю. /
підпис ПІБ

КИЇВ-2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ
І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

«ЗАТВЕРДЖУЮ»
завідувач кафедри
комп'ютерних систем, мереж та кібербезпеки
_____/ Касаткін Д.Ю., к.п.н., доцент. /
підпис ПІБ, вчене звання і ступінь
«__» _____ 2025 р.

ЗАВДАННЯ

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ РОБОТИ СТУДЕНТУ

Лукашенку Дмитру Юрійовичу
(прізвище, ім'я, по батькові)
Спеціальність (напрям підготовки): 123 «Комп'ютерна інженерія».
Освітня програма: Комп'ютерні системи та мережі

Тема магістерської роботи: «Розробка засобів забезпечення резильєнтності мережі на прикладному рівні»

затверджена наказом проректора з науково-педагогічної роботи та цифрової трансформації НУБІП України від "29 " жовтня 2024 р. № 1941 "С"
Термін подання завершеної роботи на кафедру 13 листопада 2025р
Вихідні дані до магістерської роботи: середовище віртуалізації VirtualBox, Міжмережевий екран веб додатків open-appsec, міжмережвевий екран веб додатків Modsecurity, спеціально вразливий веб додаток OWASP BWA
Перелік питань, що підлягають дослідженню:

1. Аналіз концепції резильєнтності та підходів до її забезпечення на прикладному рівні. Аналіз актуальних загрози прикладного рівня
2. Обгрунтування архітектурних та WAF рішень для забезпечення резильєнтності прикладного рівня
3. Проектування та моделювання Web Application Firewall. Тестування рішення, дослідження ефективності двох рішень: сигнатурного та ML

Дата видачі завдання " 29 " жовтня 2024 р.
Керівник магістерської роботи _____/ Коваленко О.Є. д.т.н., професор /
(підпис) (ПІБ, вчене звання і ступінь)
Завдання прийняв до виконання _____/ Лукашенко Д.Ю. /
(підпис) (ПІБ)

РЕФЕРАТ

Пояснювальна записка: 86с., 61 рис., 2 додатка, 22 використаних джерела.

РЕЗИЛЬЄНТНІСТЬ, БЕЗПЕКА, НАДІЙНІСТЬ, АДАПТИВНІСТЬ ВІДНОВЛЮВАЛЬНІСТЬ, OWASP, WAF, OPEN-APPSEC, MODECURITY

Мета роботи – дослідити підходи до забезпечення резильєнтності інформаційних систем на прикладному рівні моделі OSI, проаналізувати еволюцію актуальних загроз веб-додатків, дослідити ефективність двох різних WAF систем

Об’єкт – резильєнтність інформаційних систем і динаміка змін актуальності загроз веб-додатків у контексті еволюції OWASP Top 10.

Предмет – методи, технології та засоби забезпечення резильєнтності на прикладному рівні, зокрема застосування брандмауера веб-додатків (WAF) для виявлення та блокування вразливостей

Робота складається з трьох розділів.

У першому розділі проведений аналіз концепції резильєнтності та підходів її забезпечення на прикладному рівні. Аналіз актуальних загрох прикладного рівня

У другому розділі розглянуто сучасні архітектурні рішення інтегрування WAF у мережу, здійснено порівняльний аналіз open-source WAF рішень

У третьому розділі детально описаний процес розгортання WAF, проведено тестування та дослідження ефективності двох WAF, сингатурного та WAF на базі машинного навчання

В результаті виконання магістерської роботи проведено аналіз резильєнтності, сучасних веб загроз. На основі проведеного аналізу можна зробити висновок, що WAF є найдієвішим засобом забезпечення резильєнтності. роведено порівняльний аналіз двох ключових архітектур захисту веб-додатків: сигнатурного WAF (ModSecurity) та WAF на основі машинного навчання (open-appsec). На основі проведеного дослідження можна стверджувати, що обране ML-WAF рішення є ефективним компонентом.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IT – Information Technology;

WAF – Web Application Firewall;

HTTP – Hyper Text Transfer Protocol;

HTTPS – Hyper Text Transfer Protocol Secure;

NGFW – Next Generation Firewall;

CDN – Content Delivery Network;

IPS – Intrusion Prevention System;

HTML – Hypertext Markup Language;

CSS – Cascading Style Sheets;

SPA – Single Page Applications;

MPA – Multi Page Applications;

OWASP - Open Web Application Security Project;

SQL – Structured Query Language;

XML - Extensible Markup Language;

XSS – Cross Site Scripting;

URL - Uniform Resource Locator;

NAT - Network Address Translation;

VPN - Virtual Private Network;

OSI - Open Systems Interconnection;

DOS - Denial of Service;

DDOS - Distributed Denial-of-service attack;

SSH - Secure Shell;

CRS – Core Rule Set;

ZAP - Zed Attack Proxy.

DHCP – Dynamic Host configuration Protocol

ВСТУП

Складні та комплексні системи широко поширені в природі та суспільстві. Сучасний світ висуває високі вимоги до IT-інфраструктури та комп'ютерних мереж, обумовлюючи складність їхніх структур. Мережі передачі даних обслуговують усі види людської діяльності. Незалежно від того, чи використовується Інтернет для професійних цілей чи дозвілля, для критично важливих для безпеки програм чи електронної комерції, він став невід'ємною частиною нашого повсякденного життя, впливаючи на функціонування суспільств. Однак Інтернет не був призначений для виконання всіх цих функцій і, як такий, є вразливим до широкого кола проблем. Зловмисні атаки, збої програмного та апаратного забезпечення, людські помилки (наприклад, неправильні конфігурації програмного та апаратного забезпечення). Вартість збоїв у комунікаційних мережах є значною і лише зростатиме, оскільки їх охоплення поширюватиметься на функціонування нашого суспільства. З бурхливим розвитком Інтернету веб-додатки набули статусу критично важливих активів, забезпечуючи взаємодію з клієнтами та доступ до конфіденційної інформації. Відмова від веб-додатків практично неможлива, оскільки це призводить до втрати конкурентоспроможності та зупинки бізнес-процесів. На тлі зростання популярності веб-додатків, зростає і необхідність їхнього захисту, адже понад 75% хакерських атак спрямовані саме на вразливості прикладного рівня. Наслідки таких атак можуть бути руйнівними: від втрати особистих даних і платіжної інформації до несанкціонованого доступу до комерційної таємниці та використання веб-додатків як точки входу в корпоративну мережу. Традиційні методи мережевого захисту, як-от міжмережеві екрани, не можуть попередити атаки на веб-сервіси, оскільки вони орієнтовані на мережевий та транспортний рівні, тоді як веб-додатки працюють на прикладному рівні. Цей факт змушує нас перейти від простої безпеки до забезпечення резильєнтності

1. ДОСЛІДНИЦЬКА ЧАСТИНА

1.1 Огляд та аналіз предметної області. Основні поняття

З розвитком цифрових технологій комп'ютерні мережі стали критично важливою інфраструктурою. Від правильно спроектованої мережі залежить функціонування державних установ, бізнесу, фінансових систем, транспортної інфраструктури. Зростання кількості загроз, а також залежність користувачів і організацій від коректного функціонування мережі змушують переглянути підхід до проектування та експлуатації мереж. Мережі повинні проектуватися орінтуючися не лише на безпеку чи надійність, а на більш масштабну характеристику - резильєнтність.

Під резильєтністю у сфері інформаційних технологій розуміють здатність системи протистояти негативним впливам, швидко відновлювати працездатність після збоїв чи атак та зберігати безперервність надання критичних послуг. Резільєнтність - це комплексна характеристика, яка поєднує надійність, доступність, відмовостійкість, захищеність та здатність до відновлення. Якщо надійність зазвичай передбачає запобігання відмовам, то резильєнтність охоплює також адаптацію та самовідновлення системи. Це особливо важливо у сучасних мережах, де повністю уникнути інцидентів практично не можливо. Ключовим завданням стає мінімізація їх впливу та швидке відновлення працездатності

Резильєнтність, як явище описується низкою міжнародних стандартів. Серед основних - стандарти ISO/IEC, рекомендації ITU, документи NIST, а також практичні настанови ENISA та NCSC. Порівнявши ці стандарти, можна зрозуміти, які підходи пропонуються для забезпечення резильєнтності

ISO/IEC 27001 є базовим міжнародним стандартом у сфері управління інформаційною безпекою. Він визначає вимоги до створення, впровадження, підтримки та постійного вдосконалення системи управління безпекою. У підході до резильєнтності він не описує детальних технічних інструкцій, і більше сфокусований на процесному підході. В ньому детально описуються визначення

ризиків для інформаційних активів, розробка політик і процедур реагування, інтеграція безпеки з бізнес-процесами. Стандарт ISO/IEC 27031 доповнює 27001 і фокусується саме на готовності до забезпечення безперервності бізнесу. Він пропонує структуру для побудови Disaster Recovery Plan та Business Continuity Plan (BCP), а також акцентує увагу на тестуванні та навчанні персоналу. Стандарти ISO підходять до резильєнтності з боку менеджменту та безперервності бізнесу, що робить його універсальним, але менш технічним.

Національний інститут стандартів і технологій США (NIST) розробив документ SP-800-160 Vol. 2 Developing Cyber Resilient Systems, який зараз є одним з найвпливовіших у сфері кіберрезильєнтності. NIST розробив модель життєвого циклу резильєнтності, яка базується на чотирьох принципах:

- anticipate – передбачати можливі інциденти та загрози;
- withstand – витримувати вплив атак або збоїв мінімізуючи наслідки атак;
- recover – відновлювати функціональність після інциденту;
- adapt – адаптувати систему для підвищення стійкості в майбутньому.

NIST також акцентує увагу на таких ключових технічних рішеннях як багаторівневий захист, відмовостійкість, ізоляції критичних компонентів, застосуванні спеціальних хибних (deception) технік.

Багаторівневий захист (defence-in-depth) передбачає інтеграцію людських, технологічних та операційних ресурсів для створення кількарівневого захисту. Ця стратегія безпеки добре зарекомендувала себе. Як приклад, можна привести використання міжмережевих екранів для розділення зон, виявлення підозрілої поведінки на рівні застосунку.

Відмовостійкість (redundancy) - дублювання критичних компонентів, що дозволяє підтримувати ключову функціональність навіть у разі виходу з ладу окремих елементів. Наприклад, створення резервних копій, включаючи конфігурації та віртуалізовані ресурси, забезпечення додаткових можливостей зберігання і обробки, дублювання обладнання.

Ізоляція критичних компонентів (segmentation/isolation) – визначення сегментів або інших наборів ресурсів на основі критичності та надійності для подальшого окремого їх захисту. Це дозволяє обмежити можливе поширення атак чи помилок. Як приклад ізоляції критичних компонентів можна привести створення віртуальної пісочниці або детонаційної камери для ненадійних URL, ізоляція функцій безпеки від функцій не пов'язаних з безпекою, ізоляція компонентів на основі бізнес-функцій. Сегментація на рівні підмережі і прив'язка їх до різних доменів безпеки також дозволяє підвищити безпеку надійність і відмовостійкість

Застосування deception-технік (cyber deception) – навмисне введення в оману потенційних зловмисників, приховування критично важливих активів. Наприклад, створення віртуальних чи фізичних систем, які імітують справжні сервери або робочі станції (honeypots). Існує розшириення концепції honeypot - створення віртуальної мережі, яка імітує інфраструктуру організації - honeynets. Іншими прикладами використання deception-технік є розміщення в системі неправильніх логінів, паролів та ключів доступу, що має назву decoy credentials. Розгортання удаваних веб-сервісів, БД або API, які імітують реальні бізнес додатки, але не містять цінної інформації, така техніка називається decoy services

ITU-T (Міжнародний союз електрозв'язку) - міжнародна організація що визначає стандарти в сфері телекомунікація та мережевої інфраструктури. Для впродядкування практик кіберзахисту ITU-T розробив низку документів, які окреслюють підходи до управління безпекою, захисту мереж і протидії кіберзагрозам. Ключовим документом, що визначає підходи до керування інформаційною безпекою, є ITU-T рекомендація X.1051. Цей документ надає детальні настанови для телекомунікаційних операторів, адаптуючи принципи міжнародних стандартів ISO/IEC 27000 до специфіки галузі. Стандарт спрямований на забезпечення конфіденційності, цілісності та доступності даних і послуг, що є основою для побудови резильєнтних систем. Додатково, X.1051 охоплює такі аспекти, як розробка політик безпеки, управління персоналом,

фізичний захист інфраструктури, управління інцидентами та забезпечення безперервності бізнесу. Ці заходи дозволяють мережі ефективно протистояти загрозам і відновлюватися після збоїв, що є суттю резильєнтності. Подальшим розвитком цих ідей є серія документів, присвячених резильєнтності мереж, серед яких варто виділити ITU-T Рекомендацію X.1060. Цей стандарт пропонує фреймворк для створення та функціонування Центрів кіберзахисту (Cyber Defence Centre, CDC). CDC розглядається як інтегрований підрозділ, що надає послуги для управління кіберризиками, охоплюючи не тільки реагування на інциденти, а й проактивну оборону. X.1060 описує процеси створення, управління та оцінки ефективності таких центрів, а також деталізує портфоліо послуг, від стратегічного планування до аналізу вразливостей. Документ переводить концепцію резильєнтності з теоретичного рівня на практичний, надаючи конкретні інструменти для побудови стійкого захисту

Агенство Європейського Союзу з кібербезпеки (ENISA) відіграє ключову роль у забезпеченні резильєнтності на рівні всього Європейського Союзу. ENISA розробляє та публікує численні звіти й рекомендації, що містять конкретні показники для оцінки та підвищенні рівня стійкості. У документі «Cybersecurity and Resilience of Communication Networks», агенство пропонує методології оцінки ризиків, спеціально адаптовані для критичної інфраструктури. ENISA займається розробкою моделей побудови стійких телекомунікаційних мереж. Ці моделі побудовані на архітектурних принципах, що включають в собі ідеї відмовостійкості та розподіленості ресурсів, з метою мінімізації наслідків можливих збоїв чи кібератак. ENISA фокусується на конкретних секторах економіки (енергетика, фінанси, транспорт, телекомунікації) та надає деталізовані інструкції, що враховують їхню специфіку. Це дозволяє організаціям не просто формально дотримуватися визначених стандартом вимог, а й реально підвищувати свою здатність протистояти загрозам. Це робить рекомендації ENISA дієвим інструментом для забезпечення резильєнтності в Європейському просторі.

1.2 Обґрунтування вибору фокусу на прикладному рівні

Враховуючи актуальність резильєнтності, дана робота зосереджується на її забезпеченні на прикладному рівні L7 мережевої моделі OSI. Нижчі рівні з L1 по L4, мають добре вивчені вектори атак, тоді як, загрози прикладного рівня, які постійно розвиваються, роблять прикладний рівень більш вразливим.

На фізичному та канальному рівнях L1-L2 резильєнтність досягається шляхом дублювання обладнання, резервування каналів зв'язку та протоколів, що забезпечують відмовостійкість (наприклад, STP і його можливості додаткового тюнінгу, PortChannel) та захист мережі (port-security, dynamic-arp inspection, DHCP snooping)

На мережевому та транспортному рівнях резильєнтність забезпечується за допомогою динамічних протоколів маршрутизації, фільтарції маршрутів, протоколам групи FHRP, міжмережевих екранів нового покоління (NGFW) та систем запобігання вторгненням (IPS/IDS), які фільтрують трафік на основі IP-адрес і портів.

Особлива вразливість та фокус на прикладному рівні пояснюється також постійною еволюцією веб-додатків, використанням нових технологій таких як, мікросервісна архітектура та інтерфейси прикладного програмування. Сучасні кібератаки все частіше спрямовані саме на цей рівень, оскільки зловмисники прагнуть обійти класичні мережеві засоби захисту. Атаки на прикладному рівні використовують логічні недоліки в програмному забезпеченні, а не технічні вразливості протоколів. Тому вони можуть успішно пройти обладнання, яке не має змоги аналізувати вміст HTTP/HTTPS-трафіку. Саме тому резильєнтність на прикладному рівні вимагає іншого, більш спеціалізованого підходу, орієнтованого на поведінку та логіку веб-додатків. Резильєнтність на цьому рівні є критично важливою, оскільки саме тут відбувається безпосередня взаємодія з кінцевим користувачем і створюється комерційна цінність послуги. Навіть якщо вся нижча мережева інфраструктура (від L1 до L4) є ідеально захищеною і стійкою,

вразливість на рівні додатка може призвести до повного колапсу бізнес-процесів, фінансових збитків, крадіжки даних або втрати довіри клієнтів. Резильєнтність на прикладному рівні гарантує, що додаток може не лише витримувати атаки, а й підтримувати свою функціональність в умовах надмірного навантаження, збоїв або непередбачуваних подій.

1.3 Комплексний аналіз архітектурних та програмних рішень для забезпечення резильєнтності прикладного рівня

Засоби забезпечення безпеки і надійності, як одних із основних складових забезпечення резильєнтності прикладного рівння часто функціонально перетинаються. Серед таких засобів варто розглянути: балансувальники навантаження, CDN (Content Delivery Network), брандмауери веб-додатків (web application firewalls)

Балансувальники навантаження - це мережевий пристрій або сервіс, який розподіляє мережевий трафік між кількома серверами або ресурсами для забезпечення високої доступності, продуктивності та відмовостійкості. Він діє як "диспетчер трафіку", спрямовуючи запити користувачів на різні сервери, щоб уникнути перевантаження одного вузла та забезпечити безперебійну роботу додатків. Основне завдання мережного балансувальника полягає в тому, щоб збільшити доступність та швидкість роботи веб-сервісів та застосунків для кінцевих користувачів. [6]

Існує кілька методів реалізації балансувальників навантаження:

- round robin - трафік буде скерований на перший доступний сервер (вибирається випадковим чином), після чого ця машина переміщається в кінець черги — наступного разу мережний балансувальник звернеться до неї тільки після перебору інших серверів. Цей метод підходить для пулу серверів зі схожими/однаковими конфігураціями та невеликою кількістю активних підключень;
- найменша кількість підключень (least connection) - трафік буде скерований на сервер із найменшою кількістю активних підключень;
- метод найменшого часу відповіді (least response time method) - nрафік буде скерований на сервер з найменшою кількістю активних підключень та найменшим середнім часом відповіді (враховуються обидва параметри);
- метод хешу - цей метод використовує хеш-функцію для зіставлення IP-адреси клієнта з певним сервером. Запити від одного й того самого джерела спрямовуються на один і той самий сервер. Це корисно, якщо потрібно зберегти сесію для клієнта на одному сервері або забезпечити постійне з'єднання між ними. [6]

Балансувальники навантаження дозволяють забезпечити такі складові резильєнтності як доступність і відмовостійкість, але не забезпечують захист від веб-загроз

CDN (Content Delivery Network) - це мережа з серверів, які розташовані в різних точках світу. Вона потрібна для прискореного доставлення або розподілу статичного контенту вашого сайту, такого як: зображень, аудіозаписів, відео та інших файлів. CDN можуть значно покращити швидкість завантаження веб сайтів, зберігаючи копії контенту на серверах, розташованих у різних точках світу. Це означає, що користувачі, які знаходяться далеко від сервера, на якому зберігається веб сайт, все одно можуть отримувати контент швидко. Оскільки, виконується розподілення навантаження на декілька серверів це покращує безпеку і ускладнює доступ для зловмисників. CND дозволяють підвищити стійкість [7]

Міжмережеві екрани нового покоління (Next Generation Firewalls) - е інноваційне рішення для забезпечення мережевої безпеки, яке значно перевершує можливості традиційних фаєрволів. У той час як класичні міжмережеві екрани обмежуються фільтрацією трафіку на основі IP-адрес і портів, NGFW додає до цього глибокий аналіз даних, контроль додатків, ідентифікацію користувачів і функції запобігання загрозам. NGFW включає інтегровані системи виявлення та запобігання вторгнень, що дозволяє вчасно виявляти та нейтралізувати складні атаки, такі як шкідливі програми, атаки нульового дня, та експлойти. NGFW може розпізнавати окремих користувачів, а не лише пристрої, що дозволяє встановлювати політики безпеки на рівні індивідуальних акаунтів. Це особливо важливо для контролю доступу до критично важливих ресурсів. NGFW дозволяє ідентифікувати і контролювати окремі додатки, що працюють у вашій мережі, незалежно від портів або протоколів, які вони використовують. Це дозволяє адміністраторам мереж встановлювати політики безпеки на рівні конкретних додатків, забезпечуючи таким чином більш точний контрольх. [8]

Брандмауер веб-додатків (Web Application Firewall, WAF) – це пристрій, який захищає веб-додатки від більшості існуючих на сьогоднійшній день атак в тому числі, від OWASP Top Ten. 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. В результаті WAF захищають критично важливі для бізнесу веб-додатки і веб-сервери від атак.

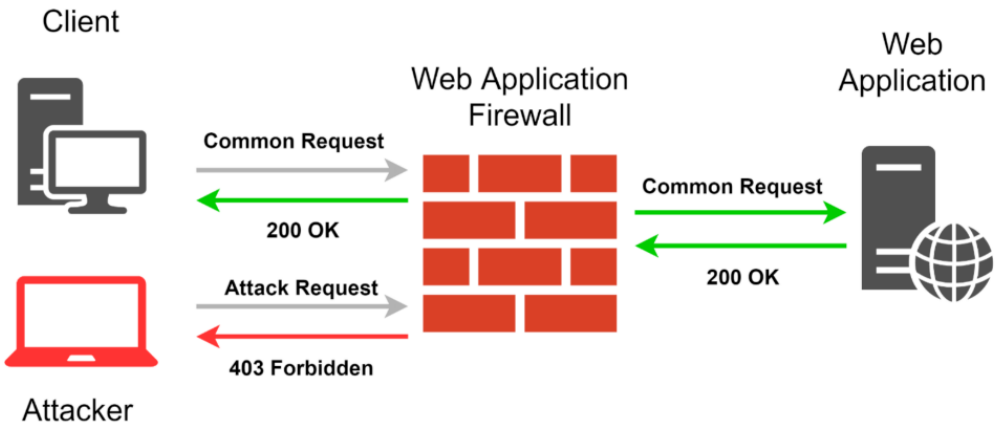


Рисунок 1.1 – Схема роботи WAF

Є помітний збіг між міжмережевими екранами наступного покоління (NGFW) та міжмережевими екранами веб-додатків (WAF). Сучасні мережеві брандмауери наступного покоління (NGFW) мають можливості контролю та аналізу трафіку на прикладному рівні. Це дозволяє їм ідентифікувати та керувати програмами, фільтрувати вміст і виявляти загальні мережеві загрози Web-Application Firewall (WAF) відрізняється від них більш вузькою спеціалізацією. WAF - це єдиний інструмент, розроблений виключно для захисту веб-додатків. Він дозволяє глибше аналізувати HTTP/HTTPS трафік, структуру веб-запитів і типові атаки. WAF ефективно протидіє таким загрозам, як SQL-ін'єкції, міжсайтовий скриптинг (XSS), маніпуляції з файлами cookie та інші вектори. Навіть найсучасніший NGFW, не може забезпечити такий рівень детального аналізу і захисту від специфічних веб-атак. WAF є спеціалізованим інструментом, що доповнює загальний мережевий захист, забезпечуючи сфокусовану і глибоку оборону.[**Error! Reference source not found.**]

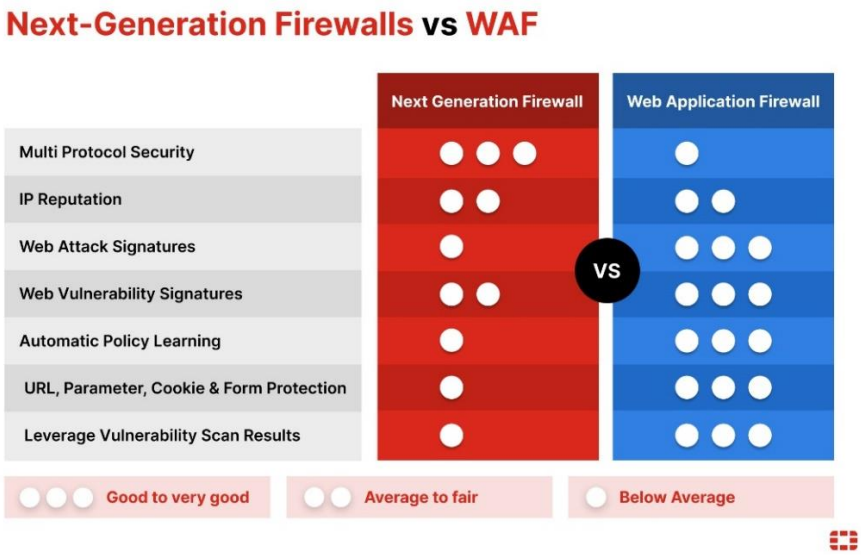


Рисунок 1.2 – Відмінності між NGFW та WAF

Жоден окремий засіб не забезпечує повноцінної резильєнтності, поєднання описаних засобів дозволяє створити багаторівневий захист, що значно підвищує здатність веб-сервісу протистояти збоям та атакам на різних рівнях.

Для забезпечення іншої важливої характеристики резильєнтності - здатності до відновлення після інцидентів, збоїв або цілеспрямованих атак використовуються як архітектурні, так і управлінські та технічні рішення. Регулярне створення резервних копій баз даних, конфігурацій та критичних компонентів дозволяє швидко відновити стабільний стан роботи. Використання кластерних рішень, георграфічно розподілених серверів, реплікованих баз даних надає можливість уникнути єдиної точки відмови. У випадку збою одного вузла робота автоматично переключається на інший. Постійний моніторинг подій безпеки, логування та чіткі сценарії реагування зменшують час виявлення проблем (Mean Time to Detect, MTTD) і час на відновлення (Mean Time to Recover, MTTR). Відновлення як характеристика резильєнтності досягається не окремим засобом, а комплексом описаних механізмів, які спрямовані на швидке повернення системи в робочий стан із мінімальними втратами.

1.4 Аналіз загроз прикладного рівня

Для проведення аналізу актуальних загроз веб-додатків, було використано методологію OWASP Top 10. OWASP - це міжнародна некомерційна організація, яка займається класифікацією векторів атак та вразливостей, зосереджується на аналізі та поліпшенні безпеки програмного забезпечення. Розгляд списку 2021 року у поєднанні з аналізом попередніх версій та поточних тенденцій дозволяє не лише зрозуміти поточний стан, але й сформулювати обґрунтовані прогнози щодо майбутньої еволюції загроз. Чергове оновлення відбулося у 2021 році, і в порівнянні з попередньою версією за 2017 рік, воно не просто змінило порядок категорій, а й продемонструвало зміщення парадигми в оцінці ризиків. Найбільш помітною зміною стало переміщення категорії "Broken Access Control" на перше місце у списку 2021 року. Це відображає необхідність у правильному налаштуванні доступу до ресурсів, який часто недооцінюється в реальних середовищах. Категорія "Broken Authentication", яка раніше займала провідні позиції, втратила свій пріоритет, що частково пояснюється розвитком типових засобів аутентифікації, та фреймворків таких як OAuth 2.0 або OpenID Connect.

Новою зміною стала також поява категорії "Cryptographic Failures", яка замінила попередній пункт "Sensitive Data Exposure". Ця зміна наголошує не просто на необхідності захисту даних, а на використанні правильних криптографічних практик: наявності сучасних шифрувальних алгоритмів, захисту TLS-з'єднань, належного управління ключами тощо. Було впроваджено нову категорію - Insecure Design, яка звертає увагу на важливості вбудовування безпеки на рівні архітектури додатку. Це демонструє поступове переорієнтування підходів до захисту: від реактивного до проактивного, де загрози попереджаються ще на етапі проєктування, а не лише виявляються в кінцевому коді.

Деякі категорії загроз, які були присутні раніше, були вилучені. Наприклад, такі загрози як "XML External Entities (XXE)" та "Insecure Deserialization" втратили статус окремих пунктів, що може свідчити про зменшення їх поширеності або включення в ширші узагальнені категорії. Крім того, оновлений список 2021 року приділяє більшу увагу питанням моніторингу інцидентів безпеки. Поява категорії "Security Logging and Monitoring Failures"

свідчить про усвідомлення критичної ролі логування подій для вчасного виявлення атак і реагування на інциденти. [10]

OWASP Top 10–2021 не лише оновлює перелік технічних загроз. Список стає відображенням ширших змін у сфері інформаційної безпеки: від локального виправлення помилок - до системного мислення, архітектурної зрілості та ефективного реагування на кіберзагрози. Це оновлення є важливим індикатором для всіх учасників життєвого циклу програмного забезпечення, і має бути враховане в практиці розробки та оцінки ризиків.

Оновлення OWASP Top 10 було опубліковане у 2021 році, але воно залишається актуальним і у 2025-му. Поясненням цьому є те, що більшість загроз, описаних у ньому, є базовими та досі активно експлуатуються у веб-додатках. OWASP спирається на великі масиви емпіричних даних, зібраних із реальних проєктів, а також на експертні оцінки.. До появи наступного офіційного оновлення цей перелік продовжує бути базовим орієнтиром для побудови безпечних веб-систем. Робота над наступною версією списку OWASP Top 10 триває і збір даних ще не завершено, але вже зараз можна виділити певні тенденції, що вказують на зміну напрямку загроз. Аналіз поточних атак та поширення нових технологій дає змогу передбачити можливі пункти наступного списку загроз. Ймовірно, в наступних версіях OWASP Top 10 зростатиме актуальність загроз, пов'язаних зі штучним інтелектом та машинним навчанням. Вже зараз ці технології все більше інтегруються у суспільство, включаючи і веб-додатки. Велике поширення та популярність хмарних та мікросервісних архітектур, вже сьогодні дають можливість передбачити, що такі загрози як Server-Side Request Forgery займатимуть вищі позиції у майбутньому списку. [10]

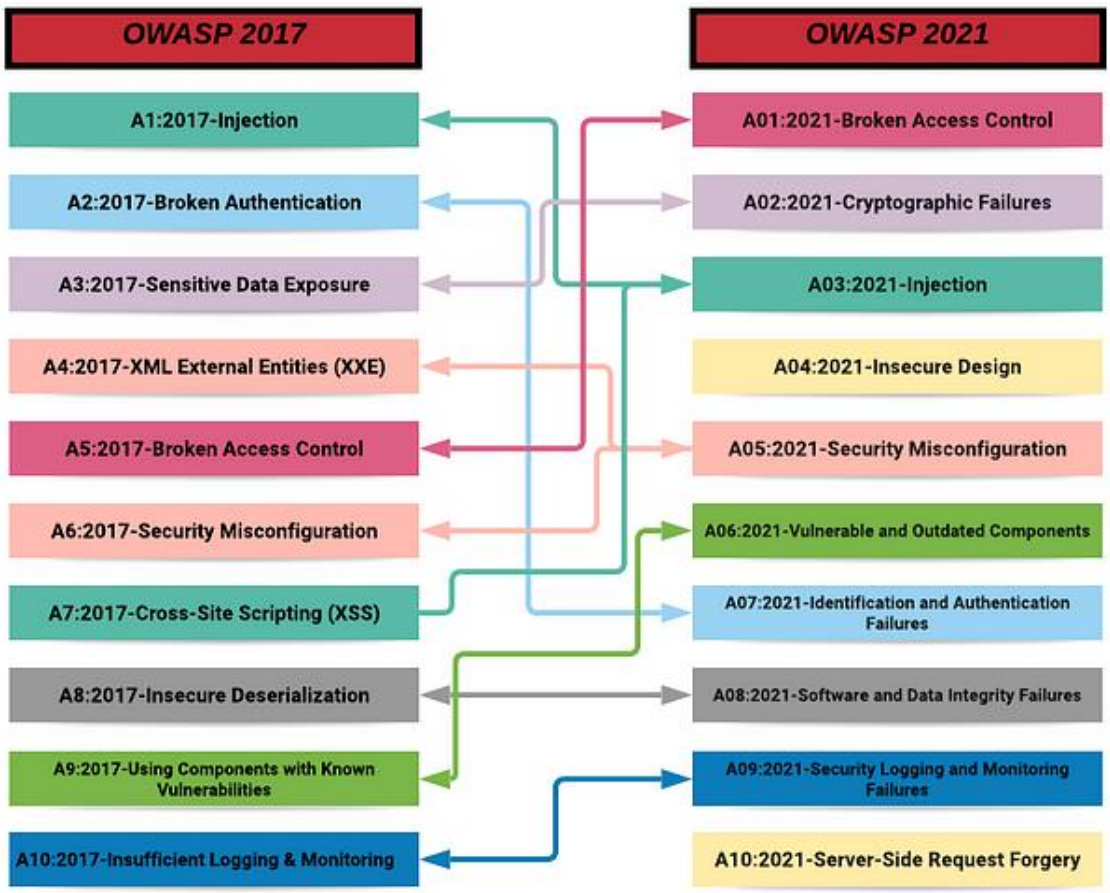


Рисунок 1.3 – OWASP Top Ten 2017 та 2021

A1: Недоліки контролю доступу (Broken Access Control) Ця категорія піднялася з п’ятого місця (A05:2017) на перше, що демонструє значне зростання її актуальності. Вразливості, пов’язані з некоректним контролем доступу, залишаються поширеними та небезпечними. Суть цієї вразливості полягає у відсутності перевірки доступу до об’єкта (адмін сторінка, файл серверу і т.д). Наприклад, у нас, як у звичайного користувача, є сторінка на якій ми можемо бачити доступну для нас інформацію. Маніпулюючи URL та замість значення «documents» підставивши «../../», ми можемо змінювати директорії та переглядати файли. OWASP зазначає, що понад 94% протестованих застосунків мали хоча б один випадок порушення контролю доступу.[10]

Рисунок 1.4 – Недолік контролю доступу додатку bWAPP

A2: Недоліки криптографії (Cryptographic Failures) У 2017 році ця категорія мала назву "Sensitive Data Exposure", однак у новій версії наголос зроблено не на наслідках (втраті чи витоку даних), а на причинах - саме криптографічних помилках. Це може бути використання слабких або застарілих алгоритмів (наприклад, MD5, SHA-1), відсутність шифрування в передаванні або зберіганні даних, неправильне управління ключами, використання застарілих TLS-протоколів. Розширення категорії відображає зростання вимог до захисту персональних, медичних або фінансових даних, а також актуальність відповідності до стандартів, таких як GDPR. Найпростіший приклад – передача даних по протоколу HTTP. Дані, що передаються по протоколу HTTP не шифруються, а при проходженні даних від комп’ютера користувача до Web-серверу, вони проходять досить велику кількість різних вузлів. Через відсутність шифрування дані особливо вразливі до перехоплення [10]

A3:Ін’єкції. Категорія "Ін’єкції" залишилася серед лідерів, хоча опустилася з першого на третє місце Зловмисник використовує ін’єкційні методи, такі як SQL, Shell, та LDAP-ін’єкції, які дозволяють зловмиснику вводити в додаток ненадійні вхідні дані, які потім оброблюються інтерпретатором, як частина команди або запиту. В 2021 році, XSS-атаки, які раніше виділялися в окрему категорію (A07:2017), але в 2021 році XSS стали частиною загального класу ін’єкцій, оскільки за своєю суттю це теж введення зовнішнього кодую. Всі дані, як правило, зберігаються в реляційних базах даних, звернення до яких відбуваються у вигляді запитів, які написані на спеціальній мові запитів. В більшості базах даних використовують мову структурованих запитів SQL, через це SQL-Injection є однією з найбільш розповсюджених атак типу ін’єкція. SQL – Injection – це вразливість, яка дозволяє зловмиснику втручатися в запити, які додаток робить до своєї бази даних. Зазвичай, це відкриває доступ до даних, які не можливо переглянути звичайним способом, або може призвести до компроментації облікового запису. Розглянемо просту SQL-ін’єкцію на прикладі додатку, що дозволяє здійснювати корисувачам вхід до системи. При здійсненні входу користувач вводить свої облікові дані (наприклад, логін – Diplomat та пароль – Qwerty1234), після чого генерується SQL-запит: SELECT * FROM users WHERE username = 'Diplomat' AND password = 'Qwerty1234'. Виконується пошук в базі даних, буде знайдено користувача та всю інформацію про нього, відповідно ми отримуємо успішний вхід в систему. При цьому, якщо додаток вразвилий до SQL-ін’єкції зловмисник може виконати вхід в систему під обліковим записом користувача та будь-якого паролю. Для цього достатньо в полі логіну ввести значення administrator'--. [11]

Таке заповнення форми призводить до відправки SQL-запиту SELECT * FROM users WHERE username = 'administrator'--' AND password = ' ', в якому частина запиту паролю ігнорується та інтерпретується як коментар, оскільки послідовність '-- в мові SQL вважається коментарем. В результаті ми потрапряємо

до облікового запису адміністратора за допомогою будь-якого випадкового значення паролю. [11]

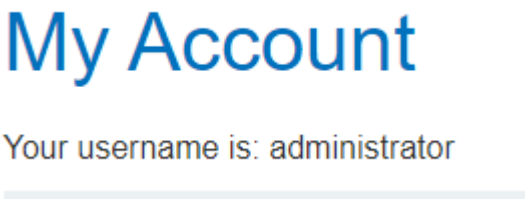


Рисунок 1.5 – Вдала експлуатація SQL-ін’єкції

Міжсайтовий скриптинг (XSS). XSS (Cross Site Scripting) – одна з атак на веб-системи, яка дозволяє впроваджувати на севрер шкідливий код, який потім виконується на стороні клієнта (в браузері клієнта). В результаті успішного проведення атаки, хакер може отримати авторизаційні дані користувача (cookie) або перенаправити жертву на іншу сторінку-клон. Користувач може не помітити підміни сторінки і ввести особисту інформацію, яка потрапить до рук хакеру. Зазвичай, особливо вразливими до XSS атак є сайти на яких є стандарнті форми (форми пошуку, коментарів, зворотнього зв’язку). На сьогоднішній день чіткої класифікації атак міжсайтового скриптинга не існує, але експерти по всьому світу виділяють три основні типи XSS атак (Reflected XSS, Stored XSS, DOM XSS).[12]

Reflected XSS (відбитий, або непостійний XSS). Найбільш розповсюджений тип XSS. Атаку такого типу XSS можна провести просто підставивши в кінець URL скрипт, який буде автоматично виконаний в браузері жертви. Працює цей принцип наступним чином:

- хакер заздалегідь створює URL-адресу, яка містить шкідливий код і відправляє її своїй жертві;

- жертва переходить за цією адресою;

- в браузері жертви виконується шкідливий код і зловмисник отримує результат виконання атаки (cookie жертви).

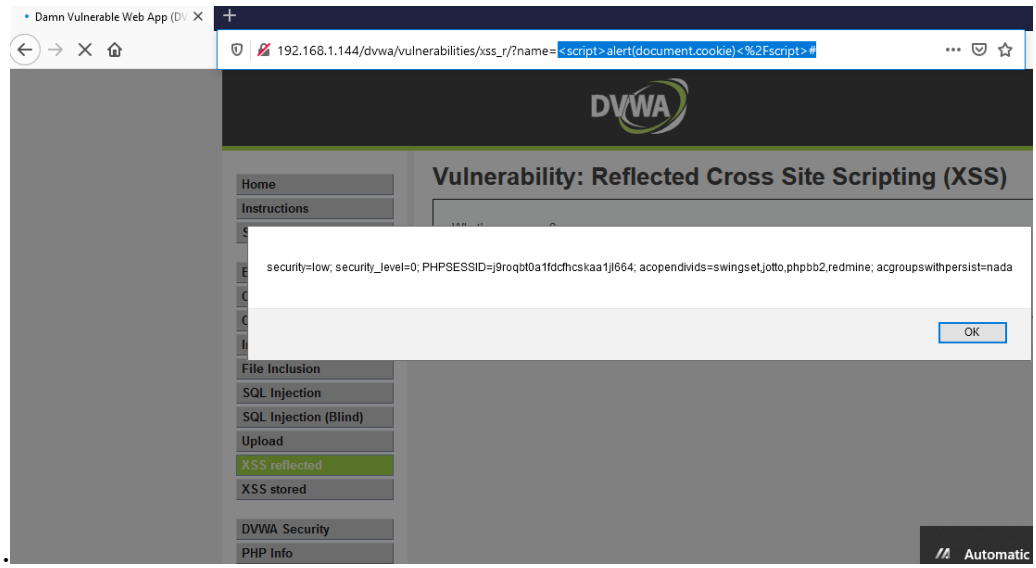


Рисунок 1.6 – Приклад простої Reflected XSS атаки

Хакер, використовуючи методи соціальної інженерії може змусити перейти жертву за URL-адресою та викрасти його особисті дані.

Stored XSS (постійний XSS). Є більш складною та небезпечною атакою, яка дозволяє зловмиснику розмістити шкідливий код на сервері, який буде виконуватися кожного разу при відвідуванні сайту (або конкретної сторінки сайту). На відміну від непостійної XSS атаки, постійна XSS атака не потребує переходу по спеціально сформованому URL. [12]

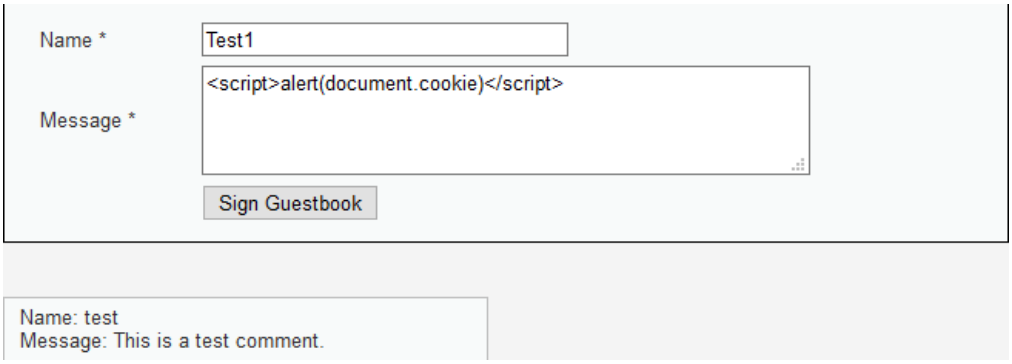


Рисунок 1.7 – Приклад простої Stored XSS

A4: Небезпечне проектування (Insecure Design). Категорія Insecure Design з’явилася вперше у списку OWASP Top 10 саме у 2021 році. Це є наслідком того що деякі вразливості не є просто результатом помилок у коді, а виникають значно раніше - на етапі архітектурного або концептуального проектування. Категорія охоплює помилки у проєктуванні контролів безпеки, відсутність або недієвість захисних механізмів, брак багаторівневої моделі захисту (defense in depth), а також загальну відсутність стратегічного мислення у контексті безпеки. На відміну від реалізаційних помилок, які можна відносно швидко виправити, архітектурні недоліки потребують суттєвих змін у логіці системи або взагалі повного рефакторингу, що може бути складним або навіть неможливим. Один із найвідоміших прикладів небезпечного проектування - механізм відновлення пароля на основі запитань і відповідей, наприклад: «Яке ім’я вашого улюбленого домашнього улюбленця?» або «Яка у вас була перша школа?». У сучасних умовах такі питання є ненадійними: багато хто зможе знайти ці відповіді у відкритих джерелах, особливо в соцмережах. Спроба реалізувати "легку" аутентифікацію обертається серйозною вразливістю, що випливає не з коду, а з хибної логіки захисту. Категорія Insecure Design не описує конкретні недоліки, а скоріше підкреслює потребу у плануванні захисту на етапі розробки архітектури та у впровадженні практик threat modeling.

A5: Недоліки параметрів безпеки (Security Misconfiguration) В цій вразливості людський фактор відіграє найбільшу роль. Вектор атаки зловмисника націлений на отримання доступу до облікових записів за замовчуванням, сторінок, що не використовуються, незахищених файлів і каталогів. Ці недоліки часто дають зловмисникам можливість отримати доступ до деяких системних даних або функцій. Іноді такі недоліки можуть призвести до повного краху системи. Першим прикладом можуть бути налаштування облікового запису адміністратора за замовчуванням (наприклад, логін – admin, пароль – admin). Часто такий обліковий запис використовується для налаштування проекту. Після проведення всіх необхідних налаштувань важливо не забути змінити

налаштування облікового запису, щоб запобігти можливості отримання доступу до облікового запису адміністратора через застосування стандартних налаштувань. Другим прикладом може бути незакритий доступ до каталогів серверу. Ці налаштування проводяться в файлі «Robots.txt», в якому забороняється (або дозволяється) індексація сторінок сайту пошуковими системами. Приклад вмісту файлу Robots.txt. Через неправльно заповлений файл Robots.txt зловмисник може отримати доступ до панелі адміністатора, директорії з паролями навіть не виконуючи логін на сайт.

Ін’єкція зовнішніх об’єктів XML (XXE), яка в 2017 році була окремим пунктом, стала частиною security misconfigurations. XML (Extensible Markup Language) – мова розмітки, схожа на HTML, але на відміну від HTML, XML призначений для більш зручного зберігання та передачі даних, в тому числі через Інтернет. XML має функцію під назвою Document Type Definition (DTP), яка містить інформацію про структуру XML-документу. Ін’єкція зовнішнього об’єкту XML – це вразливість веб безпеки, яка дозволяє зловмиснику втручатися в обробку даних XML додатком. Це може дозволити перегдялати файли у файловій системі серверу і взаємодіяти з будь-якими іншими внутрішніми або зовнішніми системами, до яких веб-додаток може отримати доступ. Розглянемо приклад ін’єкції XML, що дозволяє переглянути файл /etc/passwd. Додаток не забезпечує захисту від атак XXE, тому ми експлуатуємо вразливість XXE відправляючи на сервер корисне навантаження.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [ <!ENTITY xxe SYSTEM
"file:///etc/passwd"> ]>
<stockCheck><productId>&xxe;</productId>
</stockCheck>
```

Рисунок 1.8 – Експлуатація XXE

Дана структура XXE визначає зовнішню сутність &xxe, значення якої є вмістом файлу /etc/passwd. Це призводить до того, що відповідь додатку буде включати в себе вміст файлу. [**Error! Reference source not found.**]

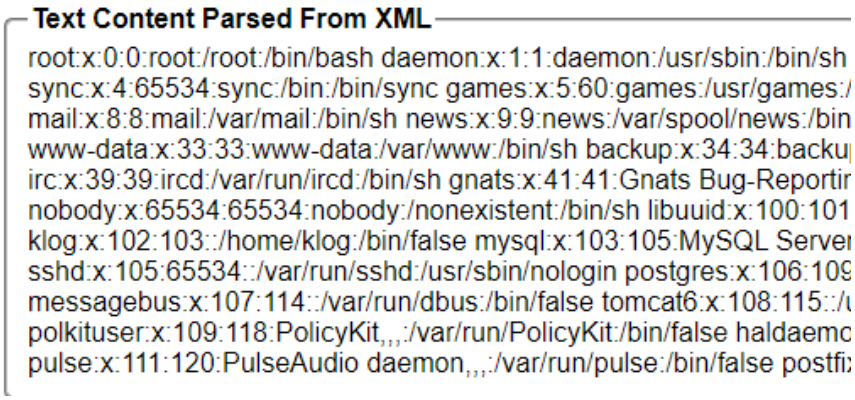


Рисунок 1.9 – Результат експлуатації XXE

A6:Використання компонентів з відомими вразливостями. Ця категорія що раніше відповідала A09:2017 стосується використання сторонніх бібліотек, модулів, фреймворків та інфраструктурних компонентів, які мають відомі вразливості або більше не підтримуються. У 2021 році вона стала ще більш критичною у зв’язку з масовим поширенням open-source залежностей. На даний момент більшість веб-додатків мають складну структуру, вони використовують фреймворки та бібліотеки (комерційні або бескоштовні). Одна з проблем – підримувати оновлення таких компонентів (ця проблема поширюється ще більше, коли різні бібліотеки та фреймворки використовують ішні підкомпоненти або бібліотеки). Зловмисник використовуючи інструменти автоматичного сканування або за допомогою ручного аналізу додатку знаходять компоненти, інформація про вразливість котрих вже відома широкій публіці. Тому на сьогоднійший день важливо використовувати компоненти лише з офіційних джерел, та слідкувати за оновленнями компонентів. [16]

A7:Недоліки системи аутентифікації та зберігання сесій. Дана вразливість ділиться на 2 вектори атаки: brute-force і session hijacking. Брутфорсом називається метод взлому облікових записів шляхом підбору потрібного ключа для входу (комбінації значень логіну та паролю). Суть методу полягає в послідовному автоматизованому перебору всіх можливих комбінацій символів з метою рано чи піздно знайти правильну. Програмне забезпечення для брутфорсу геренує варіанти комбінацій і перевіряє кожну з них. З точки зору математики вирішити задачу подібним способом можна завжди, але затрати часу на пошук не у всіх випадках може виправдати мету, так як поле пошуку, при дотриманні парольних політик може бути величезним[13].

Важливою засобом протидії brude force атакам є дотримання парольної політики. При створенні паролю важливо дотримуватися наступинх правил: парольна комбінація повинна містити букви верхнього та нижнього регістрів, спеціальні символи, повинна складатися з восьми або більше символів, парольна фраза не повинна бути словом, персональною інформацією або номером телефону.

Ще один вектор атаки, пов’язаний з недоліками аутентифікації - session hijacking (захоплення сеансу). Це атака, при якій сеанс корисувача може бути захоплений зловмисником. У більшості випадках, при здійсненні логіну у веб-додаток, сервер створює унікальну сесію для поточного корисувача (cookie). Значення сесії зберігається в браузері користувача і при виконанні наступного запиту до серверу, користувач разом з запитом GET, передає значення сесії, для своєї ідентифікації, як користувача сайту.

Зловмисник, використовуючи різні методи взлому сеансу (XSS-атака, за допомогою шкідвилого ПЗ, встановленого на комп’ютер користувача, методом грубої сили, аналіз трафіку) може викрасти cookie користувача та виконати вхід в його обліковий запис. [14]

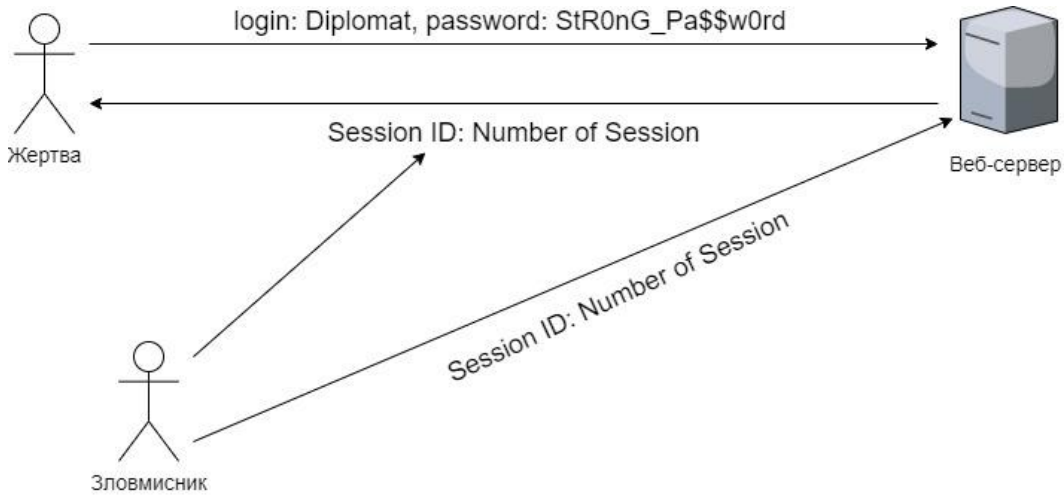


Рисунок 1.10 – Схема атаки захоплення сеансу

Для запобіганням атакам session hijacking ID-сеансу повинні захищатися прапорами Secure та HTTPonly. З використанням цих атрибутів, при спробі зловмисника перехопити запит, вся інформація про сеанс буде прихована.

A8: Порушення цілісності програмного забезпечення та даних (Software and Data Integrity Failures). Ця категорія є новою у списку OWASP Top 10 та описує ризики, пов’язані з порушенням цілісності програмного забезпечення та даних. На противагу від класичних вразливостей у коді, ця категорія фокусується на загрозах, що виникають через довіру до зовнішніх компонентів та інструментів, без належної перевірки їх автентичності або цілісності. Типовою ситуацією є використання сторонніх плагінів, бібліотек або модулів із неперевірених джерел (наприклад, публічні репозиторії, CDN-сервіси). Якщо в процесі CI/CD (Continuous Integration / Continuous Deployment) не впроваджено механізми перевірки надійності таких компонентів, це може призвести до компрометації системи, виконання шкідливого коду або отримання несанкціонованого доступу до ресурсів.

До цієї категорії також віднесено колишню вразливість A08:2017 - Insecure Deserialization. Сучасні додатки активно користуються програмними об’єктами, які в різні моменти часу потрібно передавати або просто зберігати. Серіалізація – це представлення специфічних для мови програмування структур і об’єктів в

єдиному форматі, як правило у вигляді рядка певного формату або послідовності байтів. Десеріалізація - це зворотній процес: відновлення структур та об’єктів із серіалізованого рядка або послідовності байт. Небезпечна десеріалізація – це десеріалізація користувацьких даних на веб-сайті. Вдалі атаки небезпечної десеріаліації можуть дозволити зловниснику виконати обхід аутентифікації, підвищення рівня привілегій на сторінці та віддаленого виконання коду. [15]

A9: Проблеми моніторингу та ведення журналів (Security Logging and Monitoring Failures) Аналог A10:2017 (Insufficient Logging and Monitoring), але з фокусом на оперативність реагування на інциденти. Уразливості цієї категорії включають відсутність належного журналювання подій безпеки, неефективний або повністю відсутній моніторинг, погано налагоджену інтеграцію з SIEM, відсутність тривог. Як результат - затримка у виявленні інцидентів, втрата доказової бази, неможливість аналізу подій після атаки.

A10: Server-Side Request Forgery. Server-Side Request Forgery (SSRF) – нова категорія вразливостей у OWASP Top 10 2021, яка увійшла до списку вперше, одразу посівши десяту позицію. Вона належить до числа тих вразливостей, які довгий час залишалися поза належною увагою, але з поширенням мікросервісних архітектур, контейнеризації, хмарних сервісів та внутрішніх API стала серйозною загрозою. [17]

SSRF виникає тоді, коли зловмисник змушує серверну частину вебзастосунку виконувати HTTP-запити до довільних ресурсів. Особливу небезпеку ця вразливість становить тоді, коли сервер має доступ до внутрішньої інфраструктури, ізольованої від зовнішнього середовища - таких як внутрішні API, метадані хмарних сервісів (наприклад, AWS), конфігураційні панелі або сховища з чутливою інформацією. Як приклад, можна уявити вебсервіс, який дозволяє користувачу завантажити зображення за URL-адресою. Якщо введення не перевіряється належним чином, атакуючий може ввести http://localhost:8080/internal-status і отримати інформацію з внутрішнього адміністративного інтерфейсу, до якого ззовні немає доступу. . [17]

Висновки до розділу

У цьому розділі було проведено огляд та аналіз теоретичних засад, необхідних для розуміння концепції резильєнтності На основі міжнародних стандартів, таких як NIST, ISO/IEC, ITU-T можна зробити висновок, що сучасна концепція резильєнтності виходить за рамки традиційної безпеки. Резильєнтність – ширка характеристка, що охоплює здатність системи не лише протистояти загрозам, але й швидко відновлюватися та адаптуватися до них. Цей багаторівневий підхід є критично важливим, оскільки повністю уникнути інцидентів у сучасних мережах практично неможливо.

Розглянули необхідність фокусу саме на прикладному рівні моделі OSI. На відміну від нижчих рівнів, які захищаються традиційними брандмауерами, загрози прикладного рівня постійно розвиваються і еволюціонують. Проведено комплексний аналіз архітектурних та програмних рішень забезпечення основних характеристик резильєнтності – захисту від загроз та здатності до відновлення. Визначено, що WAF та NGFW надають найбільший ефективний захист. WAF, як вузькоспеціалізований інструмент, відрізняється від NGFW здатністю глибоко аналізувати веб-трафік, що робить його незамінним для протидії цільовим атакам, таким як SQL-ін'єкції та міжсайтовий скриптинг. Проте саме поєднання засобів забезпечення резильєнтності створює багаторівневий захист, що значно підвищує здатність веб-сервісу протистояти збоям та атакам на різних рівнях.

Аналіз списку OWASP Top 10-2021 підтвердив актуальність загроз на прикладному рівні та їхню еволюцію, пов'язану з архітектурними недоліками та новими технологіями. Це дозволило мені зробити висновок, що WAF є не просто одним із засобів захисту, а фундаментальним елементом у багаторівневій архітектурі кіберрезильєнтності.

2. ОБГРУНТУВАННЯ АРХІТЕКТУРНИХ ТА WAF РІШЕНЬ ДЛЯ ЗАБЕЗПЕЧЕННЯ РЕЗИЛЬЄНТСНОТІ ПРИКЛАДНОГО РІВНЯ

2.1 Брандмауери веб-додактів в дизайні мережі

WAF працює на основі набору правил, що називаються політиками, які використовуються для фільтрації більшості відомих на сьогоднішній день атак. Багато служб WAF надають набір правил за замовчуванням, який періодично оновлюється. Це називається сигнатурним підходом.

WAF можуть працювати за моделлю негативної безпеки (чорний список), позитивної безпеки (білий список) або за гібридною моделлю. У моделі чорного списку використовуються попередньо встановлені сигнатури для блокування явно шкідливого веб-трафіку, а також сигнатури, призначенні для запобігання атакам, що використовують певні вразливості веб-сайтів і додатків. Наприклад, якщо кілька IP-адрес відправляють набагато більше пакетів, ніж це типово, брандмауер, налаштований за чорним списком може запобігти DDOS-атаці.

Модель білого списку дозволяє веб-трафік, що відповідає спеціально налаштованим критеріям. Наприклад, брандмауер можна налаштувати так, щоб дозволяти запити HTTP GET тільки з певних IP-адрес. Брандмауери з моделлю білого списку найкраще підійдуть для веб-додатків у внутрішній мережі, які призначені для використання тільки обмеженою групою людей, наприклад співробітниками компанії. [18]

WAF може інтегруватися в мережу наступними способами: режим мережевого моніторингу через SPAN порт, bridge mode, reverse proxy).

В режимі моніторингу пакети не проходять через брандмауер веб-додатків. Функція аналізатору комутованих портів (Switched Port Analyzer, SPAN) виконує пересилку копії трафіку, що надходить на якийсь порт, через інший порт цього ж комутатора. В такому режимі брандмауер виконує аналіз копії відстежуваного трафіку, а не пакетів, що пересилаються в дійсності. Перевага роботи в такому режимі в тому, що WAF не впливає на трафік, що

дозволяє уникнути проблем з продуктивністю та затримками. Недоліком роботи в такому режимі є те, що WAF працює з копією трафіка і не може запобігати атакам на веб-додатки.

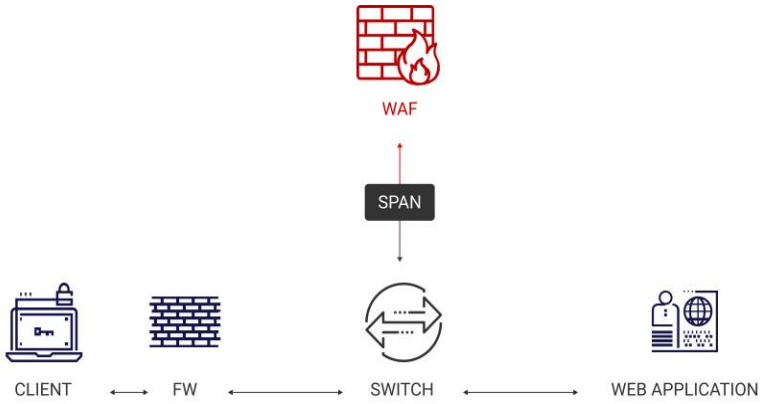


Рисунок 2.1 – Приклад реалізації WAF в режимі моніторингу

В режимі моста (bridge mode) WAF знаходиться на одній лінії між мережевим екраном і веб-серверами і діє як міст 2 рівня.

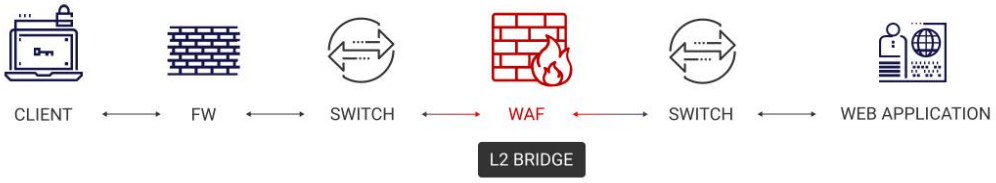


Рисунок 2.2 – Приклад реалізації WAF в режимі моста

Зворотній проксі-сервер. Як правило, проксі-сервери виступають у ролі посередників при здійсненні онлайн-з’єднань. Проксі можуть ділитися на типи за різними критеріями. Тип проксі залежить від виду пристою, який діє в якості проксі-серверу, рівня анонімності клієнта при використанні проксі, способу управління даними. Відповідно до ще одного критерію – розташування в струкруті мережі – проксі-сервері діляться на зворотні і прямі.

Прямий проксі сервер – при використанні терміну «проксі», частіше за все мають на увазі саме прямий проксі сервер. Прямі проксі – це типи проксі , які використовуються клієнтами для приховання своїх IP-адрес і збереження анонімності при роботі в Інтернеті. При використанні прямого проксі-серверу пристій віправляє звичайний запит, ніби проксі-серверу і не існує, але всі запити до цільової системи будуть проходити через проксі-сервер. Проксі приймає запити і перенаправляє їх через свою IP-адресу, приховуючи справжню IP-адресу користувача. Найчастіше прямі проксі-сервери використовуються звичайними користувачами для обходу блокованих сервісів.

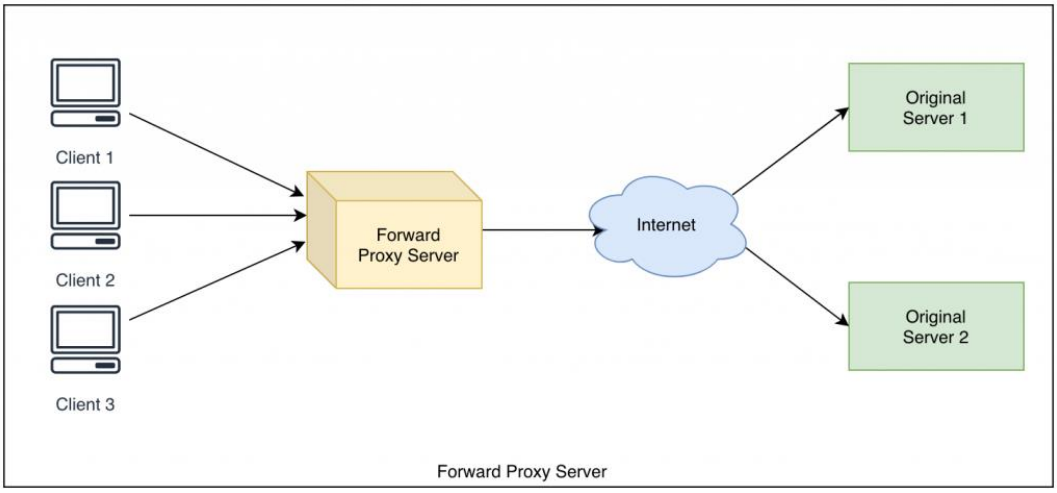


Рисунок 2.3 – Приклад роботи прямого проксі-серверу

Зворотній проксі-сервер – це проксі-сервер, який приймає запити від імені веб-серверів. Зворотній проксі працює не для клієнтів, а для веб-серверів. Якщо прямий проксі-сервер призначений для забезпечення анонімності клієнтів, то зворотній проксі-сервер призначений для забезпечення анонімності веб-серверів. Вони приховують від клієнтів справжнє місцеположення серверів.

Зворотній проксі приймає запити з Інтернету і визначає, чи потрібно переадресовувати запит на реальний сервер. [**Error! Reference source not found.**]

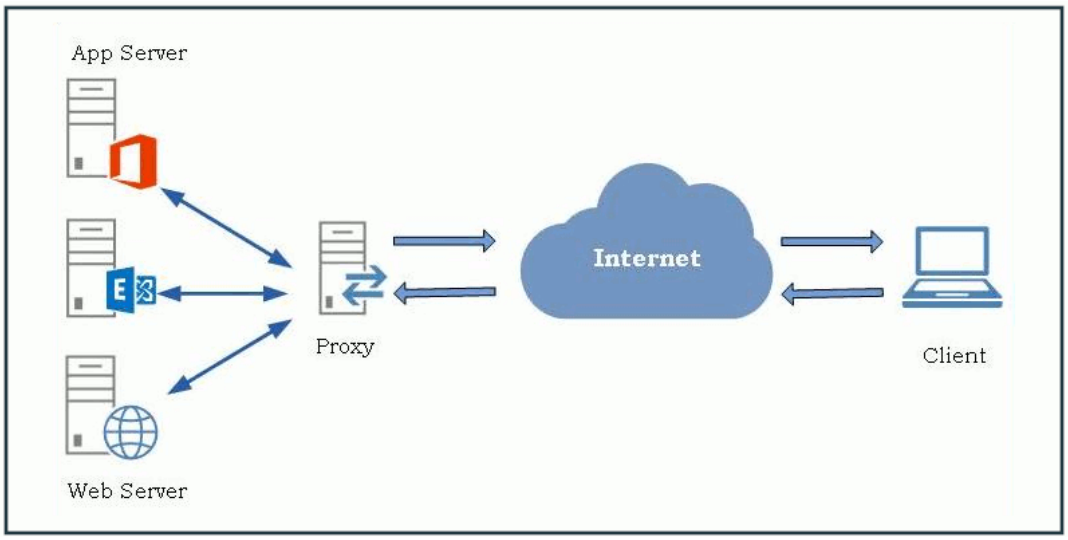


Рисунок 2.4 – Приклад роботи зворотнього проксі-серверу

Зворотні проксі-сервери можуть використовуватися для:

- балансування навантаження. Зазвичай веб-сайти з великою кількістю щоденних користувачів не можуть обробляти весь трафік за допомогою одного вихідного серверу. Отже, зворотній проксі-сервер може рівномірно розподіляти навантаження між внутрішніми серверами;

- додаткова безпека внутрішніх серверів. Якщо веб-сервер використовує зворотній проксі його адреса приховується, а користувачі можуть отримати доступ тільки до IP-адреси зворотнього проксі. Це вводить додатковий елемент захисту. Наприклад, набагато складніше провести атаку типу «відмова в обслуговуванні»;

- кешування. Це процес збереження копія файлів в кеші для більш швидкого повторного доступу. Кешування дозволяє веб-сайтам ефективно повторно використовувати раніше отримані дані. Це дозволяє веб-додаткам працювати більш ефективно;

- SSL-шифрування. Шифрування і дешифрування з’єднань для кожного користувача може виявитися неефективним для вихідного серверу. Зворотній проксі сервер може виконувати цю роботу, шифруючи і дешифруючі всі запити.[19]

Реалізація міжмережевого екрану в якості зворотнього проксі-северу є найбільш популярною та найбільш вживаною.

Рисунок 2.5 – Реалізація WAF в якості зворотнього проксі-серверу

WAF може бути реалізоваинй одним з наступних спопосбів, кожен з яких має свої переваги та недоліки:

- Мережевий WAF, як правило апаратний. Встановлючись локально мінімізують затримку, але являються більш дорогим варіантом.

- Хост-орієнтований WAF. Це рішення дешевше ніж мережевий WAF і пропонує більше можливостей для налаштування. Недоліком хост-орієнтованого WAF є споживання ресурсів локального серверу, складність реалізації та витрати на обслуговування.

- Хмарний WAF. Забезпечує найпростішу реалізацію, мають мінімальну початкову вартість, пропонують рішення, яке постійно оновлюєтсья для захисту від нових загроз без додаткові роботи або витрат з боку користувача. Недолік хмарного WAF полягає у віданні відповідальності третій стороні. Одним з найпопулярніших хмарних брандмауерів веб-додатків є Cloudflare WAF.

WAF-рішення поділяються на комерційні (Enterprise-level) та open-source. Серед комерційних рішень для захисту веб-додатків найпоширенішими є рішення від таких великих вендорів, як Imperva, Akamai, F5 та Fortinet. Вони пропонують повністю керовані рішення, які зазвичай включають у себе не тільки захист від типових веб-атак (SQL injection, XSS, CSRF), але й більш складні функції, серед яких: автоматичне оновлення правил та моніторинг безпеки в реальному часі,

аналіз трафіку за допомогою штучного інтелекту для виявлення нових вларзливостей, ітнеграція з іншими компонентами безпеки, такими, як системи запобігання вторгнень (IPS). Для прикладу, Imperva пропонує рішення, яке включає в себе як технології WAF, так і розширене захист від DDoS-атак, а також надає можливість використання кастомізованих правил для більш специфічних потреб. У свою чергу, Akamai забезпечує місткісне масштабування та оптимізацію продуктивності, а також можливість використання різних рівнів захисту від атак - від базового фільтрування до глибокого аналізу за допомогою штучного інтелекту . Недоліком таких рішень є висока вартість і необхідність в підготовці до роботи з інтерфейсами складних корпоративних платформ. Водночас вони є чудовим вибором для великих компаній з високими вимогами до безпеки та стабільності роботи.

У той час як комерційні рішення забезпечують високу якість і підтримку, вони можуть бути дорогими для малих та середніх підприємств. Відкриті рішення, навпаки, є безкоштовними, а деякі з них забезпечують високий рівень кастомізації та масштабованості. Найбільш популярними open-source WAF на ринку є ModSecurity, NAXSI, open-appsec та Coraza [20]

Таблиця 2.1 – Порівняльна характеристика open-source WAF рішень:

Характеристика	ModSecurity	open-appsec	NAXSI	Coraza
Метод виявлення загроз	Сигнатурний (OWASP CRS)	Поведінковий аналіз (ШІ)	Whitlist + евристика	Сигнатурний (OWASP CRS)
Покриття загроз OWASP Top 10	Повне	Повне	Часткове	Повне
Ймовірність хибних спрацювань	Висока	Низька	Середня	Середня
Захист від атак нульового дня	Відсутній	Наявний	Відсутній	Відсутній
Можливість створення власних правил	Повна	Обмежена	Базова	Повна
Системи логування та моніторингу	Підтримується	Вбудоване логування	Базове логування	JSON-логування
Інтеграція з NGINX	Так	Так	Так	Так
Інтеграція з Apache/IIS	Так	Ні	Ні	Ні
Вплив на продуктивність	Середній/високий	Низький	Низький	Низький

Загалом, вибір WAF повинен ґрунтуватися не лише на технічних характеристиках, але й на контексті використання, розмірі компанії, технічному рівні персоналу та наявності ресурсів. WAF слід розглядати як частину багаторівневої стратегії безпеки, що доповнює NGFW, IPS та інші засоби захисту, забезпечуючи стійкість мережі включно з прикладним рівнем

2.2 Обгрунтування вибору open-appsec WAF

На підставі проведеного аналізу архітектурних рішень та моделей безпеки WAF, для практичної реалізації було обрано рішення open-appsec. Цей вибір обґрунтований ключовими перевагами, які відрізняють його від традиційних аналогів і відповідають сучасним вимогам до резильєнтності.

open-appsec - це відкритий (open-source) брандмауер веб-додатків, розроблений компанією Check Point. Його ключовою особливістю є використання передових технологій машинного навчання для захисту веб-додатків та API від сучасних кіберзагроз. На відміну від традиційних WAF, які базуються на фіксованих сигнатурах і правилах, open-appsec здатний навчатися та адаптуватися до нових, невідомих атак.

Однією з головних переваг open-appsec є його гнучкість та широка сумісність з різними середовищами розгортання. Він розроблений для безшовної інтеграції в сучасні архітектури. Open-appsec може бути встановлений як модуль для популярних веб-серверів, таких як NGINX та Apache, що дозволяє інтегрувати його безпосередньо в існуючу інфраструктуру. Додатково, він повністю сумісний з платформами контейнеризації, такими як Docker та Kubernetes, що робить його ідеальним для захисту додатків, побудованих на мікросервісній архітектурі. Open-appsec також легко розгортається в провідних хмарних платформах: Amazon Web Services (AWS), Microsoft Azure та Google Cloud Platform (GCP). Така універсальність дозволяє використовувати open-appsec для захисту додатків будь-якого масштабу та складності, що є критично важливим для забезпечення резильєнтності в динамічних IT-середовищах. Якщо сервер обслуговує додатки локально і не працює як проксі між зовнішнім та внутрішнім доменами, open-appsec все одно може перевіряти трафік. Для цього вам необхідно змінити порт для локальних додатків на вищий (наприклад, з 80 на 8080) і додати правило проксі-сервера, що перенаправляє трафік із відкритого домену та порту на той самий локальний сервер, але вже на новий, вищий порт.

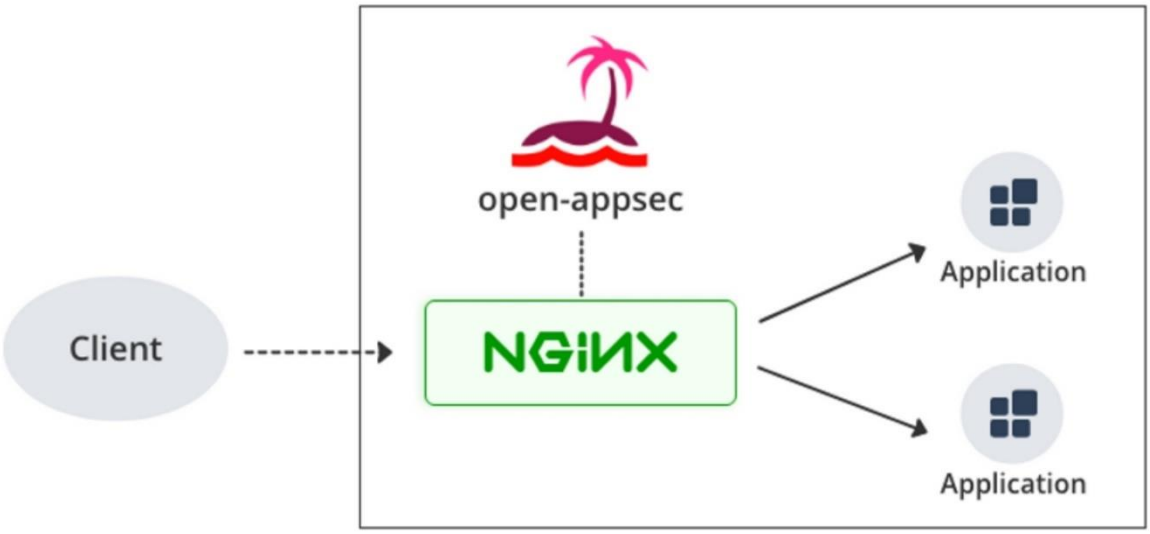


Рисунок 2.6 – Схема розгортання open appsec як надбудови для NGINX

open-appsec використовує контекстний двигун машинного навчання, який застосовує триетапний підхід для виявлення та запобігання атакам на веб-додатки та API.

Фаза 1 Розбір та декодування даних: На цьому етапі двигун аналізує всі поля HTTP-запиту, виконує декодування (наприклад, base64) та нормалізацію даних. Цей процес дозволяє представити запит у форматі, зручному для подальшого аналізу.

Фаза 2 - Пошук індикаторів атаки: Система шукає у проаналізованих даних патерни, що вказують на спроби використання відомих вразливостей (наприклад, SQL-ін'єкцій, XSS). Ці індикатори допомагають класифікувати потенційну загрозу.

Фаза 3 - Винесення остаточного рішення: На цьому етапі відбувається контекстна оцінка. Двигун враховує безліч факторів (середовище, користувача, URL, специфічні поля) і на основі зваженої функції присвоює запиту оцінку впевненості (confidence score). Це дозволяє open-appsec ухвалювати точні та обґрунтовані рішення щодо блокування трафіку. [21]

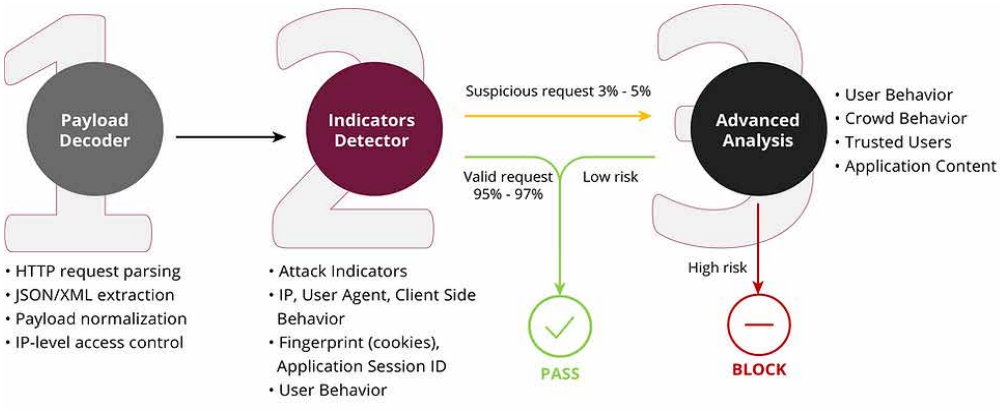


Рисунок 2.7 – Принцип роботи двигуна машинного навчання WAF open appsec

Машинне навчання часто сприймається як "чорний ящик", процеси якого важко зрозуміти та контролювати. Для вирішення цієї проблеми open-appsec використовує елементи гри для візуалізації та демонстрації прогресу навчання. Цей підхід дозволяє адміністраторам постійно відстежувати рівень навчання системи та отримувати чіткі вказівки щодо подальших дій. Це забезпечує прозорість роботи WAF, робить його налаштування більш інтуїтивним і дозволяє користувачам бути впевненими в ефективності захисту, який надається системою. [21]

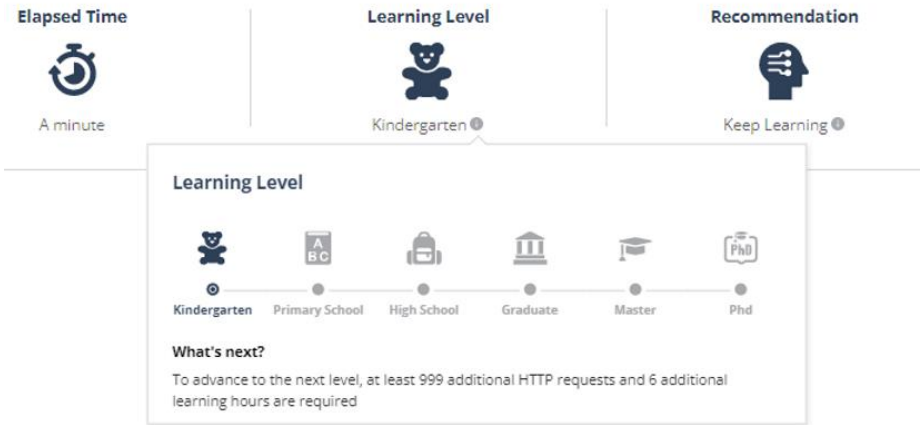


Рисунок 2.8 – Можливість відстеження процесу навчання WAF

Open-appsec пропонує два основні підходи до керування, що відповідають потребам різних команд і середовищ. Перший метод - декларативна конфігурація, яка реалізуєя, коли уся конфігурація здійснюється через локальні файли, Kubernetes YAML, CRD (Custom Resource Definitions) та анотації, що дозволяє зберігати конфігурацію як код. Це спрощує її версіонування та автоматизацію. Логування подій здійснюється у стандартний вивід або через Syslog, а для моніторингу може бути використана інтеграція з Prometheus та Grafana. Другий метод - конфігурація через веб-інтерфейс, що надається як послуга (SaaS). Цей підхід є більш інтуїтивним і орієнтований на простоту використання, оскільки керування здійснюється через зручний веб-інтерфейс. Така модель спрощує розгортання завдяки покроковим майстрам налаштування, забезпечує централізоване зберігання логів у хмарі, надає можливості для комплексного аналізу подій та моніторингу стану всіх агентів. Для автоматизації тут також доступні Terraform Provider та GraphQL API. [21]

2.3 Порівляльний аналіз open-appsec з популярними WAF-рішеннями

У більшості WAF-рішень, що представлені на ринку використовується оболонка ModSecurity та набір сигнатур і політик від OWASP CRS (Core Rule set). ModSecurity досяг кінця свого життєвого циклу (End of Life, EOL), в 2024 році, а OWASP CRS випустив велике оновлення з версії 3.x.x до 4.x.x, що стало першим значним оновленням за останні 8 років. Ці зміни повинні були призвести до суттєвих покращень у всіх WAF, які залежать від OWASP CRS, проте відповідно до дослідження «Best WAF Solution in 2024-2025 Real-World Comparison» хоча деякі WAF демонструють суттєві поліпшення, інші показали гірші результати ніж раніше і навіть не оновили свій CRS до нової версії. При виборі WAF існують дві найважливіші метрики: security quality (true positive rate) та detection quality (false positive rate).[22]

Security Quality (True Positive Rate) - здатність WAF правильно виявляти та блокувати шкідливі запити має вирішальне значення в роботі WAF. Він повинен ефективно блокувати навіть атаки нульового дня, а також активно протидіяти відомим методам атак, що використовуються хакерами. Цей показник оцінює здатність WAF коректно ідентифікувати та запобігати атакам. Розрахунок TPR здійснюється за формулою:

$$TPR = \frac{TP}{TP + FN} \quad (2.1)$$

де TP (True Positives) – кількість правильно заблокованих шкідливих запитів;

FN – кількість шкідливих запитів, яку WAF помилково пропустив.[22]

Detection Quality (False Positive Rates) – здатність WAF правильно обробляти легітимні запити, також є важливою, оскільки, будь яке втручання в ці запити може призвести до значних порушень у роботі та збільшенні навантаження на адміністраторів, оскільки буде вимагати додаткових налаштувань. Чим вищий показник TNR, тим ефективніше WAF запобігає хибним спрацюванням, які можуть призвести до блокування легітимних користувачів та збоїв у роботі веб-додатку. Висока якість виявлення є невід'ємною частиною забезпечення безперебійної роботи системи та комфортного користувацького досвіду. Розрахунок FPR здійснюється за формулою:

$$FPR = \frac{TN}{TN + FP} \quad (2.2)$$

де TN (True Negatives) – кількість правильно пропущених легітимних запитів. Це звичайний трафік, який WAF не заблокував;

TN (Negative) – загальна кількість усіх легітимних запитів у наборі даних;

FP (False Positives) – кількість помилково заблокованих легітимних запитів. Це трафік, який мав бути пропущений, але WAF помилково його заблокував.[22]

Збалансована точність (Balanced Accuracy, BA) - це інтегральний показник, який надає збалансовану оцінку ефективності WAF, враховуючи обидві попередні

метрики. Він розраховується як середнє арифметичне між TPR і FPR, забезпечуючи об'єктивну оцінку продуктивності незалежно від пропорцій легітимних і шкідливих запитів у наборі даних.

Вибір саме цих метрик є ключовим, тому що WAF-рішення повинно не тільки ефективно блокувати атаки, але й безперешкодно пропускати легітимний трафік. Можна зробити висновок, збалансована точність найкраще відображає здатність WAF досягти необхідного компромісу між високим рівнем захисту та стабільною роботою системи.

Для проведення аналізу було використано два набори даних: легітимний та шкідливий, який містив загрози. Набір легітимних запитів був ретельно розроблений для моделювання реальних сценаріїв. Він включав 1 040 242 HTTP-запити, зібрані з 692 реальних веб-сайтів. Ці дані були отримані шляхом імітації дій типового користувача, таких як реєстрація, пошук товарів, додавання їх у кошик, завантаження файлів. Різноманітність типів веб-сайтів наприклад, платформи електронної комерції зі складною логікою дозволила імітувати широкий набір сценаріїв. Такий підхід забезпечив високу точність тестування, він не включав в собі синтетичних наборів даних, він включав перевірку всіх компонентів HTTP-запитів, що є поширеною причиною хибних спрацювань у реальних додатках.

Набір шкідливих запитів містив 73 924 запитів, які охоплювали широкий спектр поширених векторів атак, як-от: SQL-ін'єкції, міжсайтовий скриптинг (XSS), XML External Entity (XXE), path traversal, виконання команд, а також вразливості Log4Shell та Shellshock. .[22]. Цей набір запитів був зібраний з авторитетних відкритих джерел, в тому числі, з репозиторію WAF Payload Collection на GitHub, створеного компанією mgm security partners GmbH. Цей підхід забезпечив використання різноманітних реальних та актуальних зразків атак, що дозволило провести всебічну оцінку ефективності WAF-рішень в умовах, максимально наближених до реальних кіберзагроз. В дослідженні взяли участь як комерційні, так і open-source WAF-рішення, серед яких: Microsoft Azure WAF,

CloudFlare WAF, AWS WAF, NGINX ModSecurity, F5 NGINX AppProtect, open-appsec, F5 BIG-IP Advanced WAF, Fortinet FortiWeb, Google Cloud Armor.

Аналіз результатів дослідження показав, що існує значна різниця в ефективності WAF. Ключові висновки стосовно показника якості безпеки (True Positive Rate, TPR) - open-appsec досяг найвищого показника TPR - 99.368% з налаштуваннями за замовчуванням, наступними за ним були F5 NGINX App Protect із профілем Strict Rule (97.849%) та Microsoft Azure WAF (97.526%), при цьому останній не був оновлений до нового OWASP CRS. .[22]

Рисунок 2.6 – Результати досліджень по показнику TPR (True Positive Rate)

Результати інших протестованих WAF-рішень виявилися менш успішними. Imperva Cloud WAF продемонстрував найнижчу якість безпеки, досягнувши лише 11.97% TPR з налаштуваннями за замовчуванням. Це свідчить про значні прогалини в ефективності багатьох WAF, оскільки вони неспроможні протидіяти навіть відомим і давно опублікованим векторам атак. Такий результат підкреслює ризик, пов'язаний з використанням рішень, що не забезпечують постійної адаптації до загроз.[22]

Аналіз показнику якості виявлення (detection quality) також виявив значні відмінності між WAF-рішеннями. Найнижчий показник хибних спрацювань продемонстрував Imperva, досягнувши ідеального результату в 0.009%. Cloudflare показав близький до нього результат у 0.062%. Рішення open-appsec у профілі Critical, своєю чергою у FPR у 0.81%. Протилежні результати показали інші WAF. Google Cloud Armor із дуже високим показником хибних спрацювань - 50.283%, а також Microsoft Azure WAF із найвищим FPR – 54.242%. Такі результати свідчать про те, що ці продукти потребують значних зусиль для налаштування, що ускладнює їхнє застосування в реальних робочих середовищах. .[22]

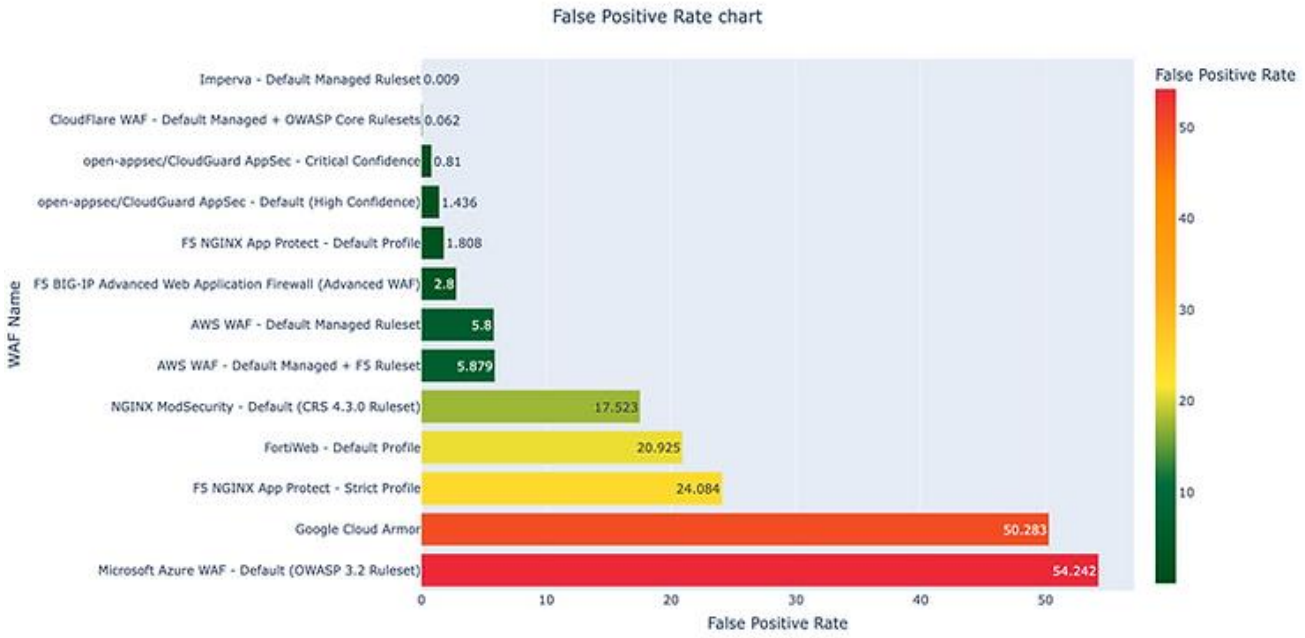


Рисунок 2.7 – Результати дослідження по показнику FPR (False Positive Rate)

Для відображення ефективності WAF використовується графічне представлення, яке показує взаємозв'язок між якістю безпеки Security Quality) та якістю виявлення Detection Quality. Ця візуалізація дозволяє інтуїтивно зрозуміти, наскільки добре кожне рішення досягає подвійних цілей: блокування шкідливих запитів та пропуску легітимних. Рішення, що розташовані у верхньому правому куті графіка, демонструють найкращий баланс між цими двома показниками. Це означає, що вони забезпечують високий рівень захисту при мінімальній кількості хибних спрацювань, що є критично важливим для стабільної роботи системи в реальних умовах.[22]

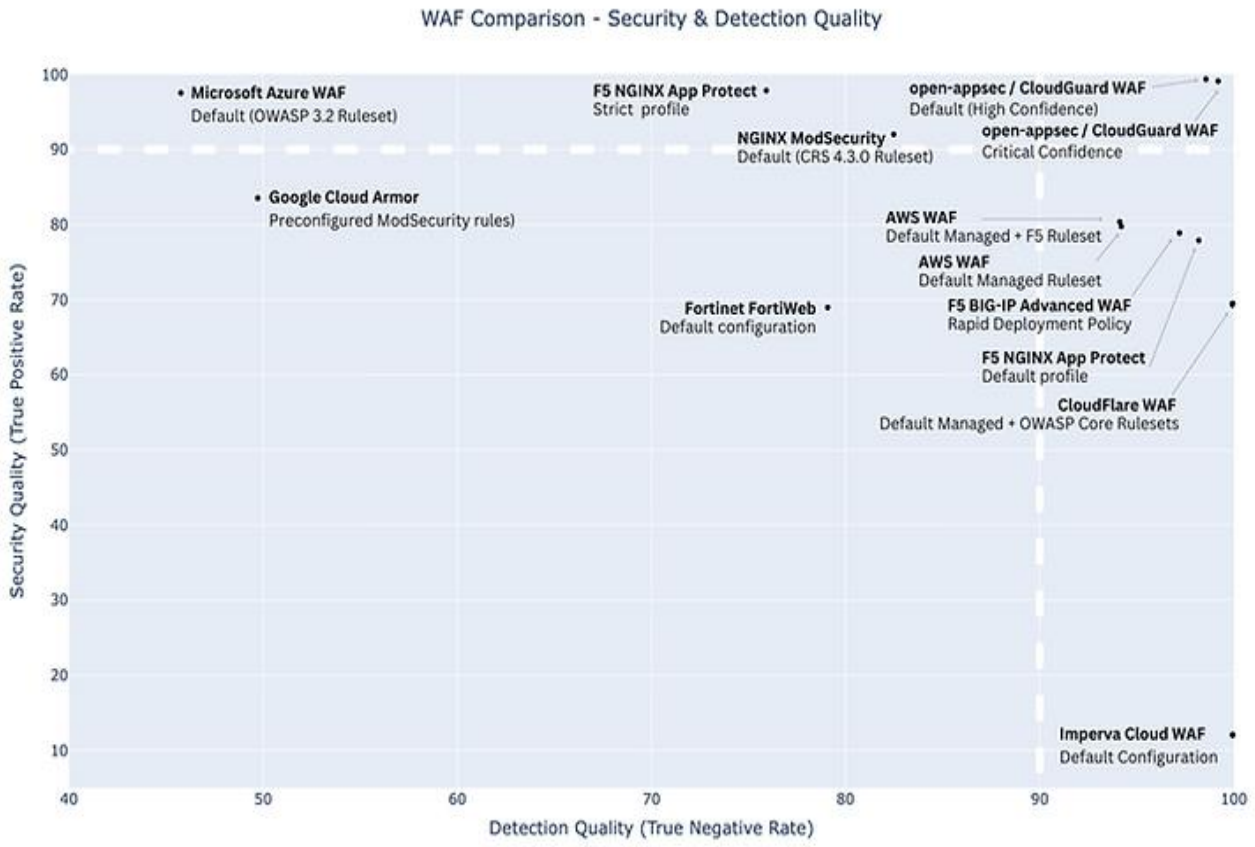


Рисунок 2.8 - Взаємозв’язок між Security Quality та Detection Quality

Збалансована точність надає більш цілісну оцінку продуктивності WAF, оскільки враховує два ключові показники, якість безпеки і якість виявлення. Вищий показник збалансованої точності свідчить про те, що WAF-рішення є оптимальним. Саме цей показник дозволяє визначити, наскільки добре WAF може функціонувати в реальних умовах, в яких особливо важливі як функції захисту, так і відсутність хибних спрацювань.[22]

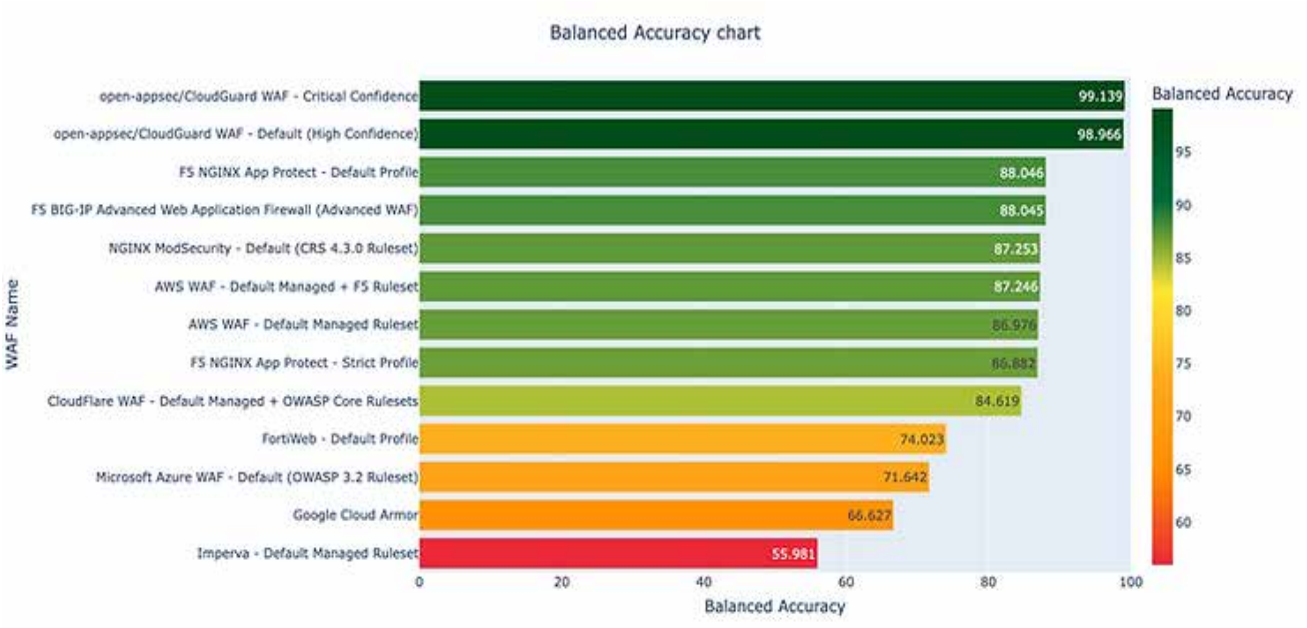


Рисунок 2.9 – Результати дослідження по показнику Balanced Accuracy

Аналіз показника збалансованої точності підтвердив лідерство open-appsec. За цим критерієм open-appsec WAF у профілі Critical очолив рейтинг з результатом 99.139%. Другий найкращий результат показав open-appsec WAF у профілі Default з показником 98.966%. WAF від Imperva мав найнижчу збалансовану точність, що становила лише 55.981%.[22]

Висновки до другого розділу

Було проведено комплексний аналіз сучасних підходів до забезпечення резильєнтності веб-додатків, за допомогою використання брандмауерів веб-додатків (WAF). Розглянуто основні архітектурні моделі інтеграції WAF у мережу, включаючи режими моніторингу, моста та зворотного проксі, причому останній був визнаний найбільш ефективним для забезпечення повноцінного захисту.

Для практичної реалізації було обрано open-source WAF open-appsec, що базується на технологіях машинного навчання. Цей вибір був обґрунтований його ключовими перевагами, які відрізняють його від традиційних рішень: здатність до адаптації та проактивного захисту від атак нульового дня, низька кількість хибних спрацювань та висока ефективність

Було детально проаналізовано внутрішню архітектуру open-appsec та його триетапний підхід до аналізу трафіку, що поєднує розбір, пошук індикаторів та контекстне ухвалення рішень. Це дозволило зрозуміти, чому open-appsec демонструє виняткову збалансовану точність.

На основі отриманих висновків можна стверджувати, що open-appsec є оптимальним рішенням для побудови надійного, адаптивного та ефективного захисту веб-додатків, що повністю відповідає сучасним вимогам резильєнтності.

3. ПРОЕКТУВАННЯ ТА МОДЕЛЮВАННЯ WEB APPLICATION FIREWALL

3.1 Складові тестового стенду

З метою практичної перевірки ефективності обраного брандмауера веб-додатків open-appsec та демонстрації його здатності протистояти сучасним загрозам, було налаштовано просте тестове середовище. Основними компонентами тестового середовища є дві віртуальні машини, розгорнуті за допомогою програмного забезпечення VirtualBox. Вибір цього інструменту обґрунтований його безкоштовністю, та легкістю налаштування. Кожна з віртуальних машин виконує певну функцію. Перша віртуальна машина (WAF-шлюз) - виконує роль захисного шлюзу, на якому розгорнуто NGINX та open-appsec WAF. Ця машина працює під управлінням Ubuntu Server 24.04.3 LTS (як рекомендованої та стабільної платформи для open-appsec). Для забезпечення ефективної роботи WAF враховано його мінімальні системні вимоги: 2 vCPU та 4 ГБ оперативної пам'яті.

Друга віртуальна машина - OWASP BWA (OWASP Broken Web Applications Project) - це колекція спеціально вразливих веб-додатків. Головна мета програми – зібрати в одне ціле всі спеціально вразливі веб-додатки, щоб надати можливість ентузіастам та спеціалістам безпеки практикуватися у взломі та захисті веб-додатків. Додатково, цей додаток має передінстальований модуль Modsecurity

Така конфігурація дозволяє порівняти різницю в рівні захисту, порівнюючи результати атак, здійснених з WAF і без нього та дослідити рівень захисту двох WAF.

3.2 Налаштування тестового середовища

Налаштування тестового середовища виконувалося в 3 етапи. На першому етапі було здійснено інсталяцію операційної системи на віртуальну машину (ВМ 1), призначену для функціонування захисного шлюзу. В головному екрані VirtualBox натискаємо на «Створити». У вікні, що відкриється налаштовуємо ім’я віртуальної машини, обираємо необхідний iso-image, тип операційної системи, папку, в якій буде зберігатися створена віртуальна машина і натискаємо Next

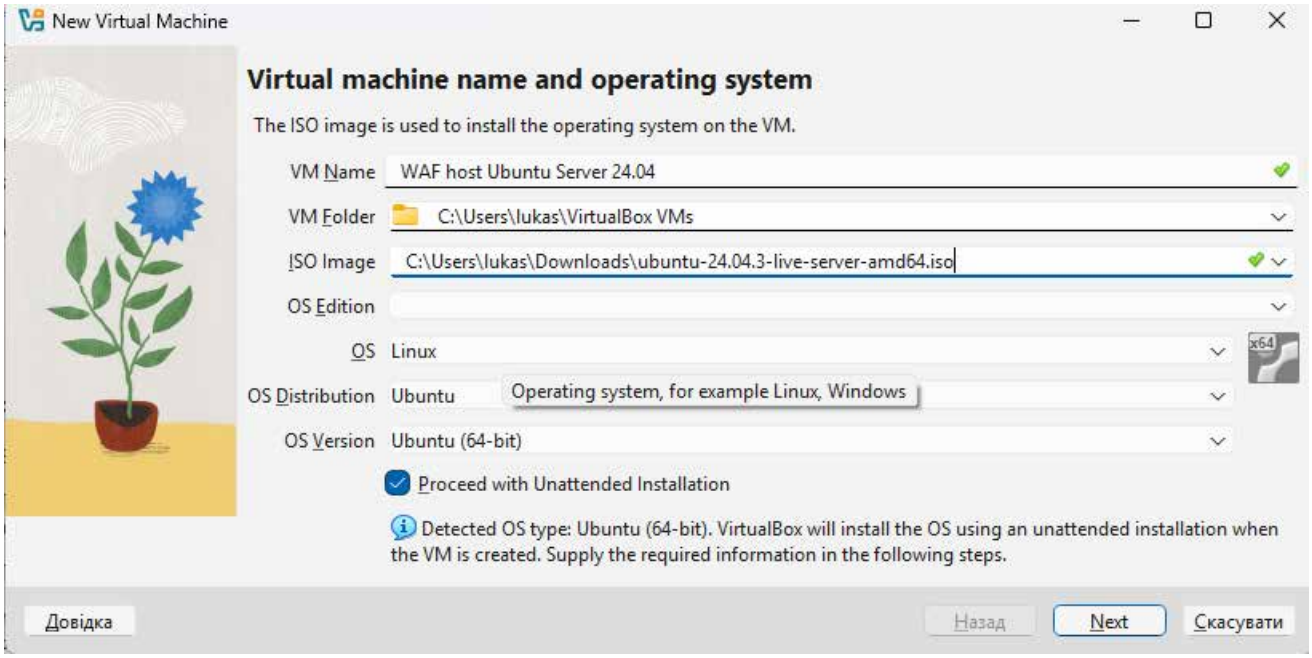


Рисунок 3.1 – Початковий етап створення віртуальної машини

У наступному вікні налаштовуємо ресурси для віртуальної машини, встановлюємо обсяг оперативної пам’яті 4Гб, кількість віртуальних процесорів (vCPU) - 2, об’єм диску - 30 Гб. Таких ресурсів буде достатньо для забезпечення стабільної та ефективної роботи open-appsec у тестовому режимі. Ці ресурси гарантують, що двигун машинного навчання WAF зможе коректно виконувати аналіз трафіку в режимі реального часу, не спричиняючи вузьких місць у продуктивності.

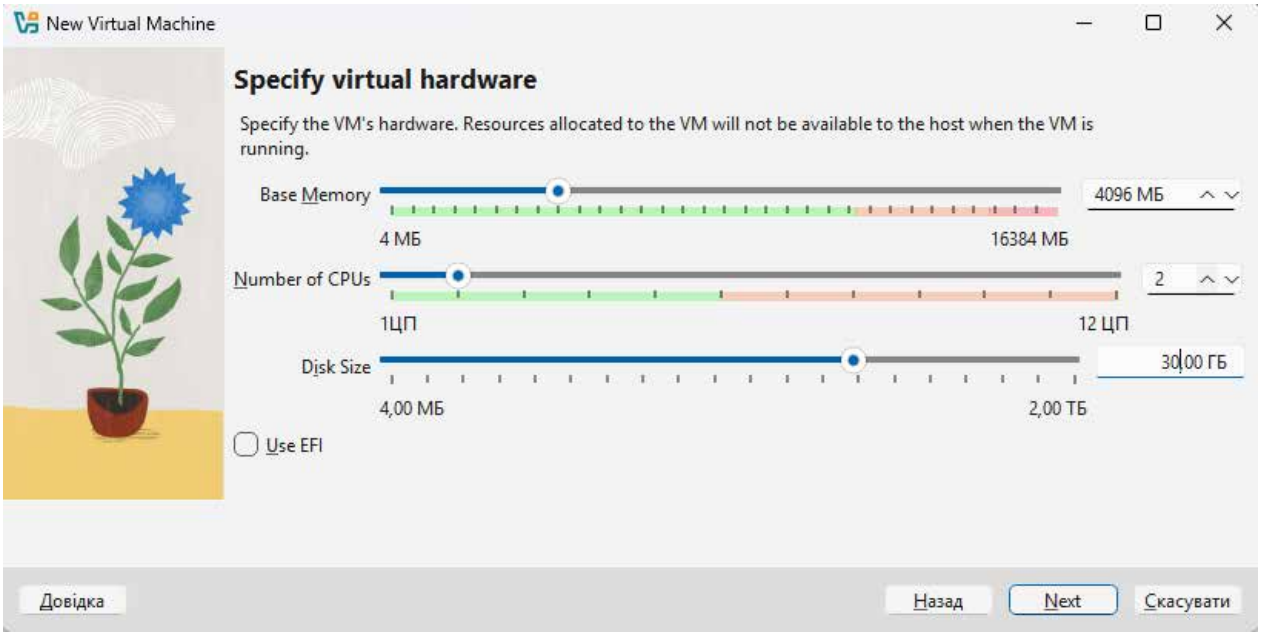


Рисунок 3.2 – Налаштування ресурсів для віртуальної машини

У фінальному вікні перевіряємо внесені налаштування та натискаємо «Закінчити» підтверджуючи створення віртуальної машини

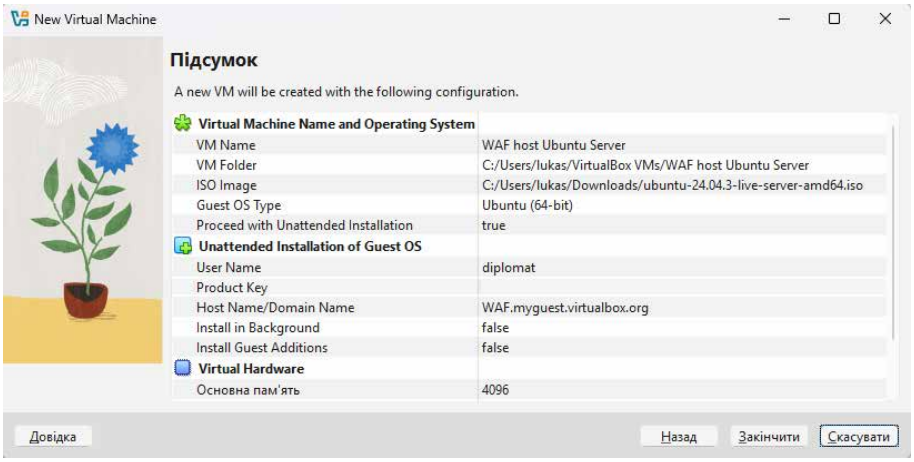


Рисунок 3.3 – Завершення налаштування ВМ

Після завершення налаштувань параметрів віртуальної машини і її успішного створення, переходимо в її налаштування натиснувши «Settings» та в мережевих налаштуваннях змінюємо тип мережевого адаптера на «Bridged Adapter»

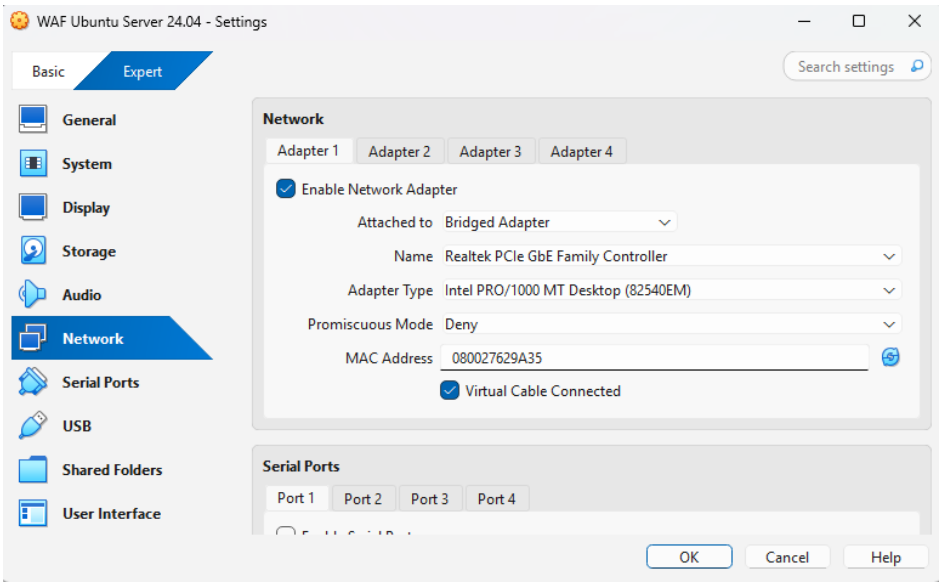


Рисунок 3.4 – Зміна мережевих налаштувань віртуальної машини

Після чого вмикаємо віртуальну машину і проводимо налаштування гостьової ОС. Обираємо мову системи:

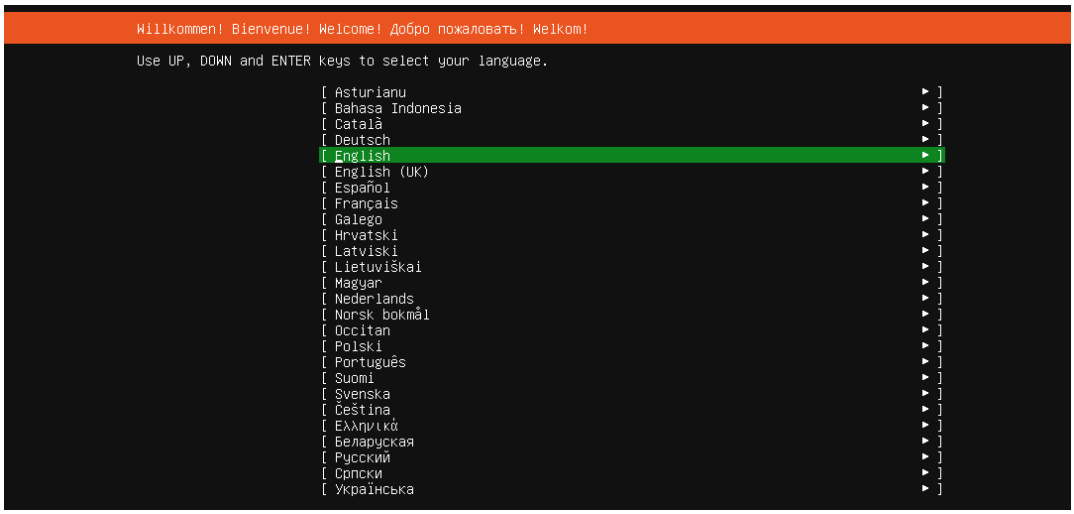


Рисунок 3.5 – Вибір мови

На наступному кроці обираємо звичайний тип інсталяції (Ubuntu Server)

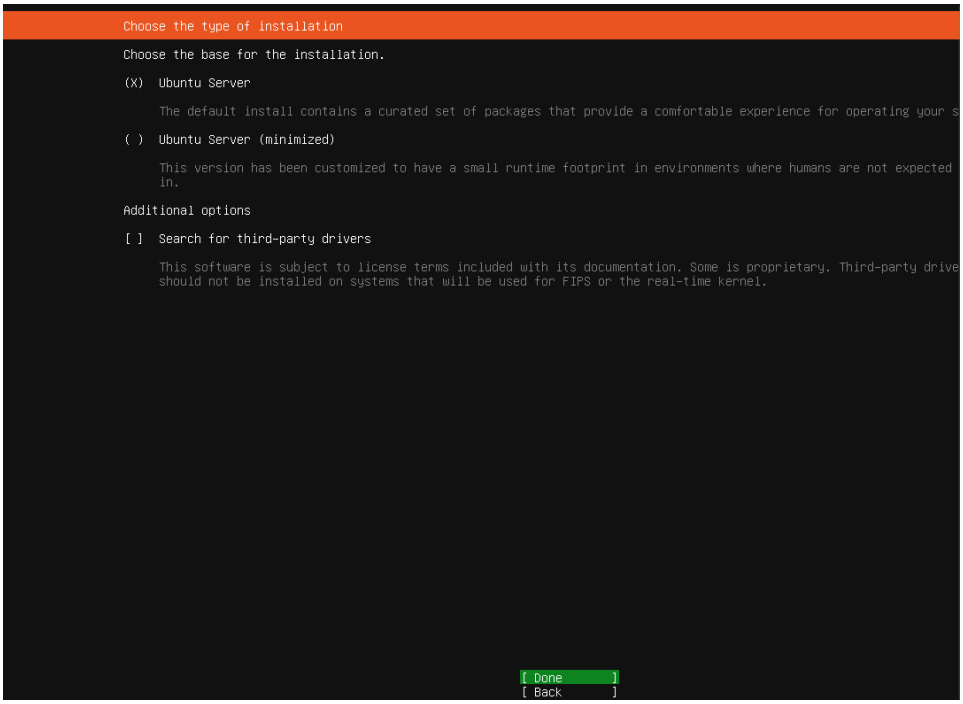


Рисунок 3.6 – Вибір типу інсталяції

На наступному кроці налаштовуємо мережу, обираючи метод отримання адреси «Manual» та конфігуруємо IP-адресу, мережу, default-gateway та dns-сервер

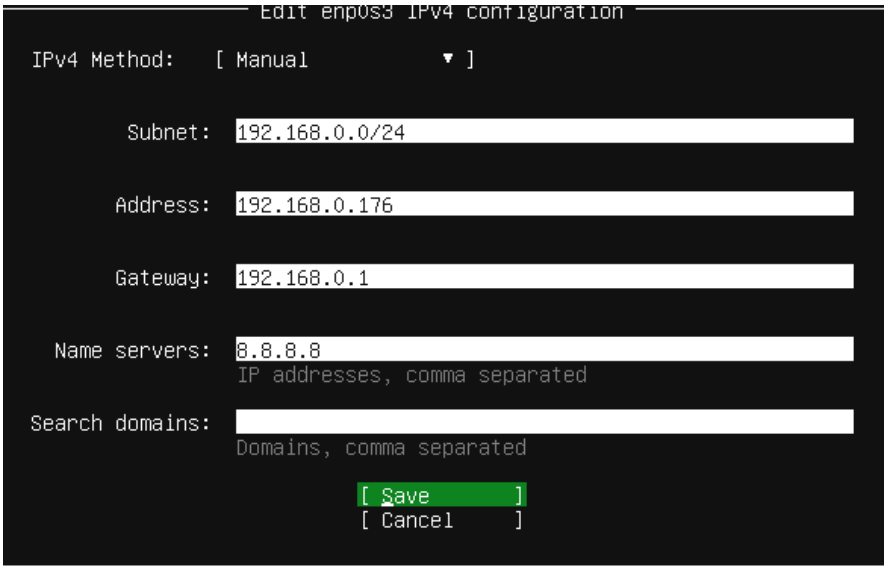


Рисунок 3.7 – Налаштування мережевих параметрів

На фінальному етапі, налаштовуємо користувача для входу в систему і перезавантажуємо віртуальну машину для завершення встановлення гостьової ОС

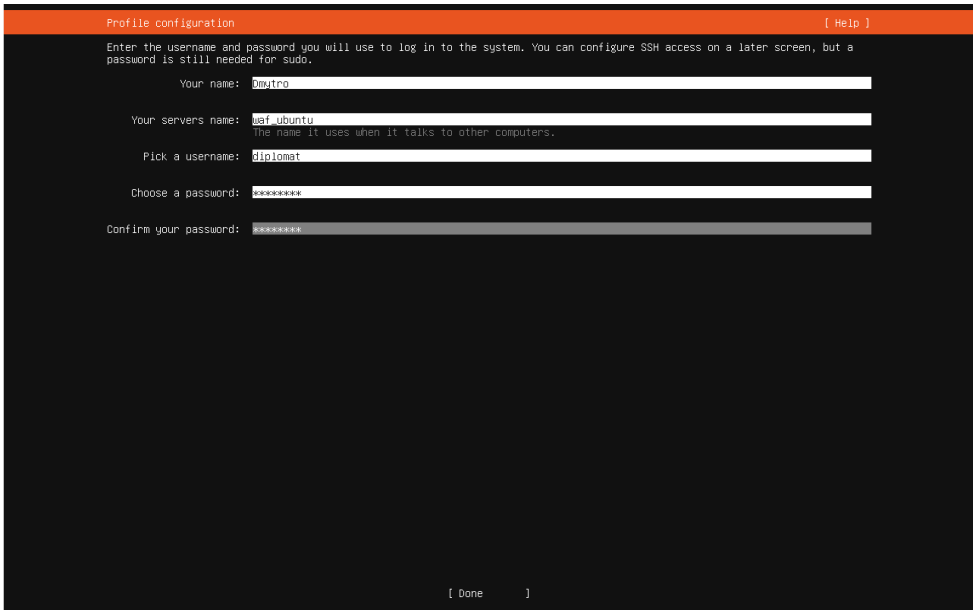


Рисунок 3.8 – Налаштування користувача

На другому етапі, було виконано розгортання віртуальної машини із спеціально вразливим додатком OWASP BWA та передвстановленим модулем Modsecurity. У середовищі Virtual Box обираємо «Machine» - «New», налаштовуємо ім’я та тип операційної системи, визначаємо ресурси віртуальної машини – 1024 Мб пам’яті та 1vCPU. На етапі налаштування віртуального диску необхідно обрати пункт «Use an Existing Virtual Hard Disk File» і додати необхідний нам, попередньо завантажений файл віртуального диску «OWASP Broken Web Apps-cl1»

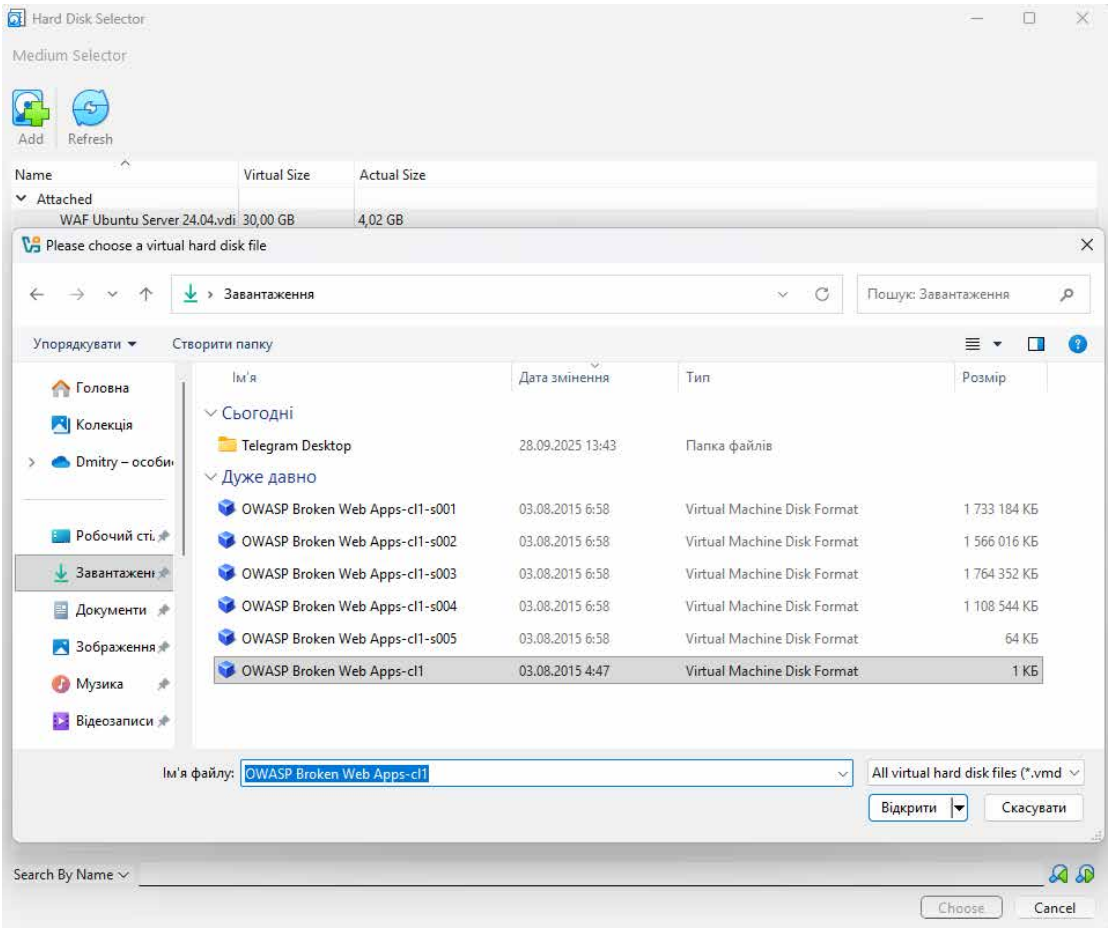


Рисунок 3.9 – Вибір файлу віртуального диску

Перед запуском віртуальної машини змінюємо налаштування мережевого адаптера в підменю «Network» із випадаючого списку обираємо «Bridged Adapter»

На третьому етапі налаштування тестового стенду було проведення розгортання Nginx Proxy Manager та open appsec. Конфігурація і подальший моніторинг подій безпеки WAF буде здійснюватися через центральний веб-інтерфейс (SaaS варіант розгортання). Переходимо на SaaS портал використовуючи і створюємо акаунт. Після чого, перед нами відкриється вікно із етапами налаштувань і інструкцією для розгортання агента

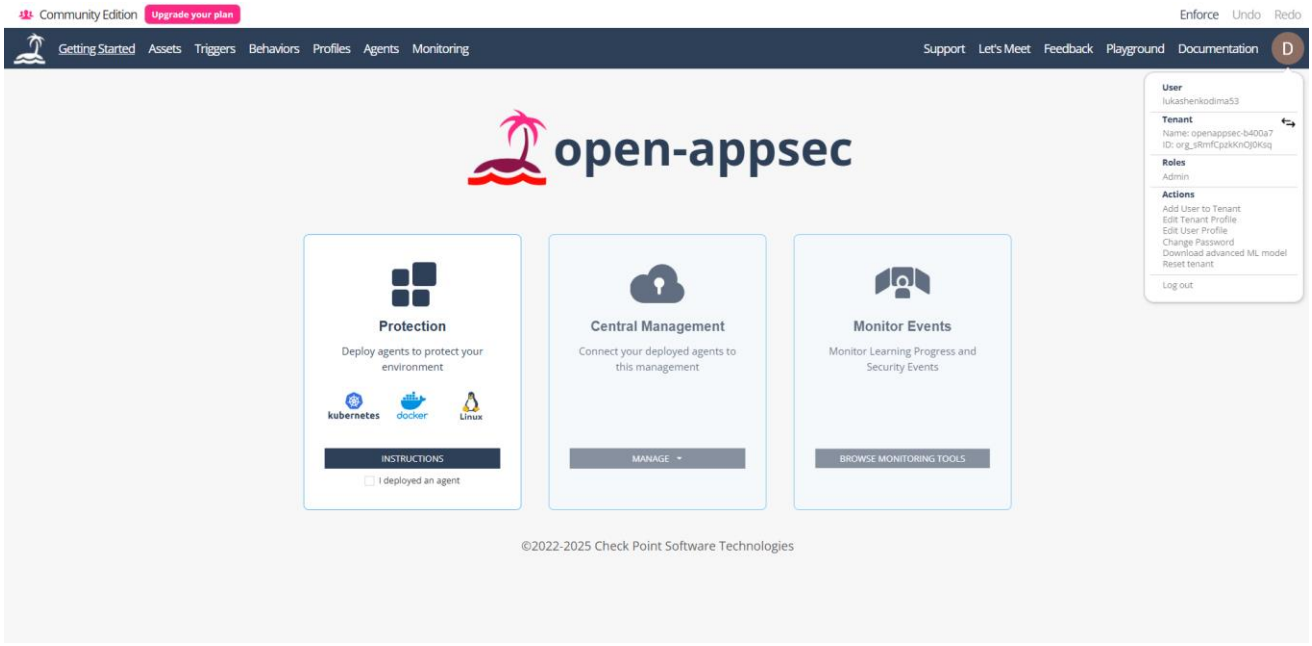


Рисунок 3.10 – Реєстрація на порталі SaaS

Переходимо на вкладку «Profiles», обираючи «Docker Profile» із випадаючого списку, потрапляємо у вікно конфігурації нового профілю. Налаштовуємо ім’я профілю, в Sub Type із випадаючого списку обираємо «Nginx Proxy Manager + open-appsec». У вікні Management, обираємо опцію «This management». При увімкненій опції «This management» усі налаштування open-appsec здійснюються централізовано через хмарну консоль. Локальні конфігураційні файли на самому агентові більше не використовуються. Це

забезпечує максимальну простоту адміністрування та моніторингу, перетворюючи WAF на повноцінну Software-as-a-Service (SaaS). Центральний WebUI надає повний контроль над усіма параметрами, логуванням та комплексним аналізом подій безпеки. При увімкненій функціїї «Declarative configuration» основна конфігурація WAF зберігається та редагується у локальних декларативних файлах наприклад, Kubernetes YAML, CRD або конфігураційних файлах. Веб-інтерфейс у цьому випадку виконує допоміжну функцію: він відображає усі налаштування у режимі read only, а також надає зручний моніторинг статусу агентів, логів та подій безпеки. Для завершення конфігурації профілю натискаємо клафішу «Enforce» та копіюємо Authentication Token [23]

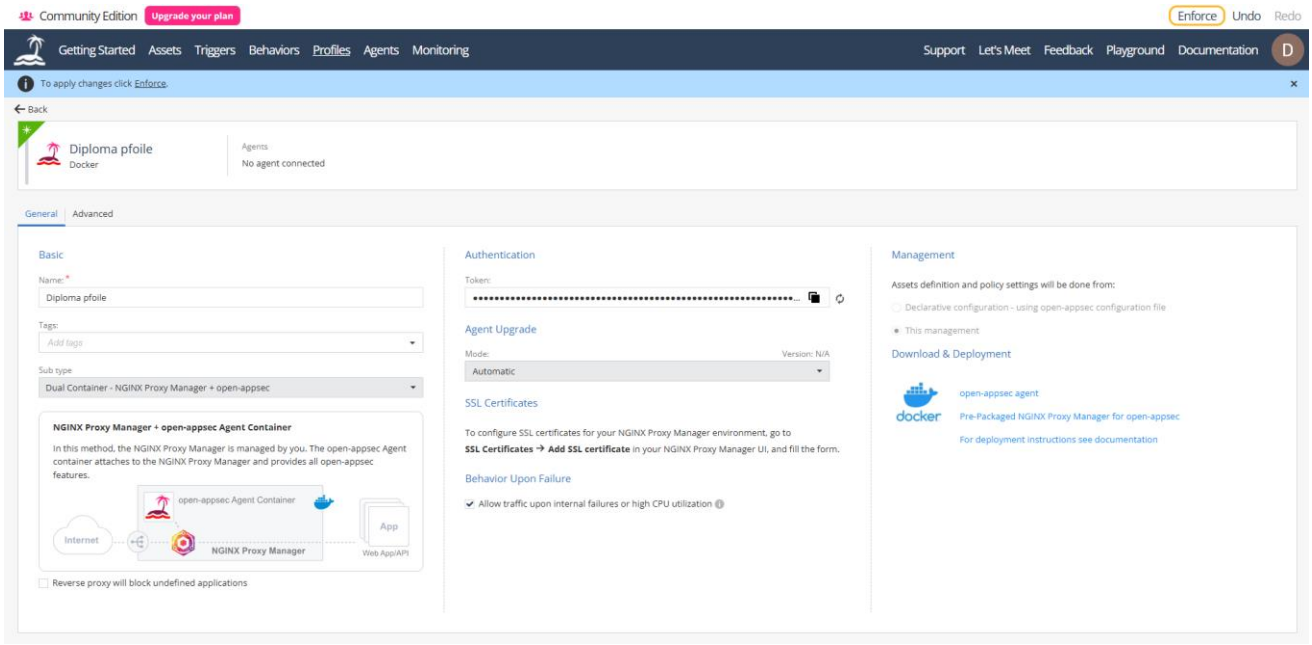


Рисунок 3.11 – Конфігурація Profile

Переходимо до налаштування віртуальної машини Ubuntu Server ВМ-1. Першим кроком необхідно виконати встановлення Docker Service (Docker Engine і Docker Compose). Враховуючи виклики, пов'язані з ручним налаштуванням зовнішніх репозиторіїв та вирішенням проблем із цілісністю GPG-ключів, було прийнято рішення використати офіційний інсталяційний скрипт Docker get-docker.sh. Видаляємо будь-які наявні або попередні версії docker і завантажуємо офіційний скрипт [23]

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
curl -fsSL https://get.docker.com -o get-docker.sh
```

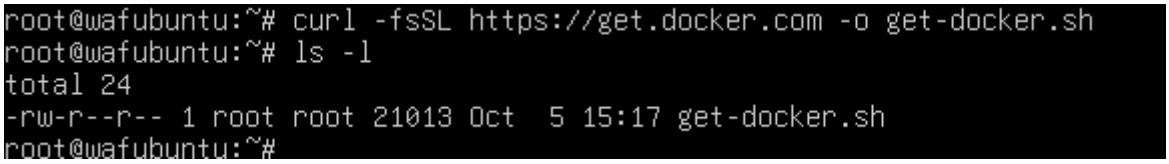


Рисунок 3.12 – Завантажений скрипт на інсталяцію Docker

Скрипт автоматично додає офіційний репозиторій, GPG-ключ та інсталює Docker Engine, Docker CLI та Docker Compose Plugin. Перевіряємо, чи успішно встановився докер

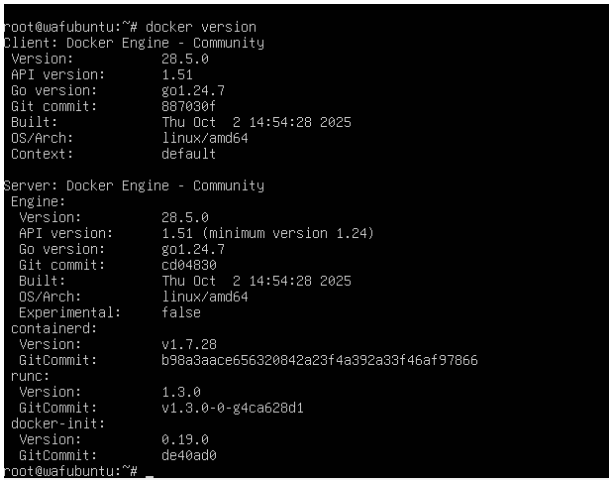


Рисунок 3.13 – Успішно встановлений Docker

На наступному етапі було виконано фінальне розгортання захисного шлюзу на ВМ 1 (WAF-шлюз). Розгортання виконувалося за допомогою Docker Compose, який забезпечує інтеграцію open-appsec безпосередньо у NGINX Proxy Manager (NPM). Створюємо робочий каталог та завантажуємо конфігурацію, яка містить опис спеціалізованого контейнера NPM із вбудованим модулем open-appsec [23]

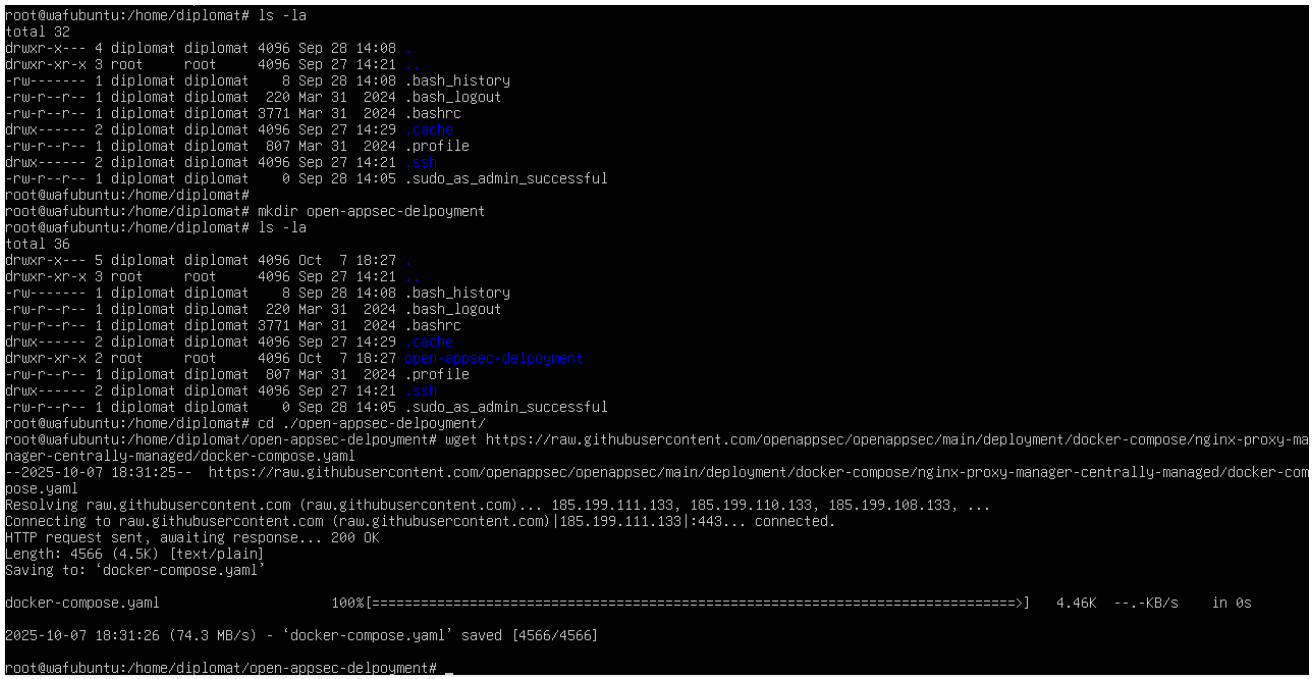


Рисунок 3.14 – Створення директорії та завантаження файлу

Завантажуємо файл змінних .env. Файл .env є ключовим елементом розгортання. Він містить змінні середовища, які керують підключенням агента open-appsec до центральної системи керування. [23]

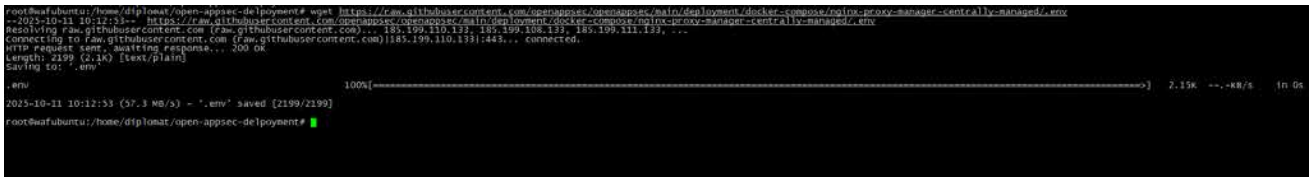


Рисунок 3.15 – Завантаження файлу змінних .env

Для реалізації моделі централізованого керування SaaS та підготовки до тестування були налаштовані наступні критичні змінні. APPSEC_AGENT_TOKEN - обов'язковий параметр. Встановлює унікальний токен профілю розгортання, отриманий з центрального WebUI open-appsec. Значення токену, було раніше скопійовано при налаштуванні Profile. USER_EMAIL (Опціонально) - вказує електронну адресу для асоціації розгортання. Для спрощення тестового стенду та мінімізації проблем з мережевою конфігурацією, було прийнято рішення розгорнути OWASP Juice Shop як додатковий контейнер безпосередньо на ВМ 1 (WAF-шлюз), використовуючи функціонал COMPOSE_PROFILES [23]

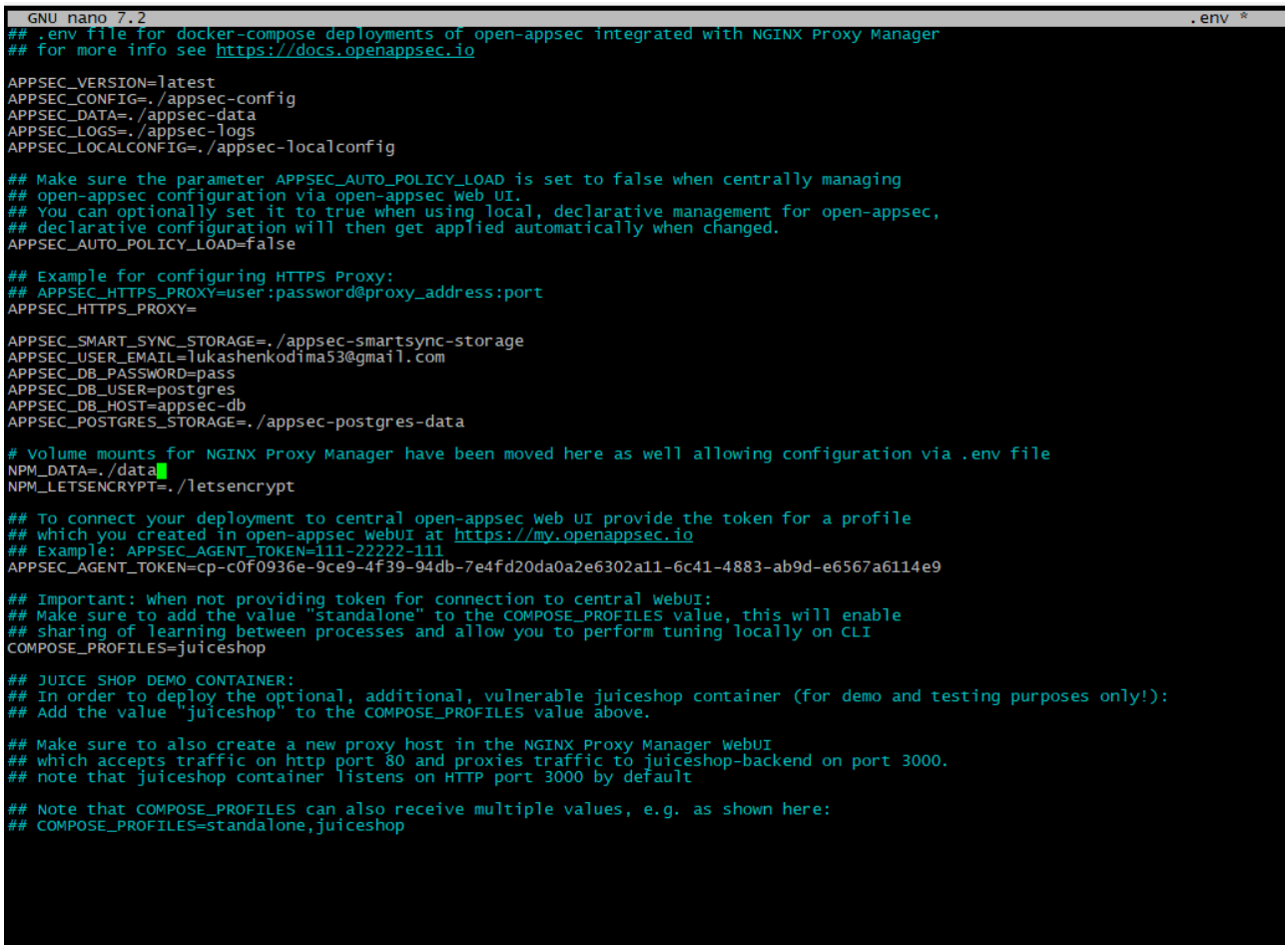


Рисунок 3.16 – Редагування файлу .env

Додатково, в файлі можна налаштувати наступні параметри:

- APPSEC_VERSION - дозволяє чітко вказати версію контейнерів open-appsec для розгортання, замість використання версії за замовчуванням;
- APPSEC_AUTO_POLICY_LOAD - При встановленні значення true дозволяє агенту автоматично застосовувати зміни у локальному конфігураційному файлі (local_policy.yaml) без необхідності перезапуску. Релевантно для практик DevOps та CI/CD (Continuous Integration/Continuous Delivery), де потрібне динамічне оновлення політик безпеки;
- APPSEC_HTTPS_PROXY - використовується для конфігурації проміжного HTTP(S) проксі-сервера, через який агент open-appsec буде здійснювати вихідні з'єднання (наприклад, до центрального WebUI). Необхідний у корпоративних або лабораторних середовищах із суворими політиками вихідних мережевих з'єднань, забезпечуючи трафіку відповідність вимогам безпеки;
- NPM_DATA - вказує локальний шлях для зберігання конфігураційних даних NGINX Proxy Manager. Забезпечує збереження конфігурації проксі-менеджера (хости, налаштування SSL) між перезапусками та оновленнями контейнерів;
- NPM_LETSENCRYPT - вказує локальний шлях для зберігання сертифікатів SSL, згенерованих Let's Encrypt. Гарантує персистентність криптографічних ключів та сертифікатів, уникнення повторної генерації при кожному перезапуску. [23]

На наступному кроці робимо deployment і додаткову перевірку.

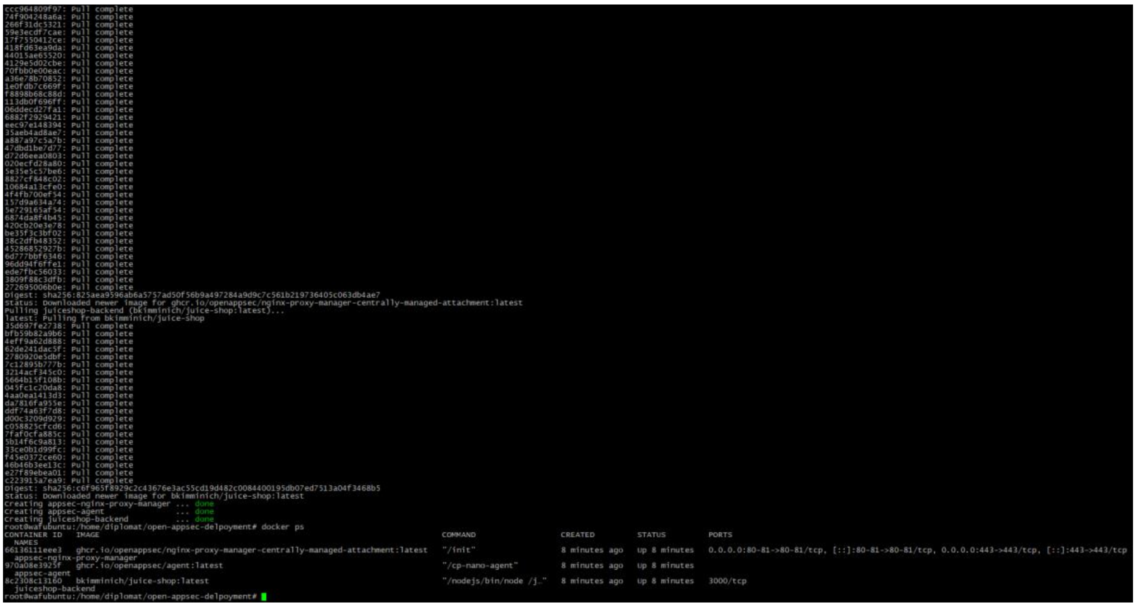


Рисунок 3.17 – Deployment контейнерів

Після проведеного deployment, переходимо за адресою http://[VM1-IP-адреса]:81. Перед нами відкриється вікно логіну Nginx Proxy Manager. Використовуємо логін admin@example.com та пароль changeme для того, щоб увійти в систему.

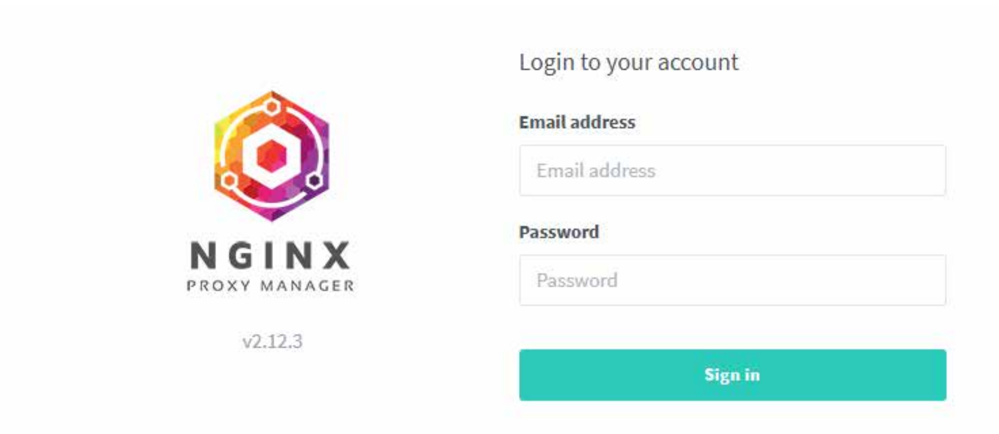


Рисунок 3.18 – Nginx Proxy Manager login page

Для забезпечення гнучкості тестового середовища було створено два окремих проксі-хости у NGINX Proxy Manager. Перший хост спрямовував трафік на внутрішній контейнер juiceshop-backend для фази навчання. Другий хост проксував трафік на ВМ 2 (DVWA) за IP-адресою, що дозволило проводити порівняльне тестування, використовуючи один WAF-шлюз для захисту різних цільових додатків

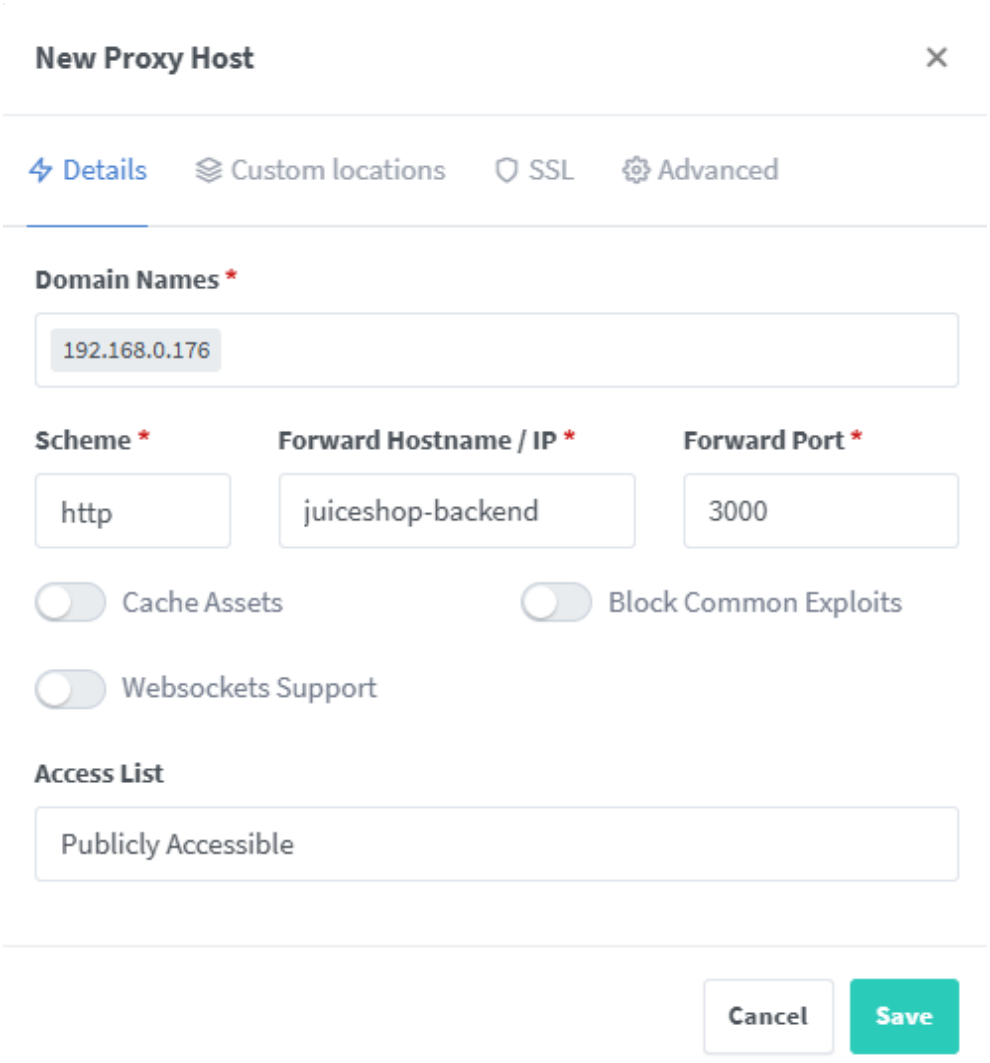


Рисунок 3.19 – Налаштування proxy-host для додатку juiceshop

Аналогічним чином додаємо другий proxy-host – ВМ 2 із встановленим додатком OWASP BWA

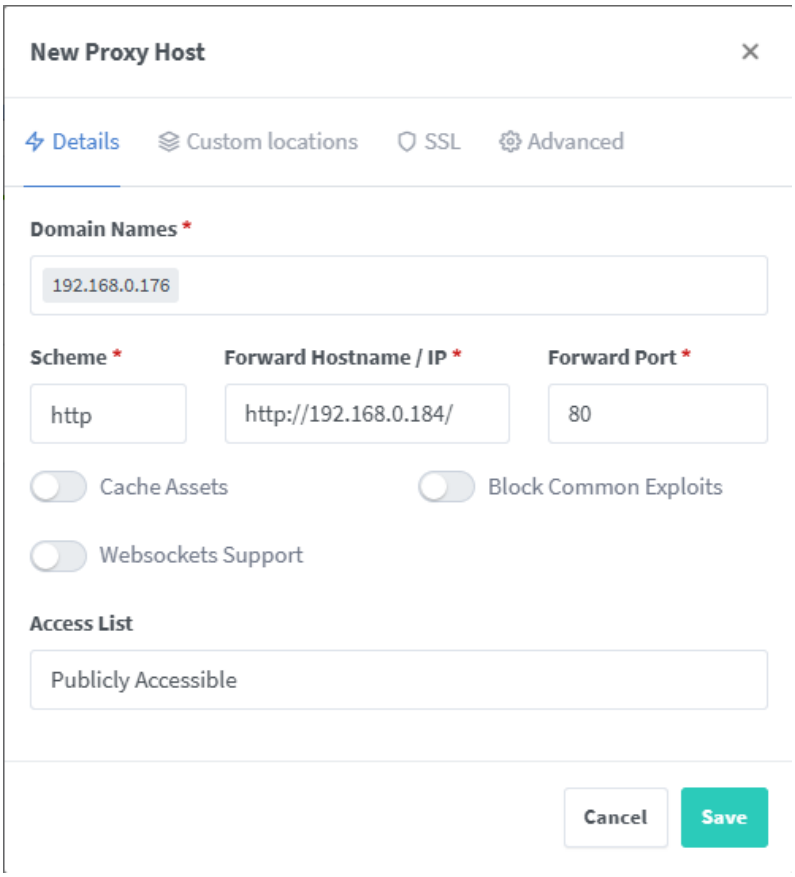


Рисунок 3.20 - Налаштування proxy-host для ВМ2

Після створення двох proxy-хостів в центральній WebUI консолі open-appsec було створено два «Assets». Актив у системі open-appsec - це логічна одиниця, яка керує моделлю нормальної поведінки, політикою безпеки та статусом навчання, незалежно від WAF-агента, який його обслуговує. Кожен актив має свою власну політику безпеки, модель навчання та статус навчання (learning progress)

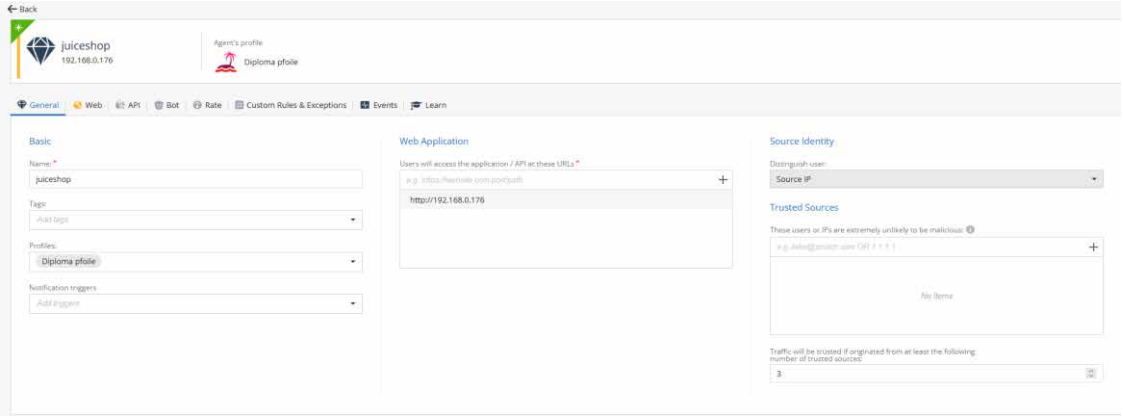


Рисунок 3.21 – Створення Asset для додатку juice shop

При конфігурації Asset у вкладці «Web» було налаштовано режим роботи WAF «Learn/Detect». На початковому етапі WAF має працювати в режимі «Learn». Це дозволяє ML-ядру пасивно аналізувати весь легітимний трафік входи, форми, API-запити, що надходить до Juice Shop. Якщо одразу активувати блокування, будь-яка невідома поведінка навіть легітимна, але ще не вивчена може бути помилково заблокована. Режим «Learn» мінімізує ризик хибних спрацювань.

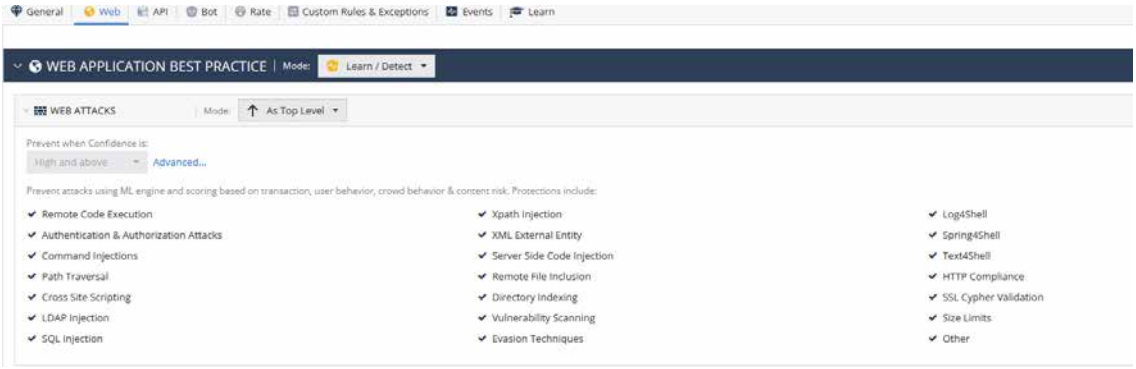


Рисунок 3.22 – Налаштування режиму роботи WAF

Аналогічним чином було налаштовано другий Asset для додатку OWASP BWA, змінюючи URL

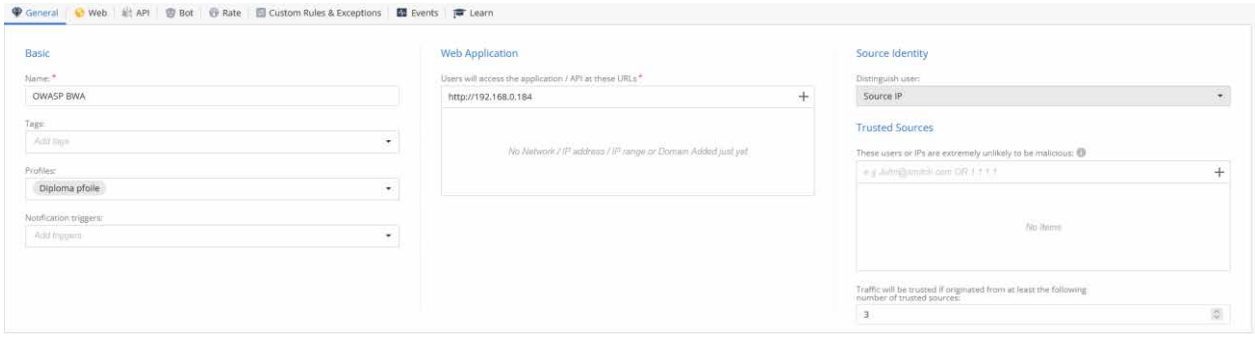


Рисунок 3.23 – Налаштування другого Asset для додатку OWASP BWA

На завершальному етапі налаштувань виконуємо встановлення Advanced AI Model. Advanced AI Model забезпечує глибокий, аналіз трафіку, та базується на складних нейронних мережах. Замість базового аналізу структури, advanced модель здійснює глибокий аналіз на основі складних алгоритмів, що підвищує якість виявлення. Для цього необхідно завантажити файл «Advanced ML model» із WebUI

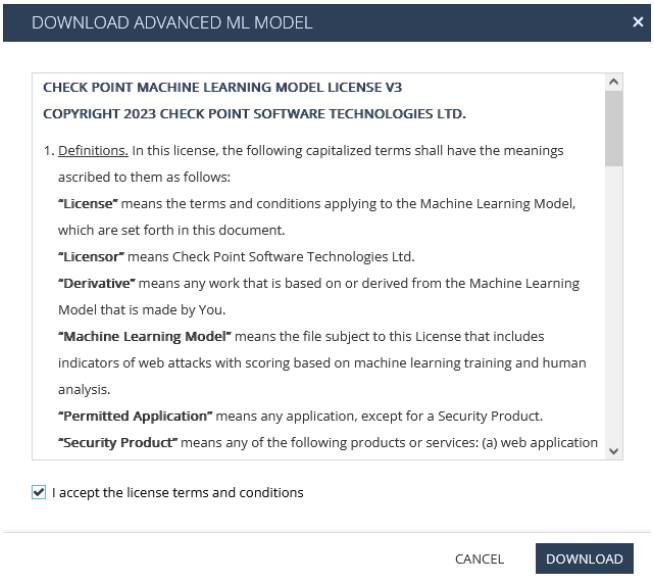


Рисунок 3.24 – Завантаження Advanced AI model

Використовуючи протокол SFTP завантажуємо файл на BM1 зі встановленим WAF open-appsec в попередньо створену директорію open-appsec-advance-model

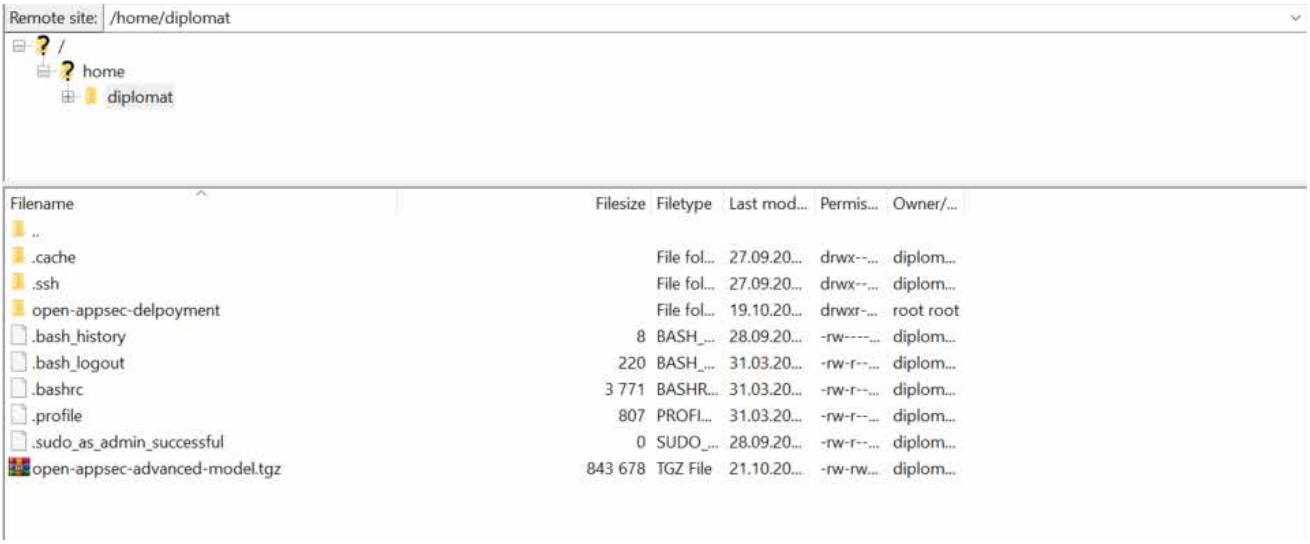


Рисунок 3.25 – Завантаження файлу на BM1

Було відредаговано файл docker-compose.yaml для того, щоб в подальшому WAF міг використовувати нову модель

```
  volumes:
    - ${APPSEC_CONFIG}:/etc/cp/conf
    - ${APPSEC_DATA}:/etc/cp/data
    - ${APPSEC_LOGS}:/var/log/nano_agent
    - ${APPSEC_LOCALCONFIG}:/ext/appsec
    - ./open-appsec-advanced-model/open-appsec-advanced-model.tgz:/advanced-model/open-appsec-advanced-model.tgz:rw
  command: /cp-nano-agent --standalone
  command: /cp-nano-agent
```

Рисунок 3.26 – Редагування файлу docker-compose

В результаті, у вікні агентів WebUI ми можемо побачити, що WAF використовує нову Advanced модель.

3.2 Тестування рішення

Сканування вразливостей вважається найбільш ефективним способом перевірки сайту на наявність великого списку відомих вразливостей і виявлення слабких місць у безпеці веб-додатків. Сканування можна використовувати як частину автономної оцінки, або як частину безперервної загальної стратегії моніторнигу безпеки. Сканери веб-вразливостей працюють за рахунок автоматизації декількох процесів. До них відносяться пошук і сканування додатків, виявлення поширених вразливостей. Існує два основні підходи до сканування – пасивний і активинй. При пасивному скануванні виконуються перевірки шляхом простого перегляду елементів сайту, щоб визначити, чи є вони вразливими. Активне сканування – це моделювання на сайт справжніх атак. Надійність сканера вразливостей залежить від методів тестування, які він запускає, а також від того, як часто оновлюється його логіка сканування. Автоматичні сканери здатні виявляти найрізноманітніші типи вразливостей, але не можуть повністю замінити тестування на проникнення під керівництвом спеціалістів.

У рамках даного дипломного дослідження, сканування вразливостей використовується як перший етап тестування. Основна мета полягає у проведенні порівняльного аналізу ефективності двох принципово різних типів Web Application Firewall (WAF): сигнатурного та ML-WAF. Сигнатурний WAF (ModSecurity): представляє традиційний, більш статичний підхід, що використовує фіксовані набори правил (Core Rule Set). Сканування дозволить генерувати ідентичні набори шкідливого трафіку (SQLi, XSS), на спеціально вразливий додаток, який буде захищений двома WAF кожен з яких буде підключений окремо. Це дасть змогу отримати кількісні показники (відсоток заблокованих атак), необхідні для обґрунтованого висновку про переваги того чи іншого підходу, особливо проти обфускованих (замаскованих) атак, де очікується найбільша різниця в ефективності.

Тестування ефективності WAF було проведено на ідентичних, ізольованих тестових стендах для забезпечення об'єктивності порівняльного аналізу. Дослідження включало два основні об'єкти: WAF на основі машинного навчання (open-appsec з Advanced AI Model) та WAF на основі сигнатур (ModSecurity з OWASP CRS). Для WAF на основі машинного навчання (open-appsec) було проведено спеціальну фазу навчання. Це є необхідним етапом, оскільки Advanced AI Model не має жодних правил за замовчуванням, а замість цього будує модель нормальної поведінки цільового веб-додатку. В якості сканера було обрано безкоштовний але ефективний продукт OWASP ZAP. Для проведення тестів було обрано додаток OWASP Mutillidae II в колекції спеціально вразливих додатків. Тести проводилися без брандмауеру веб-додатків за адресою http://192.168.0.184/mutillidae/ та через міжмережевий екран за адресою http://192.168.0.176/mutillidae/. OWASZP Zed Attack Proxy – один з найпопулярніших інструментів безпеки з відкритим кодом. Коли ZAP сканує веб-додаток він створює карту сторінок додатку. Потім він записує запити і відповіді, надіслані на кожну сторіку, і створює попередження, в окремому вікні. Додаток має інтуїтивно зрозумілий інтерфейс.

При скануванні вразливого додатку без міжмережевого екрану було виявлено 5 вразливостей високого рівня небезпеки (серед яких всі типи XSS атаки, SQL-ін’єкція, можливість віддаленого включення файлів, та можливість обходу шляху).

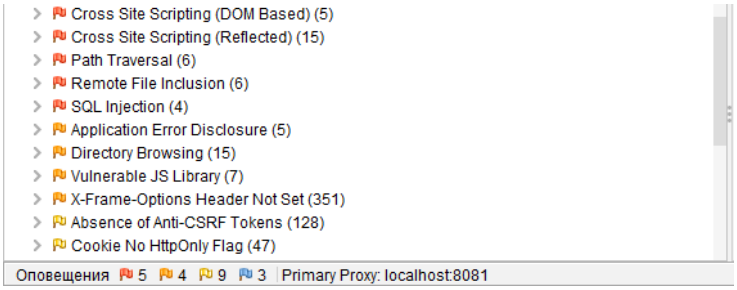


Рисунок 3.27 – Результати сканування незахищеного WAF вразливого додатку

Серед вразливостей, які OWASP ZAP відносить до середнього рівня небпезпеки найбільше було зафіксовано відсутність заголовку X-Frame (для захисту від атак типу Clickjacking) та вразливість перегляду списку каталогів веб-сайту. В нижньому правому вікні OWASP ZAP надає можливість ознайомитись з коротким описом виявлених вразливостей та переглянути URL.

Запускаємо ще одне автоматичне сканування додатку на вразливості через міжмережевий екран веб-додатків додаючи в поле «URL to attack» адресу http://192.168.0.176/mutillidae/

При скануванні вразливого додатку через міжмережевий екран open appsec не було виявлено вразливостей високого рівня небезпеки. При цьому, спостерігається суттєве зменшення числа вразливостей середнього та низького рівнів небезпеки.

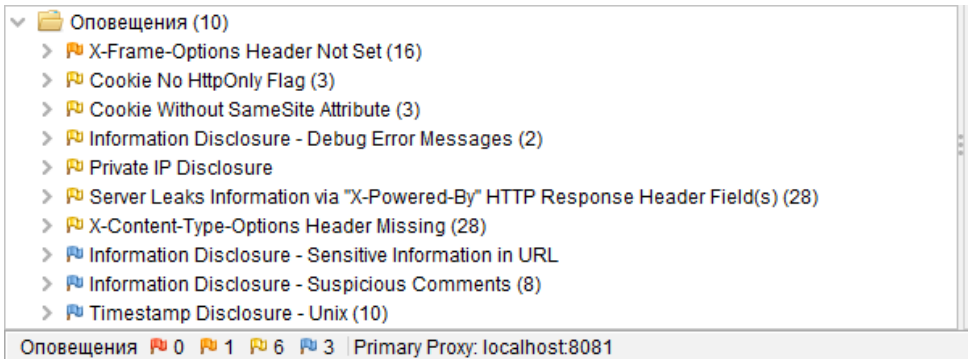


Рисунок 3.28 – Результати сканування вразливого веб-додатку через WAF open appsec

Додатково було проведено тестування через сигнатурний WAF ModSecurity, що слугував контрольним зразком для порівняння. У цьому випадку було зафіксовано 18 попереджень. Сканування під захистом open-appsec зафіксувало лише 10 попереджень. Ця менша кількість наочно демонструє різницю між двома WAF. Open-appsec із його Advanced AI Model фокусується на критичних проблемах, пов'язаних із логікою та контентом запитів, та відсіює

менш релевантні попередження, пов'язані з неідеальним налаштуванням HTTP-заголовків

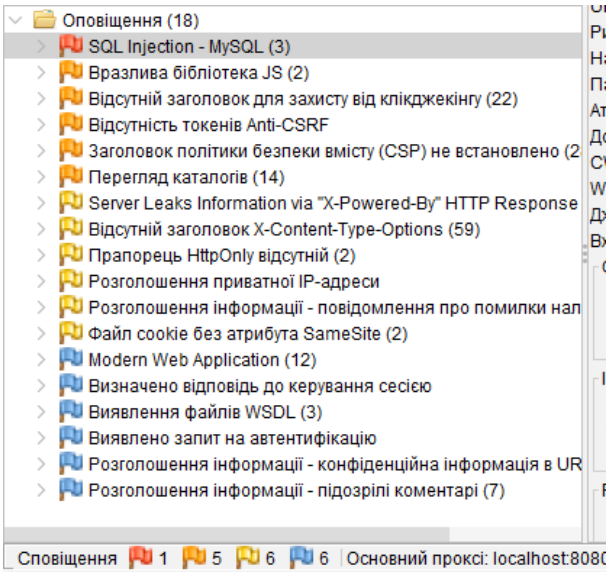


Рисунок 3.29 – Сканування додатку через сигнатурний WAF Modsecurity

На основі проведеного сканування, вираховуємо ефективність двох WAF. За основу беремо загальну кількість запитів - 3000 згенерованих через OWASP ZAP. Припускаємо, що половина з них містила шкідливий payload, тоді як інша половина містила призначена для загального збору інформації, перегляду заголовків, скануванню каталогів

Таблиця 3.1 – Ключові показники для розрахунку метрик ефективності

Метрика	Modsecurity	Open-appsec
TP (True Positive успішно Блоковано)	1485 (99.0%)	1495 (99.6%)
FN Пропущено Атак	3 (0.11%)	0
TN (Успішно Пропущено)	1500-3 = 1485	1500
FP	10	3

Автоматизоване сканування OWASP ZAP не надає точних даних про кількість False Positive (FP), для порівняльного аналізу було використано моделювання показників FP, базуючись на якісному аналізі звітності ZAP та фундаментальних відмінностях у роботі WAF. Враховуючи, що ModSecurity є більш чутливим через більш жорстке налаштування сигнатур (18 попереджень), його показник FP було встановлено на рівні 10. Для ML-WAF open-appsec демонструє вищу точність класифікації, показник FP було встановлено на рівні 3

Беручи до уваги дані із таблиці, робимо розрахунок ключових метрик, TPR та FPR

$$TPR\ Modsecurity = \frac{TP}{TP + FN} = \frac{1485}{1500} = 99.8\%$$

$$TPR\ open - appsec = \frac{TP}{TP + FN} = \frac{1500}{1500} = 100\%$$

$$FPR\ Modsecurity = \frac{TP}{TP + FN} = \frac{10}{1500} = 0.66\%$$

$$FPR\ open - appsec = \frac{TP}{TP + FN} = \frac{3}{1500} = 0.2\%$$

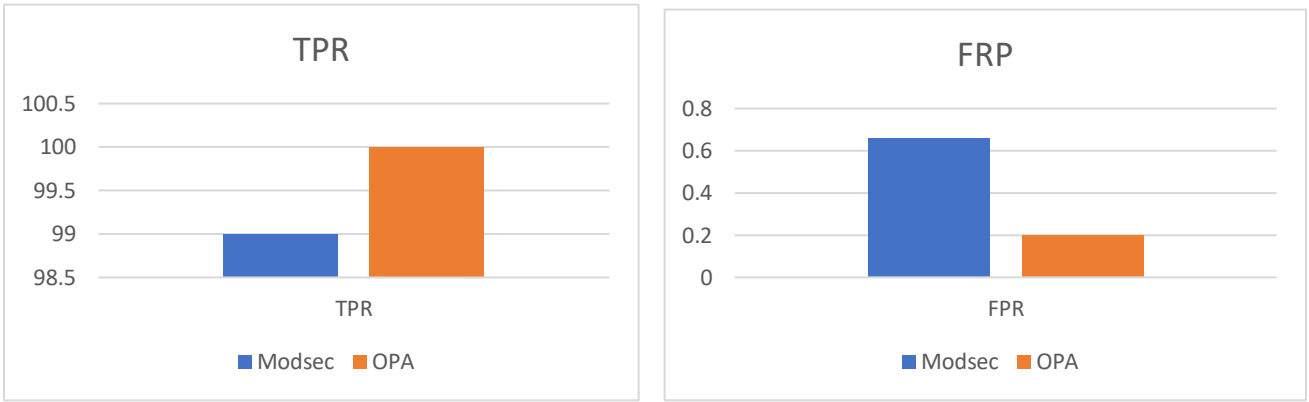


Рисунок 3.30 – Порівняльна діаграма показників TPR та FRP

На другому етапі тестування, після завершення сканування з використанням OWASP ZAP, було розпочато цілеспрямоване моделювання атак. Метою цього етапу було отримання результатів, які дозволять показати перевагу адаптивного підходу open-appsec над статичним захистом ModSecurity.

Спробуємо провести SQL-ін’єкцію, що дозволяє обійти процес аутентифікації у вразливому додатку bWAPP. При здійсненні входу користувач вводить свої облікові дані, після чого генерується SQL-запит: SELECT * FROM `heroes` WHERE login = 'alice' AND password = 'loveZombies'.

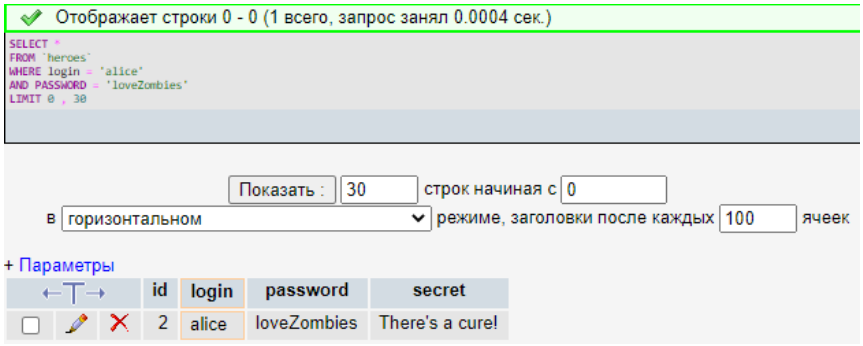


Рисунок 3.31 – Приклад легального SQL запиту

Якщо додаток вразливий до SQL-ін’єкції хакер може змінити запит SQL та увійти у систему під обліковим записом першого користувача в базі даних.

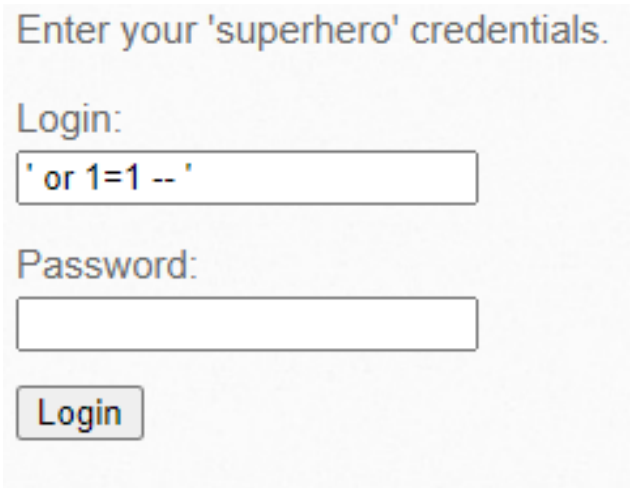


Рисунок 3.32 – Проста експлуатація ByPass SQL-ін’єкції

В результаті таких вхідних даних SQL-запит буде змінено наступним чином: SELECT * FROM `heroes` WHERE login = '' or 1=1 -- '.

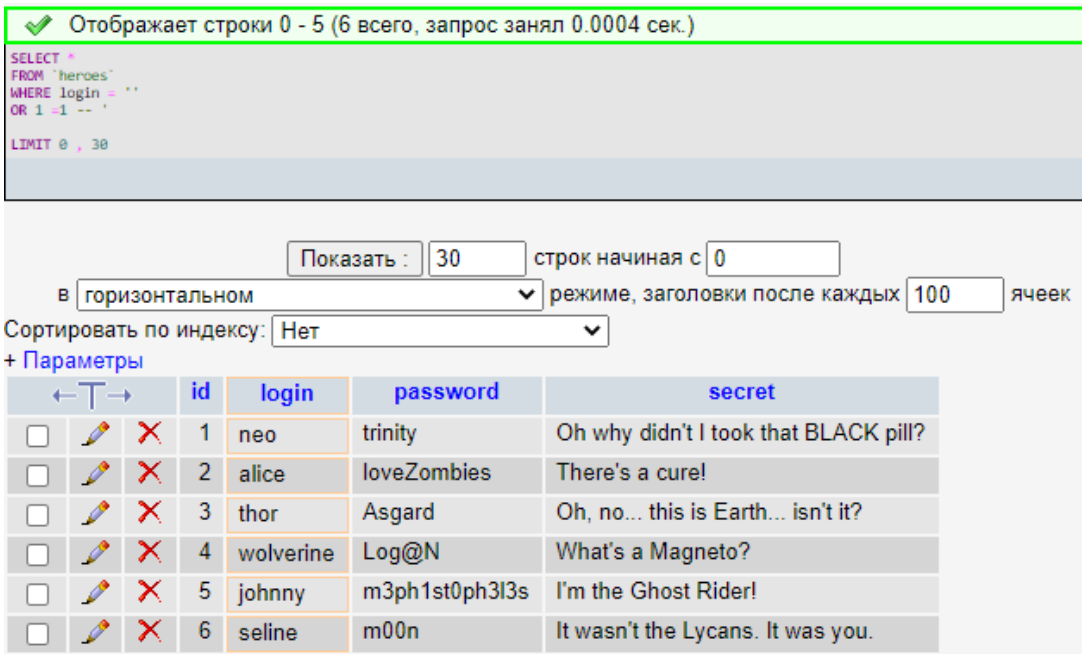


Рисунок 3.33 – Виконання зміненого SQL-запиту

В результаті виконання зміненого SQL-запиту буде здійснено вхід під обілковим записом першого користувача бази даних – neo.

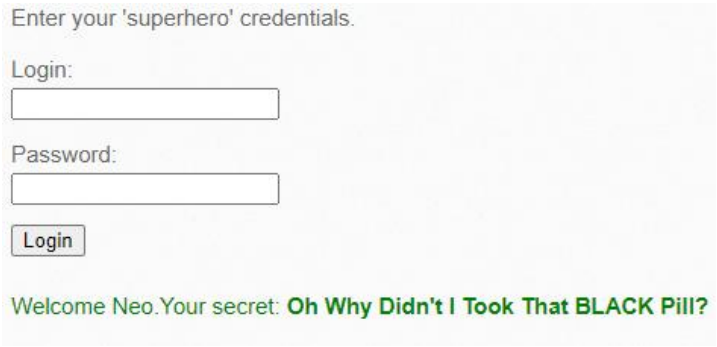


Рисунок 3.34 - Вдала експлуатація SQL-ін’єкції

Для перевірки роботи та правильності конфігуарції було здійснено спробу проведення аналогічної атаки на веб-додаток bWAPP, який перебував під захистом Web Application Firewall open appsec. Після введення шкідливого payload та ініціації запиту (натискання кнопки «Login») було отримано чітку відповідь від сервера: «Forbidden. You don’t have to access this resource». Цей результат підтверджує, що WAF успішно виявив і заблокував необфускований вектор атаки, демонструючи коректне спрацювання захисної політики.

Рисунок 3.35 – Заблокована атака

Результати успішного блокування можна переглянути в окремому вікні в Asset

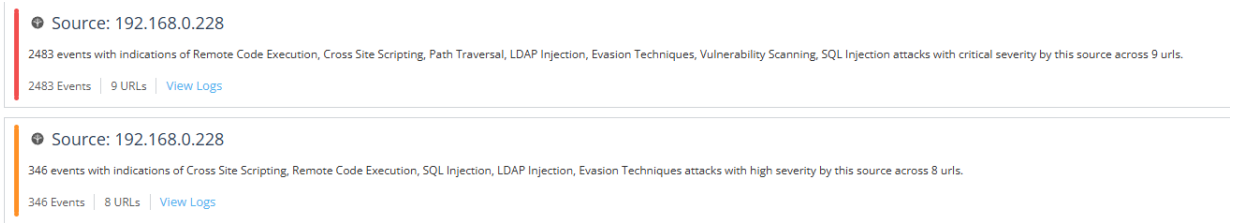


Рисунок 3.36 – Events в системному журналі open appsec WebUI

Після цього була проведена аналогічна атака на веб-додаток bWAPP, через WAF Modsecurity. Після введення корисного навантаження та натиснення кнопки «Login» отримуємо відповідь від серверу «Forbidden. You don’t have to access this resource». ModSecurity не використовує централізовану консоль управління, що сильно відрізняє його від WAF open-appsec. Для фіксації атак на веб-севрер ModSecurity використовує два типи журналів: журнал помилок (error.log) та журнал аудиту modsec_audit.log. Журнал помилок створюється при виявленні помилки або при спробі проветси атаку. Оскільки ModSecurity працює в парі з Apache всі журнали помилок (журнали помилок Apache+журнали помилок ModSecurity) створюються в одному файлі /var/log/apache2/error.logs.

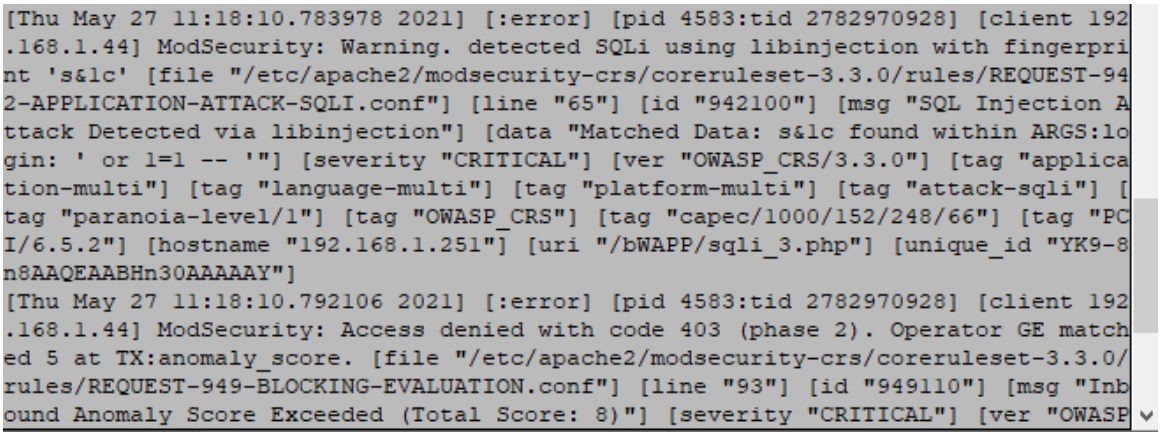


Рисунок 3.37 – Зафіксована SQL-ін’єкція в журналі помилок

З блокуванням необфускованої атаки обидва WAF вопралися. Проте варто відмітити, що, існує значна різниця у подальшій обробці, та логуванні події, що демонструє перевагу можливості централізованого управління open-appsec.

Спроба провести атаку міжсайтового скриптингу через обидва WAF у додатку DVWA

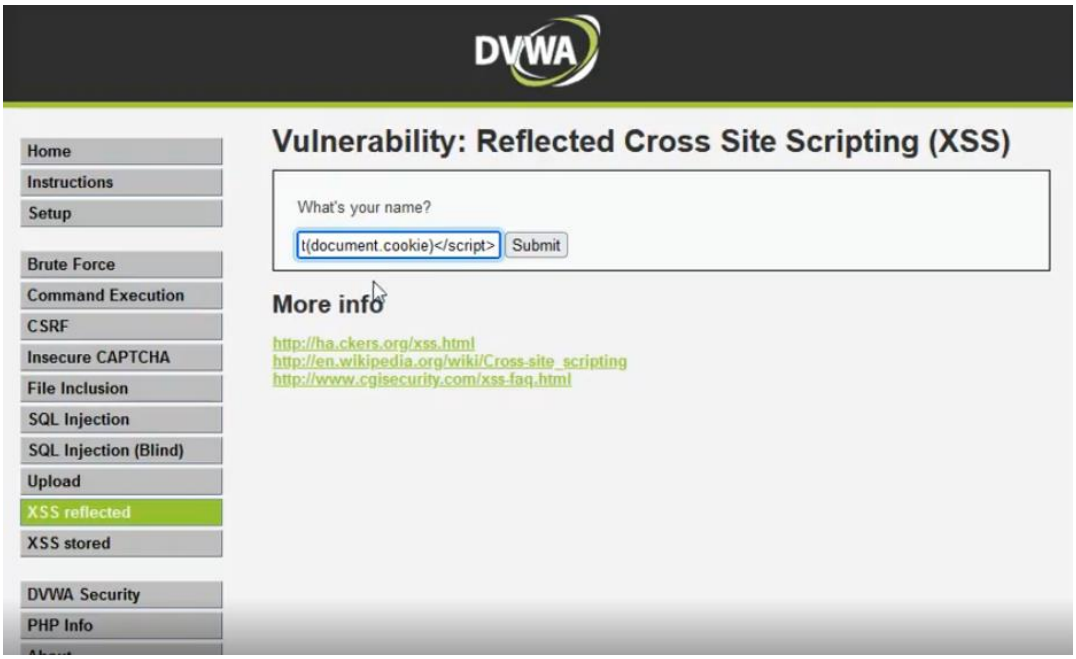


Рисунок 3.38 – Проста Reflected XSS атака

В результаті обидва WAF і сигнатуррний і ML успішно заблокують необфусковану атаку.

Наступним етапом стало цілеспрямоване тестування з використанням обфускованих векторів атак. Для цього взято запит admin'%20o%20r%201=1%23 з додатковими пробілами і коментарями, які маскують і розривають оператор or для обходу сигнатур WAF. Мета такого запиту змусити сервер виконати запит SELECT * FROM users WHERE login = 'admin' OR 1=1 -- ' AND password = '...'; Умова 1=1 завжди істинна, уся умова (login = 'admin' OR 1=1) стає істинною. Команда повертає усіх користувачів (або, принаймні, першого користувача, зазвичай admin), і аутентифікація успішно обходиться. Всього було виконано 5 обфускованих атак на обидва WAF, включно з обфускованною XSS атакю через використання вкладених тегів: <scr<script>ipt>

Таблиця 3.2 – Ключові показники отримані після проведення обфускованих атак для розрахунку метрик ефективності

Метрика	Modsecurity	Open-appsec
TP (True Positive успішно Блоковано)	4	5
FN (Пропущено Атак)	1 (0.11%)	0

Показники TN та FN є неактуальними, оскільки в заданому наборі даних не було легітмного трафіку який би міг оцінюватися обома WAF як false-positive. Відповідно, загальна метрика FPR не розраховується

$$TPR\ Modsecurity = \frac{TP}{TP + FN} = \frac{4}{5} = 80\%$$

$$TPR\ open - appsec = \frac{TP}{TP + FN} = \frac{5}{5} = 100\%$$

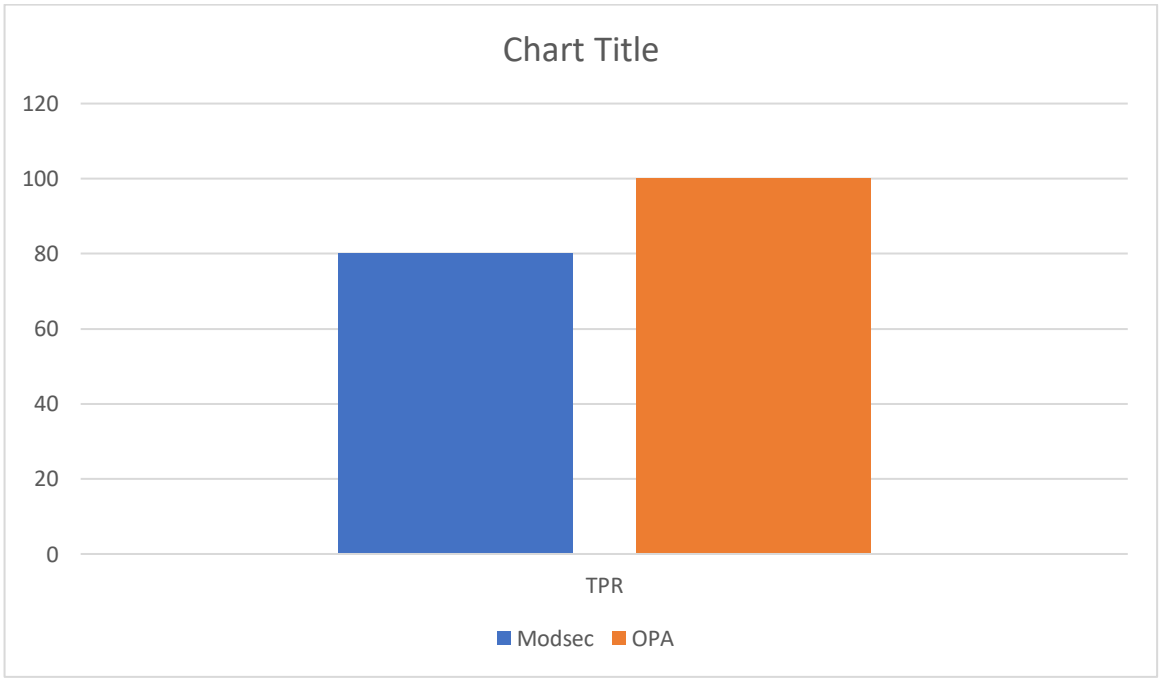


Рисунок 3.39 – Діаграма показника TPR при обфускованих атаках

Висновки до третього розділу

З метою практичної перевірки ефективності обраного брандмауера веб-додатків open-appsec та демонстрації його здатності протистояти сучасним загрозам, було налаштовано простий тестовий стенд. Архітектура стенду імітує реальне робоче середовище, де WAF виконує функцію зворотного проксі, захищаючи веб-додаток від зовнішніх атак. Основними компонентами тестового середовища є дві віртуальні машини, розгорнуті за допомогою програмного забезпечення VirtualBox.

Тестування ефективності WAF проведено на однакових тестових стендах для забезпечення об'єктивності порівняльного аналізу. Дослідження включало два основні об'єкти: WAF на основі машинного навчання (open-appsec з Advanced AI Model) та WAF на основі сигнатур (ModSecurity з OWASP CRS). Тестування системи проводилося в два етапи: на першому етапі були використанні засоби автоматизації пошуку веб-вразливостей. Сканування показло, що обидва рішення мають високий рівень захистку від необфускованих атак, при цьому, open-appsec за рахунок ML моделі менш чутливий і демонструє нижчий рівень FP. На другому етапі проводилося ручне тестування додатків на вразливості SQL-ін’єкції, міжсайтового скриптингу та проведені обфусковані атаки. Результати дослідження підтверджують ефективність ML WAF open appsec над сигнатурним Modsecurity. Open-appsec зберіг 100% TPR у всіх випадках, підтверджуючи свою здатність до класифікації аномальної поведінки незалежно від маскування. хоча жорстке налаштування сигнатур в Modsecurity має свої переваги і не потребує годин додаткового навчання. Open-appsec компенсує це прозорістю процесу машинного навчання та демонструє перевагу можливості централізованого управління open-appsec через SaaS підхід

ВИСНОВКИ

Метою даного дипломного проекту є дослідити підходи до забезпечення резильєнтності інформаційних систем на прикладному рівні моделі OSI, проаналізувати еволюцію актуальних загроз веб-додатків. Розглянули необхідність фокусу саме на прикладному рівні моделі OSI. На відміну від нижчих рівнів, які захищаються традиційними брандмауерами, загрози прикладного рівня постійно розвиваються і еволюціонують. Проведено комплексний аналіз архітектурних та програмних рішень забезпечення основних характеристик резильєнтності - захисту від загроз та здатності до відновлення. Визначено, що WAF та NGFW надають найбільший ефективний захист. WAF, як вузькоспеціалізований інструмент, відрізняється від NGFW здатністю глибоко аналізувати веб-трафік, що робить його незамінним для протидії цільовим атакам, таким як SQL-ін'єкції та міжсайтовий скриптинг. Аналіз списку OWASP Top 10-2021 підтвердив актуальність загроз на прикладному рівні та їхню еволюцію, пов'язану з архітектурними недоліками та новими технологіями. Це дозволило мені зробити висновок, що WAF є не просто одним із засобів захисту, а фундаментальним елементом у багаторівневій архітектурі кіберрезильєнтності.

Для практичної реалізації було обрано open-source WAF open-appsec, що базується на технологіях машинного навчання. Цей вибір був обґрунтований його ключовими перевагами, які відрізняють його від традиційних рішень - здатність до адаптації та проактивного захисту від атак нульового дня, низька кількість хибних спрацювань та висока ефективність. З метою практичної перевірки ефективності обраного брандмауера веб-додатків open-appsec та демонстрації його здатності протистояти сучасним загрозам, було налаштовано простий тестовий стенд. Архітектура стенду імітує реальне робоче середовище, де WAF виконує функцію зворотного проксі, захищаючи веб-додаток від зовнішніх атак

Архітектура стенду імітує реальне робоче середовище, де WAF виконує функцію зворотного проксі, захищаючи веб-додаток від зовнішніх атак.

Основними компонентами тестового середовища є дві віртуальні машини, розгорнуті за допомогою програмного забезпечення VirtualBox.

Тестування ефективності WAF проведено на однакових тестових стендах для забезпечення об'єктивності порівняльного аналізу. Дослідження включало два основні об'єкти: WAF на основі машинного навчання (open-appsec з Advanced AI Model) та WAF на основі сигнатур (ModSecurity з OWASP CRS). Тестування системи проводилося в два етапи: на першому етапі були використанні засоби автоматизації пошуку веб-вразливостей. Сканування показло, що обидва рішення мають високий рівень захистку від необфускованих атак, при цьому, open-appsec за рахунок ML моделі менш чутливий і демонструє нижчий рівень FP. На другому етапі проводилося ручне тестування додатків на вразливості SQL-ін’єкції, міжсайтового скриптингу та проведені обфусковані атаки. Результати дослідження підтверджують ефективність ML WAF open appsec над сигнатурним Modsecurity. Open-appsec зберіг 100% TPR у всіх випадках, підтверджуючи свою здатність до класифікації аномальної поведінки незалежно від маскування. Хоча жорстке налаштування сигнатур в останньому має свої переваги і не потребують годин додаткового навчання. Open-appsec компенсує це прозорістю процесу машинного навчання та демонструє перевагу можливості централізованого управління open-appsec через SaaS підхід

.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ДСТУ ISO/IEC 27001:2023 Інформаційна безпека, кібербезпека та захист конфіденційності. Системи керування інформаційною безпекою. Вимоги (ISO/IEC 27001:2022, IDT) [Електронний ресурс] – Режим доступу: https://online.budstandart.com/ua/catalog/doc-page.html?id_doc=104398 – Дата звернення: 10.08.25

2. Keeping the lights on: Simplifying the cyber resilience framework [Електронний ресурс] – Режим доступу: https://www.acronis.com/en/blog/posts/keeping-the-lights-on-simplifying-the-cyber-resilience-framework/ - Дата звернення: 10.08.25

3. Nist Special Publication 800-160, Volume 2. Developing Cyber-Resilient Systems: A Systems Security Engineering Aproach [Електронний ресурс] – Режим доступу: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-160v2r1.pdf - Дата звернення: 10.08.25

4. X.1051 : Information security, cybersecurity and privacy protection - Information security controls based on ISO/IEC 27002 for telecommunications organizations [Електронний ресурс] – Режим доступу: https://www.itu.int/rec/T-REC-X.1051-202306-I/en - Дата звернення: 10.08.25

5. Report on the cybersecurity and resiliency of the EU communications infrastructures and networks [Електронний ресурс] – Режим доступу: https://digital-strategy.ec.europa.eu/en/library/report-cybersecurity-and-resiliency-eu-communications-infrastructures-and-networks - Дата звернення: 20.08.25

6. Що таке мережевий балансувальник навантаження та як він працює [Електронний ресурс] – Режим доступу: https://blog.colobridge.net/uk/2024/01/network-load-balancer-ua/ - Дата звернення: 10.09.25

7. Що таке CDN (мережа доставки контенту)? [Електронний ресурс] – Режим доступу: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-cdn-set-dostavki-kontenta/ - Дата звернення: 10.09.25

8. NGFW: Що це таке і навіщо він потрібен вашому бізнесу? [Електронний ресурс] – Режим доступу: https://softlist.ua/cases/ngfw - Дата звернення 15.09.25

9. WAF vs. Firewall: Web Application & Network Firewalls [Електронний ресурс] – Режим доступу до ресурсу: https://www.fortinet.com/resources/cyberglossary/waf-vs-firewall - Дата звернення 16.09.25

10. Changes in OWASP Top 10: 2017 vs 2021 [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/digitalfrontiers/changes-in-owasp-top-10-2017-vs-2021-7cea4183288b – Дата звернення 16.09.25

11. What is SQL Injection? Tutorial & Examples | Web Security Academy [Електронний ресурс] – Режим доступу до ресурсу: https://portswigger.net/web-security/sql-injection. - Дата зверення 16.09.25

12. What is cross-site scripting and how to prevent it [Електронний ресурс] – Режим доступу до ресурсу: https://portswigger.net/web-security/cross-site-scripting - Дата зверення 16.09.25

13. What is XXE (XML external entity) injection? Tutorial & Examples [Електронний ресурс] – Режим доступу до ресурсу: https://portswigger.net/web-security/xxe - Дата зверення 16.09.25

14. What Is Session Hijacking: Your Quick Guide to Session Hijacking Attacks [Електронний ресурс] – Режим доступу до ресурсу: https://www.netsparker.com/blog/web-security/session-hijacking/ - Дата зверення 16.09.25

15. Exploiting insecure deserialization vulnerabilities [Електронний ресурс] – Режим доступу до ресурсу: https://portswigger.net/web-security/deserialization/exploiting - Дата зверення 16.09.25

16. Using Components with Known Vulnerabilities [Електронний ресурс] – Режим доступу: https://securityboulevard.com/2020/04/using-components-with-known-vulnerabilities-2/ - Дата звернення 16.09.25

17. What is SSRF? [Електронний ресурс] – Режим доступу: https://www.f5.com/glossary/ssrf - Дата звернення 16.09.25

18. What is a web application firewalll (WAF) ? [Електронний ресурс] – Режим доступу: https://cybersecurity.att.com/blogs/security-essentials/explain-how-a-web-application-firewall-works - Дата звернення 16.09.25

19. What is a Reverse Proxy Server? [Електронний ресурс] – Режим доступу: https://oxylabs.io/blog/reverse-proxy - Дата звернення 16.09.25

20. 4 Open Source Web Application Firewall for Better Security [Електронний ресурс] – Режим доступу до ресурсу: https://geekflare.com/open-source-web-application-firewall/ - Дата звернення 16.09.25

21. open-appsec automatic web application & API security using machine learning [Електронний ресурс] – Режим доступу: https://www.openappsec.io/ - Дата звернення 22.09.25

22. Best WAF Solution in 2024-2025 Real-World Comparison [Електронний ресурс] – Режим доступу: https://www.openappsec.io/post/best-waf-solutions-in-2024-2025-real-world-comparison - Дата звернення 22.09.25

23. Deploy NGINX Proxy Manager with open-appsec managed from central WebUI (SaaS) [Електронний ресурс] – Режим доступу: https://docs.openappsec.io/integrations/nginx-proxy-manager/deploy-nginx-proxy-manager-with-open-appsec-managed-from-central-webui-saas - Дата звернення 30.09.25

ДОДАТОК А - ПОСТЕР

Міністерство освіти і науки України

Національний університет біоресурсів та природокористування України

Розробка засобів забезпечення резильєнтності мережі на прикладному рівні

Виконавець: Лукашенко Дмитро Юрійович, комп'ютерні системи і мережі, КСІМ-24006м
Науковий керівник: Коваленко Олексій Єпіфанович, доктор технічних наук, професор

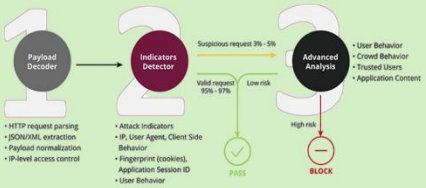
Анотація роботи

Мета – дослідити підходи до забезпечення резильєнтності інформаційних систем на прикладному рівні моделі OSI, проаналізувати еволюцію актуальних загроз веб-додатків.
Об'єкт – резильєнтність інформаційних систем і динаміка змін актуальності загроз веб-додатків у контексті еволюції OWASP Top 10.
Предмет – методи, технології та засоби забезпечення резильєнтності на прикладному рівні, зокрема застосування брандмауера веб-додатків (WAF) для виявлення та блокування вразливостей

Характеристика роботи

У роботі проведено дослідження підходів до забезпечення резильєнтності інформаційних систем на прикладному рівні моделі OSI. Розглянуто еволюцію актуальних загроз веб-додатків на основі порівняльного аналізу OWASP Top 10. Обґрунтовано вибір брандмауера веб-додатків (WAF) як ключового засобу підвищення стійкості веб-сервісів до кіберзагроз. Здійснено практичне тестування open-source WAF рішення open-appsec та виконано порівняльний аналіз з іншим WAF-рішенням Modsecurity

Принцип роботи WAF open-appsec



Актуальні загрози веб-додатків

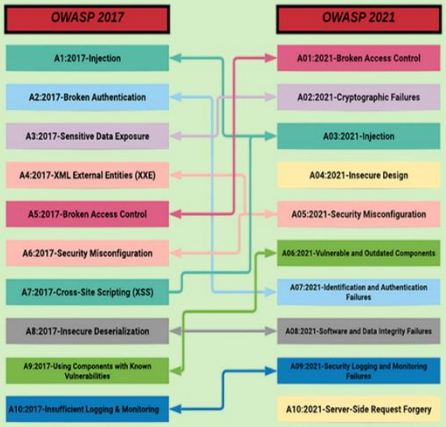
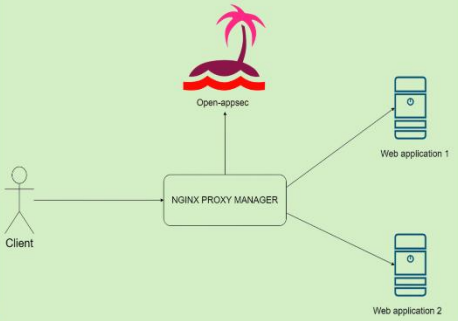


Схема розгортання WAF



Висновки

Дослідження стану кіберзагроз підтвердило необхідність використання адаптивних механізмів захисту для веб-додатків. Проведений аналіз показав, що WAF на базі машинного навчання (open-appsec) забезпечує вищу ефективність та адаптивність проти атак, мінімізуючи хибні спрацювання, у порівнянні зі статичними сигнатурами ModSecurity. Хоча open-appsec вимагає додаткового навчання під кожен сервіс, що є його незначним недоліком, його інтеграція через модель SaaS створює стійке, легко кероване та масштабоване захисне середовище. Таким чином, ML-WAF є кращим вибором для захисту сучасних динамічних веб-додатків.

ДОДАТОК Б – РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ДВОХ WAF

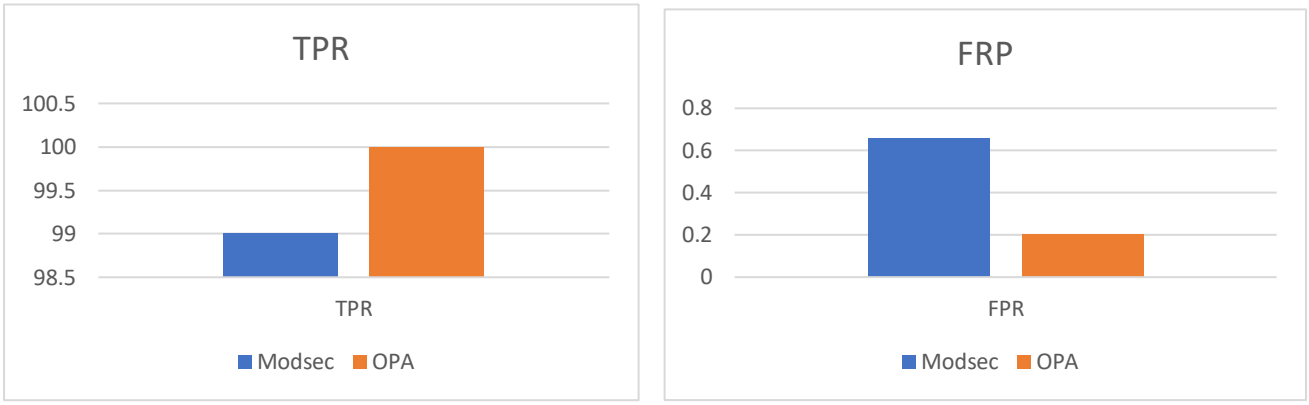


Рисунок А1 – Показники двох WAF при проведенні сканування

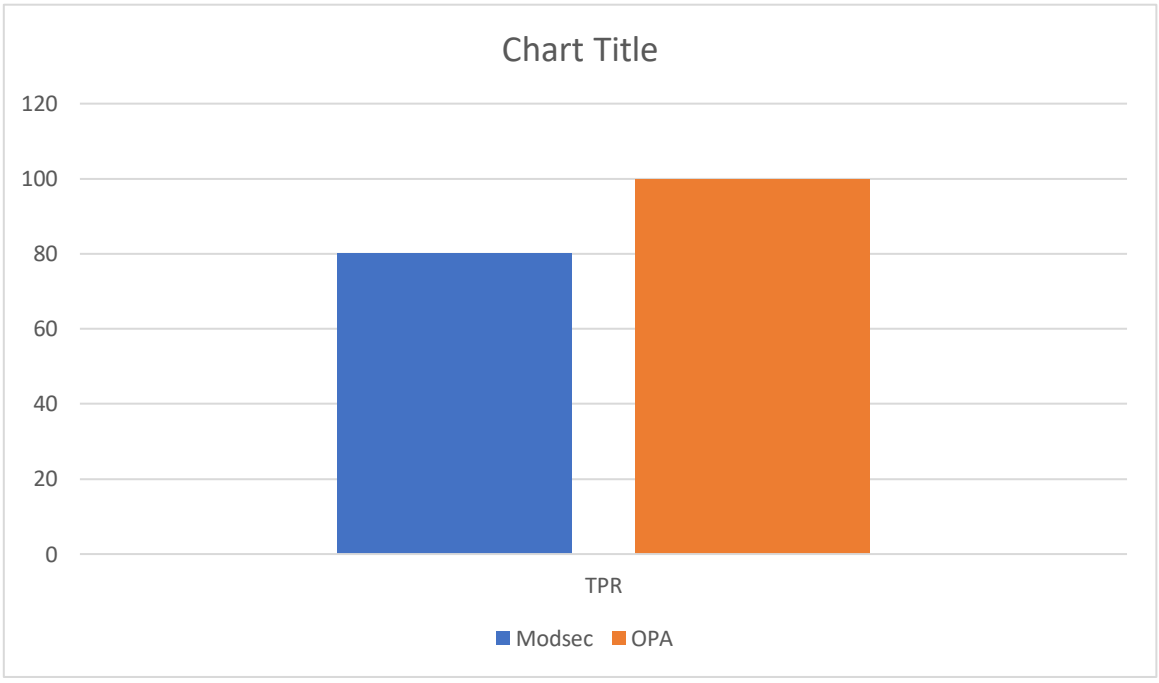


Рисунок А2 – Показники двох WAF при проведенні обфускованих атак