МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ АЕРОНАВІГАЦІЇ,
ЕЛЕКТРОНІКИ ТА ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА ТЕЛЕКОМУНІКАЦІЙНИХ ТА РАДІОЕЛЕКТРОННИХ СИСТЕМ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

_____ Роман ОДАРЧЕНКО
“_____”_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВР

Тема: «Система захисту хмарних серверів Amazon Web Services»

Виконавець: _____ Анна СТРАХ
(підпис)
Керівник: _____ Денис БАХТІЯРОВ
(підпис)

Нормоконтролер: _____ Денис БАХТІЯРОВ
(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій
Кафедра телекомунікаційних та радіоелектронних систем
Спеціальність 172 «Телекомунікації та радіотехніка»
Освітньо-професійна програма «Телекомунікаційні системи та мережі»

ЗАТВЕРДЖУЮ
Завідувач кафедри

Роман ОДАРЧЕНКО
“_____”_____2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Страх Анни Сергіївни
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема кваліфікаційної роботи: «Система захисту хмарних серверів Amazon Web Services»

затверджена наказом ректора від «29» березня 2023 р. № 421/ст

2. Термін виконання роботи: з 22.05.2023 р. по 25.06.2023 р.

3. Вихідні дані до роботи: загальна інформація, статистичні, та фінансово-економічні показники діяльності хмарних серверів Amazon Web Services»

4. Зміст пояснювальної записки: опис актуальності проблеми захисту хмарних серверів; огляд існуючих методів та технологій захисту в хмарних сервісах; розробку пропозицій щодо покращення системи захисту.

5. Перелік обов'язкового графічного (ілюстративного) матеріалу: таблиці, діаграми, графіки та схеми, що ілюструють систем захисту хмарних серверів.

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Розробити деталізований зміст розділів кваліфікаційної роботи	22.05.2023-24.05.2023	Виконано
2	Вступ	25.05.2023	Виконано
3	Аналіз предметної області	26.05.2023-29.05.2023	Виконано
4	Аналіз та вибір сервера	30.05.2023-07.06.2023	Виконано
5	Налаштування та забезпечення безпеки серверних компонентів	08.06.2023-14.06.2023	Виконано
6	Усунення недоліків та захист кваліфікаційної роботи	15.06.2023-25.06.2023	Виконано

7. Дата видачі завдання: “19" травня 2023 р.

Керівник кваліфікаційної роботи _____ Денис БАХТІЯРОВ
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Анна СТРАХ
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Система захисту хмарних серверів Amazon Web Services» включає в себе 58 сторінок, 69 рисунків, 10 використаних джерел.

КЛЮЧОВІ СЛОВА: ХМАРНІ СЕРВЕРИ, AMAZON WEB SERVICES (AWS), ЗАХИСТ ДАНИХ, КОНФІДЕНЦІЙНІСТЬ, БЕЗПЕКА МЕРЕЖІ, АУТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, КОНТРОЛЬ ДОСТУПУ, МОНІТОРИНГ.

Об'єктом дослідження система захисту хмарних серверів Amazon Web Services

Предметом дослідження є діагностика ефективності функціонування системи захисту хмарних серверів Amazon Web Services

Мета дипломної роботи є дослідження та аналіз існуючих методів та технологій захисту в середовищі AWS, розробка та впровадження нових рішень для підвищення безпеки хмарних серверів, а також оцінка ефективності інтегрованої системи захисту з використанням AWS.

У *аналітичні частині* проведенно аналіз хмарних обчислень NIST.

У *проектній частині* було проведено розгортання і налаштування інстансу Amazon EC2, а також налаштовано і забезпечено безпеку WEB-сервера Nginx і поштового сервера.

Матеріали кваліфікаційної роботи можуть бути використані для подальшого дослідження та розробки систем захисту хмарних серверів.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

AWS - Amazon Web Services.

EC2 - Elastic Compute Cloud.

Nginx - "engine x", веб-сервер і проксі-сервер.

WEB-сервер - сервер, що надає веб-сторінки та інші ресурси за запитами клієнтів.

Поштовий сервер - сервер, який обробляє, надсилає та отримує електронні листи.

API - Application Programming Interface (інтерфейс програмування додатків).

SSL - Secure Sockets Layer (протокол безпеки мережевих з'єднань).

IDS - Intrusion Detection System (система виявлення вторгнень).

IPS - Intrusion Prevention System (система запобігання вторгненням).

VPN - Virtual Private Network (віртуальна приватна мережа).

XSS - Cross-Site Scripting (міжсайтовий скриптинг).

CSRF - Cross-Site Request Forgery (міжсайтова поділка запиту).

WAF - Web Application Firewall (веб-програмний брандмауер).

ВСТУП

Тема "Система захисту хмарних серверів Amazon Web Services" має велику актуальність у сучасному світі інформаційних технологій. Зростання використання хмарних сервісів, які надають інфраструктуру та зберігання даних на віддалених серверах, зумовлене їх високою доступністю, масштабованістю та ефективністю.

Проте, залежність компаній від хмарних сервісів також вносить нові виклики у сферу безпеки. Захист хмарних серверів стає критичним завданням, оскільки зберігаються конфіденційні дані клієнтів, бізнес-інформація та інші цінні активи.

Загрози безпеці, такі як хакерські атаки, витоки даних, віруси та зловмисне програмне забезпечення, постійно зростають і вдосконалюються. Тому необхідність розробки та впровадження ефективних систем захисту хмарних серверів стає невідкладною.

Крім того, широке використання хмарних сервісів залежить від довіри клієнтів щодо безпеки їх даних. Розробка надійних систем захисту впливає на підвищення рівня довіри та стимулює подальший розвиток хмарних обчислень.

Таким чином, дана тема має велику актуальність з урахуванням швидкого розвитку хмарних технологій, зростання загроз безпеці та потреби в надійному захисті даних у хмарних середовищах.

Метою дипломної роботи є розробка та реалізація системи захисту хмарних серверів Amazon Web Services для забезпечення безпеки та захисту конфіденційності даних.

В досягненні встановленої мети необхідно вирішити наступні наукові і практичні завдання:

- аналізувати індетифікувати загрози та уразливості хмарних серверів Amazon Web Services.

- розробити стратегію захисту, що включатиме необхідні заходи для запобігання вторгнень та забезпечення конфіденційності даних.

- впровадити розроблену систему захисту на хмарних серверах Amazon Web Services.

- провести тестування та оцінку ефективності реалізованої системи захисту.

- запропонувати рекомендації щодо подальшого удосконалення та розвитку системи захисту хмарних серверів.

Ці завдання дозволять досягти мети дослідження та забезпечити надійний рівень безпеки хмарних серверів Amazon Web Services.

Об'єктом дослідження система захисту хмарних серверів Amazon Web Services

Предметом дослідження є діагностика ефективності функціонування системи захисту хмарних серверів Amazon Web Services

Методи дослідження включають аналіз літературних джерел, експертні оцінки, використання тестових середовищ, проведення експериментів та статистичний аналіз отриманих даних.

Практичне значення отриманих результатів полягає в їх застосуванні для розробки та впровадження ефективної системи захисту хмарних серверів Amazon Web Services. Отримані результати дослідження можуть бути використані організаціями, що використовують хмарні сервіси, для покращення безпеки, забезпечення конфіденційності даних та запобігання вторгнень.

Апробація отриманих результатів. Основні положення роботи доповідалися та обговорювалися на таких конференціях:

- Науково-практична конференція «Проблеми експлуатації та захисту інформаційно-комунікаційних систем», м. Київ, 2023 р.

РОЗДІЛ 1.
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Основні поняття і визначення хмарних обчислень

Хмарні обчислення - це надання обчислювальних потужностей, сховищ для БД, додатків і інших IT-ресурсів повимогам на платформах хмарних сервісів через Інтернет з оплатою за фактом використання.

Платформа хмарних сервісів надає швидкий доступ до гнучких та недорогим IT-ресурсам, необхідним як при запуску додатків для публікації фотографій мільйонів користувачів, так і при управлінні найважливішими бізнес-процесами в компанії. Хмарні обчислення дозволяють позбутися великих попередніх витрат на обладнання та заощадити час, необхідне для трудомісткого управління ім. Замість цього можна розподілити обчислювальні ресурси таких типів та розмірів, які необхідні для реалізації ваших нових яскравих ідей або для управління IT-відділом. Можна практично миттєво отримувати доступ до необхідному кількості ресурсів з оплатою по фактом використання [1].

Хмарні обчислення забезпечують простий доступ до серверів, сховищам, баз даних і великому набору сервісів додатків в Інтернет. Платформи хмарних сервісів, такі як Amazon Web Services, володіють підключеним до мережі обладнанням, необхідним для таких сервісів додатків, і виконують його обслуговування, час як ви розподіляєте і використовуєте необхідні ресурси через інтернет-додаток.

1.2. Визначення хмарних обчислень NIST.

Національний інститут стандартів і технологій (NIST) одним з перших спробувала дати визначення хмарним обчисленням у спеціальній публікації 800-145: «Хмарні обчислення це Модель надання повсюдного і зручного мережевого доступу «по мірі необхідності» до загальному пулу конфігурованих обчислювальних ресурсів (наприклад, мереж, серверів, систем зберігання,

додатків і сервісів), які можуть бути швидко надані і звільнено з мінімальними зусиллями по управлінню та необхідністю взаємодії з провайдером послуг (сервіс-провайдером)».

Хмарна Модель підтримує високу доступність сервісів іописується п'ятьма основними характеристиками (essential характеристики), трьома сервісними моделями/моделями надання послуг (service models) і чотирма моделями розгортання (Deployment models) [2].

Основні Характеристики (Essential Характеристики):

- сервіс самообслуговування за потребою або на вимогу (On- demand self-service) – споживач може самостійно забезпечувати себе обчислювальними можливостями (засобами і ресурсами), такими як:

серверний час та мережеве сховище, при необхідності автоматично, без необхідності взаємодія з кожним постачальником послуг;

- вільний доступ до мережі (Broad network access) – обчислювальні можливості або "запитувані" сервіси доступні по мережі через стандартні механізми, підтримуючі використання гетерогенних платформ тонких і товстих клієнтів (наприклад, мобільних телефонів, ноутбуків та КПК);
- об'єднання ресурсів (Resource pooling) – обчислювальні ресурси провайдера об'єднуються для обслуговування кількох споживачів з використанням множинної моделі (Multi-tenant model - Модель множинної оренди або, точніше, Модель оренди з безліччю орендарів - передбачає доступність споживачам специфічного для кожного з них портфеля та обсягу ресурсів з спільного спектра ресурсів, підтримуваних провайдером) з різними фізичними та віртуальними ресурсами, які динамічно призначаються і перепризначаються в відповідно з вимогами споживачів. Існує відчуття незалежності розташування в тому, що клієнт, як правило, не має ніякого контролю чи знань про точне місцезнаходження наданих ресурсів, але може мати можливість вказати місце розташування на більше високому рівні абстракції (наприклад, країна, штат або центр обробки даних). прикладами таких ресурсів є системи зберігання, обчислювальні можливості, пам'ять, пропускна здатність мережі, віртуальні

машини;

- швидка еластичність (Rapid elasticity) - обчислювальні можливості можуть бути надані швидко та еластично, у ряді випадків - автоматично, для оперативного підвищення масштабованості та швидкого звільнення зменшення масштабів споживання . Для споживача ці ресурси часто надаються як доступні в необмеженому обсязі, так і можуть бути придбано в будь-який момент часу в будь-кому кількості;
- вимірний сервіс (Measured Service) - хмарні системи автоматично контролюють і оптимізують використання ресурсів, використовуючи можливості вимірювання на деякому рівні абстракції, відповідного типу послуги (наприклад, зберігання, обробка, пропускна здатність і активні облікові записи користувачів). Використання ресурсів можна відстежувати, контролювати та звітувати, забезпечуючи прозорість як для постачальника, так і для споживача використовуваної послуги.

Сервісні моделі (Service Models):

- програмне забезпечення як послуга (SaaS) - споживачеві надається можливість використовувати програми провайдера, працюють у хмарній інфраструктурі. Програми доступні з різних клієнтських пристроїв через інтерфейс клієнта, такий як в WEB-браузер(наприклад, електронна пошта через Інтернет) або програмний інтерфейс. Споживач не керує і не контролює базову хмарнуінфраструктуру, включаючи мережу, сервери, операційні системи, сховище або навіть окремі функції реплікації, за можливим винятком обмежених параметрів конфігурації додатки;
- платформа як послуга (PaaS) - споживачеві надається можливість розгортання в хмарний інфраструктурі створених споживачем або придбаних додатків, створених з використанням мов програмування, бібліотек, служб і інструментів, підтримуваних постачальником. Споживач не керує та не контролює базову хмарну інфраструктуру, включаючи мережа, сервери, операційні системи або сховище, але контролює розгорнуті програми і,можливо, параметри конфігурації для середи розміщення додатку;

- інфраструктура як послуга (IaaS) - можливість, що надається споживачеві, полягає у забезпеченні обробки, зберігання, мереж та інших основних обчислювальних ресурсів, де споживач може розгортати та запускати довільне програмне забезпечення, яке може включати в себе операційні системи та програми. Споживач не керує та не контролює базову хмарну інфраструктуру, але контролює операційні системи, сховище та розгорнуті додатки; і, можливо, обмежене управління окремими мережевими компонентами (наприклад, брандмауерами хоста).

Моделі розгортання (Deployment Models):

- приватне хмара (Private cloud) - хмарна інфраструктура надається для виняткового використання однієї організацією, що складається з кількох споживачів (наприклад, бізнес-одиниць). Воно може належати, керуватися та експлуатуватися організацією, третьою стороною або який-небудь їх комбінацією, а також може існувати вприміщенні або за його межами;
- хмара Спільноти (Community cloud) - хмарна інфраструктура надано для виняткового використання певним спільнотою споживачів з організацій, які поділяють загальні принципи (наприклад, місія, вимоги безпеки, політика і міркування відповідності). Воно може належати, керуватися і експлуатуватися однієї або декількома організаціями в спільноті, третьою стороною або який-небудь їх комбінацією, а також може існувати в приміщенні або за його межами;
- публічне хмара (Public cloud) - хмарна інфраструктурапризначений для
- друг з відкритого використання широкою публікою. Воно може належати, керуватися і експлуатуватися комерційної, академічною або державної організацією або який-небудь їх комбінацією. Воно існує на території хмарного провайдера;

гібридне хмара (Hybrid cloud) - хмарна інфраструктура представляє собою сукупність двох або більше окремих хмарних інфраструктур (приватний, громадської або загальнодоступною), які залишаються унікальними об'єктами, але пов'язані стандартизованою або приватний технологією, яка забезпечує портованість даних і додатків (наприклад, розрив хмари для балансування

навантаження між хмари) [3].

1.3. Віртуальні сервери від Amazon.

EC2 – найвідоміший сервіс, що надається Amazon Web Services, репрезентуючий собою віртуальні сервери основі Windows і UNIX. Amazon Elastic Compute Cloud (Amazon EC2) забезпечує масштабовану обчислювальну потужність у хмарі Amazon Web Services (AWS). Amazon EC2 дозволяє збільшувати або зменшувати обчислювальну потужність за кілька хвилин, а чи не годин чи днів. Можна вводити в експлуатацію один серверний інстанс, сотні або навіть тисячі серверних інстансів одночасно. Можна також використовувати Amazon EC2 Auto Scaling, щоб підтримувати доступність групи інстансів EC2 і автоматично масштабувати групу в потрібному напрямку в залежності від потреб, щоб підтримувати максимальну продуктивність і скорочувати затраты[4].

Amazon EC2 розміщується в кількох місцях по всьому світу. Ці розташування складаються з регіонів і зон доступність. Кожен регіон є окремою географічною зоною. Кожен регіон має кілька ізольованих місць, відомих як зони доступність. Кожна зона доступності ізольовані, але зони доступності в регіоні пов'язані через канали з низькою затримкою. Amazon EC2 надає можливість розміщувати ресурси, такі як екземпляри, та дані у кількох місцях. Ресурси не копіюються регіонами, якщо ви не зробите це спеціально. Рисунок 1.1 ілюструє взаємозв'язок між регіонами і зонами доступності:

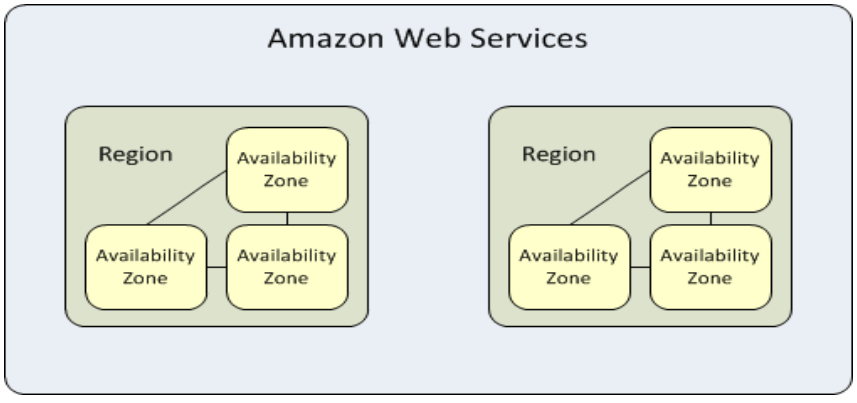


Рис. 1.1. Взаємозв'язок між регіонами і зонами доступності

У Amazon EC2 інстансами називають віртуальні сервери. Інстанси T2 - це інстанси з підвищуваною продуктивністю, які забезпечують базовий рівень продуктивності ЦПУ із можливістю йогопідвищення. Вони можуть підтримувати високу продуктивність ЦПУ до тих пір, Бувай цього вимагає робоча навантаження. Для більшості робітників навантажень спільного призначення безлімітні інстанси T2 забезпечують достатню продуктивність без додаткову плату.

Особливості інстанстів T2:

- високочастотні процесори Intel Xeon;
- постійний базовий рівень продуктивності і ЦПУ з можливістю підвищення продуктивності (регулюється кредитами ЦПУ);
- самий економічний тип інстансів спільного призначення. Доступний нарівні безкоштовного користування;
- збалансоване співвідношення обчислювальних, мережевих ресурсів і ресурсів пам'яті.

Для безкоштовного користування доступний інстанс T2 зі наступними характеристиками:

- 1 віртуальний процесор Intel Xeon Family 2,5 ГГц;
- 1 ГБ оперативною пам'яті;
- 8 Гб постійної пам'яті.

Групи безпеки AWS (security groups) пов'язані з екземплярами EC2 та забезпечують безпеку на рівні протоколу та доступу до порту. Кожна група безпеки, що працює майже так само, як і брандмауер, містить набір правил, які фільтрують трафік, вступник в екземпляр EC2 і виходить із нього. Тут немає жодних правил "Заборонити". Швидше, якщо не існує правила, яке явно дозволяє певний пакет даних, він буде відкинуто. Змінити правила для групи безпеки можна, можливо в будь-яке час; нові правила автоматично застосовуються до всім інстанс, які пов'язані з групою безпеки.

Для безпечного входу до системи Amazon EC2 використовуєкриптографію з

відкритим ключем для шифрування та дешифрування інформації. Криптографія з відкритим ключем використовує відкритий ключ для шифрування фрагменту даних, а потім одержувач використовує закритийключ для дешифрування даних. Відкритий і закритий ключі відомі якпара ключів. Криптографія з відкритим ключем дозволяє безпечно. отримувати доступ до інстанс, використовуючи закритий ключ замість пароля.При створенні інстансу, потрібно вказати пару ключів. Можна, можливо вказати існуючу пару ключів або створити нову пару ключів. У часзавантаження вміст відкритого ключа міститься в екземпляр в записи в ~ /

.ssh / authorized_keys. Щоб Увійти в систему необхідно вказати закритий ключ при підключенні до екземпляру [5].

Amazon Elastic Block Store (Amazon EBS) надає томи сховищна рівні блоків для використання з екземплярами EC2. Тома EBS – це високодоступні та надійні томи зберігання, які можна підключити до будь-кому працюючому екземпляру в тій ж зоні доступність. Тома EBS, підключені до екземпляра EC2, відображаються як томи зберігання, які зберігаються незалежно від терміну служби екземпляра.

На рисунку 1.2 видно, як відбувається підключення до інстансу:

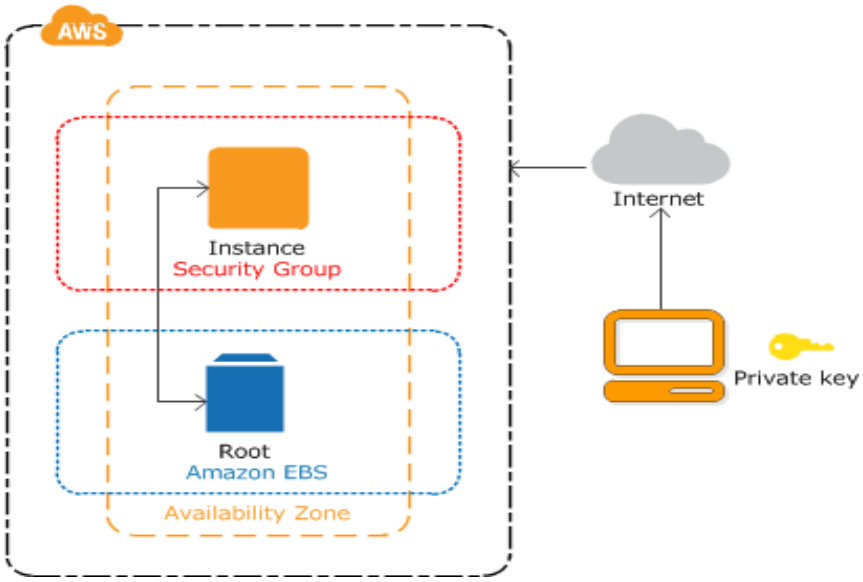


Рис. 1.2. Схема підключення до інстансу EC2

1.4. WEB-сервер і поштовий сервер

WEB-сервер – це комп'ютер, де зберігається WEB-контент. У здебільшого WEB-сервер використовується для розміщення WEB-сайтів, але існують і інші WEB-сервери, такі як ігрові сервери, сховища, FTP і т. буд. Найкращим визначенням може бути те, що WEB-сервер - це сервіс, Котрий відповідає на HTTP-запити на доставку контенту та послуг.

WEB-сервер відповідає на запит клієнта одним з наступних двох способів:

- відправка файлу клієнту, пов'язаному з запитаним URL;
- генерація відповіді шляхом виклику скрипта та зв'язку з базою даних.

на рисунку 1.3 ілюстровано, яким чином відбувається взаємодія WEB-клієнта та WEB-сервера.

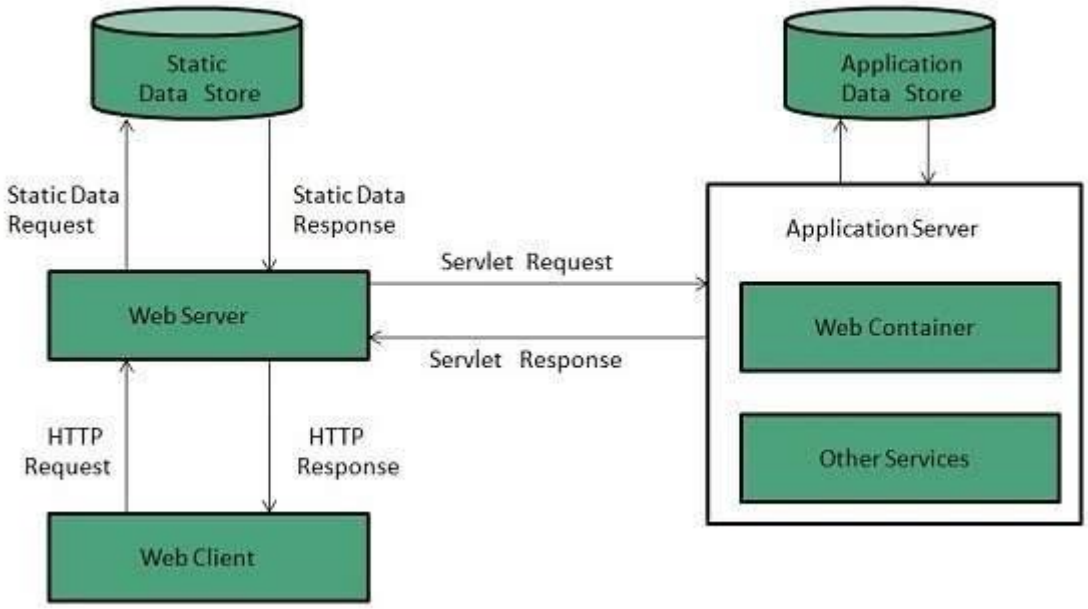


Рис. 1.3. Архітектура взаємодії WEB-клієнта і WEB-сервера

Архітектура WEB-сервера використовує наступних два підходу для обробки запитів:

- паралельний підхід;
- підхід, заснований на окремих процесах.

Паралельний підхід дозволяє WEB-серверу обробляти декілька клієнтських запитів одночасно. Це може бути досягнуто наступнимиметодами:

- мульти-процесний метод;
- багатопотоковий метод;
- гібридний метод.

При використанні мультипроцесного методу батьківський процес ініціює кілька однопоточних дочірніх процесів і розподіляє вхідні запити до цих дочірніх процесів. Кожен із дочірніх процесів відповідає за обробку одного процесу.

Відповідальність за контролем навантаження і прийняття рішення про тому, слід чи знищувати або розгалужувати процеси, лежить на батьківськомупроцесі.

У відмінність від мульти-процесного методу, багатопотоковий створює кілька однопоточних процесів.

Гібридний метод – це комбінація двох вищезазначених підходів. При такому підході створюється кілька процесів, і кожен процес ініціює кілька потоків. Кожен з потоків обробляє одне з'єднання. Використання кількох потоків в одному процесі призводить до меншою навантаження на системні ресурси [6].

Поштовий сервер - це сервер, Котрий обробляє і доставляє електронну пошту по мережі, зазвичай через Інтернет. Поштовий сервер може отримувати електронні листи з клієнтських комп'ютерів та доставляти їх на інші поштові сервери. Поштовий сервер також може доставляти електронної пошти на клієнтські комп'ютери. Клієнтський комп'ютер – це, як правило, комп'ютер, на якому ви читаєте свою електронну пошту, наприклад, комп'ютер вдома чи офісі. Крім того, в цих обставинах як клієнтський комп'ютер можна розглядатиудосконалений мобільний телефон або смартфон.

На рисунку 1.4 ілюстровано, яким чином здійснюється прийом тапередача електронного повідомлення.

Рис. 1.4. «Рух» електронного повідомлення

SMTP - це протокол прикладного рівня. Клієнт, Котрий хоче Відправити пошту, відкриває TCP-з'єднання з SMTP-сервером, а потім надсилає пошту через з'єднання. SMTP-сервер завжди перебуває в режимі прослуховування. Як тільки він прослуховує TCP-з'єднання від будь-якого клієнта процес SMTP ініціює з'єднання через цей порт (25). Після успішного встановлення TCP-з'єднання клієнтський процес відправляє пошту миттєво.

IMAP та POP – два способи доступу до електронної пошти. IMAP - це рекомендований метод, коли вам потрібно перевірити свою електронну пошту з кількох різних пристроїв, таких як телефон, ноутбук і планшет. IMAP дозволяє вам отримати доступ до електронної пошти, де б ви ні перебували з будь-якого пристрою. Коли ви читаєте повідомлення електронного пошти з використанням IMAP, ви фактично не завантажуєте та не зберігаєте його на своєму комп'ютері; натомість ви читаєте це з поштової служби. У результаті ви можете перевірити свою електронну пошту з різних пристроїв в будь-якій точці світу.

IMAP завантажує повідомлення лише при натисканні на нього, а вкладення не завантажуються автоматично. Таким чином, ви зможете перевіряти свої повідомлення набагато швидше, ніж POP.

POP працює, зв'язуючись з вашої службою електронної пошти і завантажуючи Усе ваші нові повідомлення з її. Як тільки вони завантажуються на ПК, вони видаляються із служби електронної пошти. Це означає, що після завантаження електронної пошти до неї можна отримати доступ тільки з того ж комп'ютера. Якщо ви спробуєте отримати доступ до своєю електронної поштою з іншого пристрої, раніше завантажені повідомлення не будуть вам доступні. Надіслана пошта зберігається локально ПК, а не на поштовоюсервері [7].

ВИСНОВКИ ДО РОЗДІЛУ 1

У даної главі я дав визначення поняттю хмарних обчислень, розглянув визначення хмарних обчислень, яке дав Національний інститут стандартів і технологій (NIST). Хмарні обчислення - це Модель, надає доступ до ресурсів по мережі Інтернет, яка підтримує високу доступність сервісів і описується п'ятьма основними характеристиками (essential характеристики), трьома сервісними моделями/моделями надання послуг (service models) і чотирма моделями розгортання (deployment models).

Описав сервіс Amazon EC2 і основні компоненти, такі як, інстанси, група безпеки AWS, регіони та зони доступності, Amazon Elastic Block Store. Детально описав процес підключення до інстансам EC2.

Також розглянув поняття WEB та поштових серверів, описав їхню роботу. Перерахував підходи WEB-сервера для обробки запитів – паралельний підхід та підхід заснований на окремих процесах. Паралельний підхід дозволяє WEB-серверу обробляти кілька клієнтських запитів одночасно, що може бути досягнуто з допомогою наступних методів:

- мульти-процесорний метод;
- багатопотоковий метод;
- гібридний метод.

Основна завдання протоколу SMTP полягає в тому, щоб забезпечувати передачу електронних повідомлень (Пошту). Для роботи через протокол SMTP клієнт створює TCP з'єднання із сервером через порт 25. Потім клієнт і SMTP сервер обмінюються інформацією доки з'єднання не буде закрито або перервано.

РОЗДІЛ 2.
АНАЛІЗ І ВИБІР СЕРВЕРА

У цьому розділі я проаналізую ринок WEB-серверів, їх можливостіі виберу WEB-сервер; опишу основні заходи захисту серверів.

2.1. Вибір WEB-сервера

Був час, коли WEB-сервер Apache обслуговував близько 60 відсотків, а іноді навіть більше, WEB-сайтів у світі. Цей відсоток з того часу впав нижче 30 і досі падає досить значним темпом. Тим часом, WEB-сервер Microsoft IIS займає досить стійку, дещо зростаючу частку ринку, досягнувши сьогодні 26 відсотків. Друге місце з Microsoft IIS ділить конкурент по імені NGINX (вимовляється як "engine-x"), Котрий в справжнє час також обслуговує близько 26 відсотків усіх сайтів, і цей показник постійно зростає приблизно на один відсоток щороку.

На рисунку 2.1 представлений графік, що відображає частку ринку WEB-серверів на ринку.

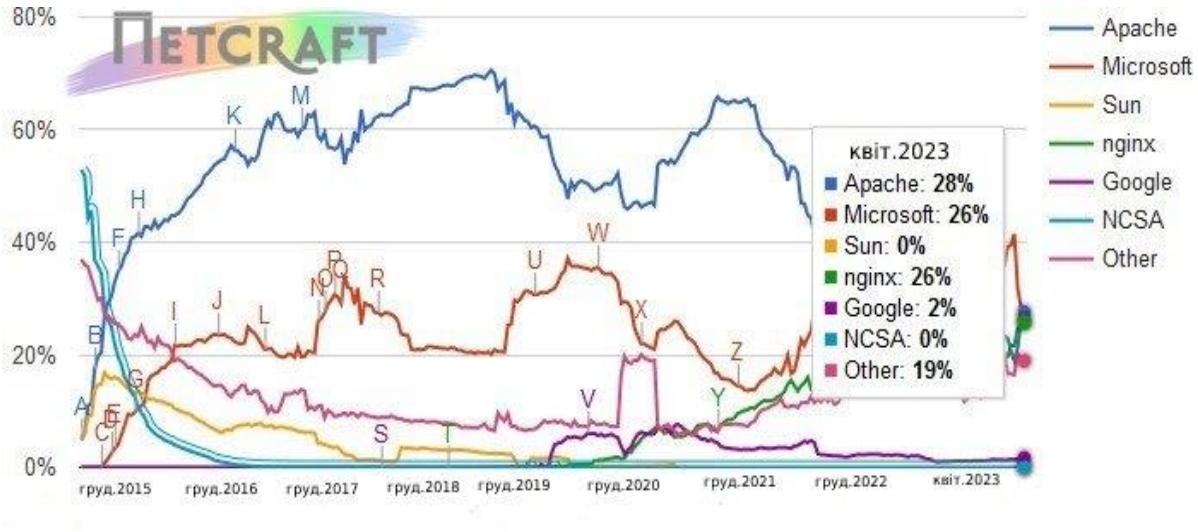


Рис. 2.1. Частка WEB-серверів на ринку

Apache. Apache - це компонент WEB-сервера популярного стека LAMP (Linux, Apache, MySQL, PHP). Незважаючи на те, що в наші дні існує безліч інших компонентів WEB-стеку (наприклад, NodeJS, JS-середовища для багатофункціональних клієнтів, різні хмарні сервіси і т. Д.), LAMP як і раніше залишається дуже популярним.

WEB-сервер Apache має багатий набір функцій, які можна, можливо увімкнути, встановивши один із приблизно 60 офіційних модулів або один з безлічі інших неофіційних модулів, які також існують.

За ці роки Apache розробив кілька методів обробки WEB- запитів для підвищення його ефективності (головним чином, використання ОЗП та затримки). У світі, де сайти мають обробляти все більше одночасних WEB-запитів, і де розміри обслуговуваних сторінок стають набагато більше, необхідні були нові методології.

Методологія обробки запитів Apache може бути налаштована одним із трьох способів. Нижче я наведу три основні модулі мультипроцесингу (MPM):

- Модель процесу - це оригінальний метод "pre-fork"; він погано масштабується зі багатьма одночасними з'єднаннями, так як він споживає багато оперативною пам'яті і може навіть відмовитися від з'єднань при високих навантаженнях.
- Модель воркера - цей метод створює єдиний процес управління, що відповідає за запуск дочірніх процесів. Кожен дочірній процес потім створює фіксоване кількість потоків, і навіть потік слухача. Потік слухача прослуховує з'єднання і передає їх потоку для обробки, коли вони надходять. Хоча ця модель масштабується набагато краще, ніж метод «pre-fork», вона все ж таки може зіткнутися з проблемами масштабування для сайтів із дуже великим трафіком через єдине вузьке місця процесу керування.
- модель події - це схоже на робочу модель, але вона створює один потік слухача, який прослуховує з'єднання та передає їх у робітник потік для обробки. Цей MPM набагато ефективніше обробляє тривалі з'єднання в одному потоці (обробка KeepAlive). Починаючи з Apache 2.4 модель подій вважалася стабільною

і тепер також є за замовчуванням, якщо операційна система може підтримувати її.

Nginx. NGINX був створений у відповідь на завдання C10K з обробки як щонайменше 10 000 одночасних клієнтських підключень на одному сервері. NGINX використовує асинхронну архітектуру, керовану подіями, обробки такої величезної кількості сполук. Ця архітектура робить обробку високих і вагальних навантажень набагато більш передбачуваної з крапки зору використання ОЗУ, використання ЦП та затримки.

Основна відмінність між NGINX та Apache з погляду моделей подій полягає в тому, що NGINX не встановлює додаткові робітники процеси на з'єднання. У більшості випадків рекомендована конфігурація NGINX - це запуск одного робітника процесу на процесор,що максимізує ефективність обладнання.

NGINX також має багатий набір функцій і може виконувати різні ролі сервера:

- зворотний проксі-сервер для протоколів HTTP, HTTPS, SMTP, POP3і IMAP;
- балансувальник навантаження і HTTP-кеш;
- інтерфейсний проксі для Apache і інших WEB-серверів, поєднує гнучкість Apache з гарною продуктивністю статичного вмісту NGINX.

NGINX підтримує обробники FastCGI та SCGI для обслуговування сценаріїв динамічного вмісту, таких як PHP і Python. Він використовує стек LEMP: різновид LAMP з використанням фонетичного написання NGINX (Linux, "En-juhn-ex", MySQL, PHP).

Порівняння Apache і Nginx. Apache на кожен запит від клієнта створює окремий процес (або потік залежить від обраного mpm модуля). Виглядає це в такий спосіб - клієнт надсилає запит, WEB-сервер створює окремий процес на цей запит, відповідає клієнту і блокує процес доти, доки клієнт не закриє з'єднання. Процес у будь-якій ОС вимагає пам'яті та ресурсів, а коли процесів ставати непристойно багато, обробка з'єднань непристойно сповільнюється, пам'ять закінчується, CPU зростає. Для дрібних проектів така реалізація архітектури обробки з'єднань не додасть головного болю, але для високонавантажених

проектів доведеться ставити дуже потужне залізо або Шукати альтернативні варіанти.

Nginx складається з master-процесу і кількох дочірніх процесів. Майстер процес зазвичай один - він створює дочірні процеси (воркери,завантажувач кеша і кеш менеджер), зчитує конфігурацію і відкриває порти. Воркерів зазвичай кілька, розробники nginx радять кількість воркерів визначати рівним числу ядер машини. Ці дочірні процеси обслуговуватиму всі з'єднання з клієнтами в неблокуючій манері. У nginx використовується нескінченний цикл, Котрий біжить по всім з'єднаннямита відповідає на запити клієнтів. Коли з'єднання закривається, воно видаляється з event loop. Це Рішення ідеально підходить для проектів, які обслуговуючих понад 10000 з'єднань одночасно. При цьому завантаження CPU і використання пам'яті зазвичай рівномірні, без видимих піків.

При віддачі статичного контенту NGINX приблизно 2,5 разу швидше, ніж Apache, ґрунтуючись на результатах тестів продуктивності, виконуючих до 1000 одночасних підключень. Інший тест з 512 одночасними підключеннями показав, що NGINX працює приблизно в двічі швидше і споживає трохи менше пам'яті (4%). Споживання пам'яті серверів можна, можливо побачити рисунку 2.2.

Рис. 2.2. Споживання пам'яті WEB-серверів Apache і Ngnix

Під час тестування віддачі динамічного контенту, Apache та Nginx показують приблизно однакові результати. Причиною цього є те, що майже Усе запити обробляються в середовищі виконання PHP, а не вОсновний частини WEB-сервер. З крапки зору PHP (і, мабуть, інших мов), продуктивність сервера динамічних сторінок практично дорівнює при правильною на будівництві модуля Apache (PHP-FPM + FastCGI).

Обидва проекти мають відмінну репутацію в галузі безпеки для своєї бази

коду на C. Однак кодова база NGINX значно менша на кілька порядків, що, безумовно, є великим плюсом здалекоглядної точки зору безпеки. Доступні звіти про вразливість для Apache 2.2 та 2.4. NGINX також має список останніх рекомендацій щодо безпеки. Також в на відміну від Apache, модулі Nginx не можуть бути динамічно завантажені на льоту та вимагають складання. Це складніше, але вважається безпечніше [8].

2.2. Заходи захисту серверів

Дослідження показують, що щодня зламується 37 000 WEB- сайтів, що відповідає 13,5 мільйонам WEB-сайтів в рік. Зловмисники використовують широкий спектр методів поширення своїх шкідливих програм від використання вразливостей програмного забезпечення до зараження рекламних мереж. Звичайний міжмережевий екран недостатньо для безпеки WEB-сервер. Тому необхідно розгорнути серію захистів, одну за іншою .

Насамперед необхідно подбати про безпечне підключення до WEB-серверу. Використання SSH не вирішує проблему нелегітимного доступу до сервера. Адже, якщо, дозволено аутентифікацію на основі пароля, зловмисники можуть дістатися до даних сервера. Саме по цією Тому рекомендується використовувати ssh-ключі для авторизації. В основі технології - пара криптографічних ключів, які використовують для перевірки автентичності як альтернативу аутентифікації за допомогою пароля. Система входу використовує закритий і відкритий ключі, які створюють до автентифікації. Закритий ключ зберігається в таємниці надійним користувачем, у той час як відкритий ключ може лунати з будь-якого сервера SHH, до якого потрібно підключитися [7].

Позбутися від переважного більшості мережевих атак можна, можливо відключивши мережеві служби, закривши всі порти, за винятком, тільки тих, які справді потрібні - це самий фундаментальний принцип в мережевий безпеки. Заборонити Усе і дозвольте тільки те, що потрібно [9].

Брандмауер WEB-додатків - невід'ємна деталь механізму захисту WEB-серверів. Він фільтрує, відстежує і блокує HTTP- трафік в WEB-додаток і з нього.

WAF відрізняється від звичайногобрандмауера тим, що WAF може фільтрувати вміст певнихWEB-додатків, у той час як звичайні брандмауери служать як шлюзу безпеки між серверами. Перевіряючи HTTP-трафік, він може запобігти атаки, пов'язані з вразливістю безпеки WEB- додатків, такими як використання SQL-ін'єкцій, міжсайтовий скриптинг (XSS), використання файлів і неправильна конфігурація безпеки.

Усі WEB-сервери оптимізовані для забезпечення функціональності у конфігурації за замовчуванням. Це уявляє зловмиснику велику область атаки. Отже, забезпечення безпеки конфігурації сервера є першим кроком у безпеці WEB-служби. У конфігурації по замовчуванням більшість WEB-серверів відображають велике кількість інформації про версію програмного забезпечення, список файлів у каталозі, списку модулів WEB-сервера і т. д. Якщо ці налаштування явно не вимкнені, вони надають велику кількість інформації для атакуючого. Виходячи із сукупності всіх факторів, необхідно запобігти відображенню цієї інформації. Також потрібно відредагувати конфігураційний файл для захисту від поширених атак, як Slowloris, Syn flood.

Рекомендується встановлення WEB-сервера тільки з необхідними модулями та функціями, щоб зловмисники не використовували вразливості невикористовуваних модулів; видалення файлів WEB-сторінки по замовчуванням, сценарії cgi за замовчуванням та каталог керівництва WEB-сервера, оскільки вони містять інформацію про WEB-сервер.

Слід обмежити доступні методи HTTP. Переважнабільшість WEB-додатків використовують запит «GET», запит «POST» або "HEAD" для відображення WEB-сторінок. За замовчуванням WEB-сервери дозволяють використовувати інші типи запитів, такі як "DELETE",

"TRACE", "TRACK" і т.д., які можуть використовуватися зловмисниками для одержання системної інформації. З цієї причини їх можна вимкнути.

Якщо сайт працює з приватними даними користувачів, такими як номери кредитних карток, паролі від інших сервісів, або ж надає доступ до інший важливою інформації, яку може отримати зловмисник, необхідно подбай про

шифрування. Рекомендується використовувати SSL/TLS для шифрування даних, щоб захиститися від атак типу man-in-the-middle [10].

Використання CDN, а саме прикордонних серверів, для приховування справжнього IP-адреса сервера, може бути гарною мірою захисту. УВ даний час багато CDN надають послуги із захисту від DDoS-атак. абсолютно безкоштовно, що так само буде незайвим для безпеки WEB- сервера.

Використання мережевих сканерів безпеки для виявлення небезпечних конфігурацій, відсутності необхідних оновлень та наявності вразливостей серверів може стати гарною точкою в захист-WEB сервера.

Для захисту поштового сервера, насамперед необхідно додати та налаштувати SPF ,DKIM записи та механізм DMARC, які можуть допомогти запобігти підробці електронної пошти та гарантувати, що легітимні електронні листи доставлятимуться до папки «Вхідні» одержувача, а нев папку «Спам».

Як і у випадку з WEB-сервером, рекомендується використання шифрування для захисту повідомлень електронної пошти - навіть якщо вони будуть перехоплені, їх вміст буде неможливо прочитати.

При боротьбі зі спамом можна, можливо скористатися наступними правилами:

- відхилити електронну пошту, якщо у клієнта SMTP ні записи PTR;
- увімкнути обмеження HELO / EHLO Hostname в Postfix;
- відхилити електронну пошту, якщо ім'я хоста клієнта SMTP немає дійсного запису A;
- відхилити електронне лист, якщо в MAIL FROM домену ні niMx записи, ні A записи;
- запуск Greylisting в Postfix;
- використання загальнодоступних антиспамових чорних списків
- блокувати спам в електронної поштою, перевіряючи Заголовок і текст повідомлення.

ВИСНОВКИ ДО РОЗДІЛУ 2

У цьому розділі я розглянув і порівняв WEB-сервера Apache та Nginx. Я вибрав Nginx як WEB-сервер тому, що на сьогоднішній день він набирає популярність і відтісняє Apache з лідируючої позиції в рейтингу серверів, що використовуються по всьому світу, з точки зору продуктивності, якщо мова йде про обробці статичного контенту, оптимальності використання ресурсів Nginx так ж перевершує Apache. Швидкість обробки динамічного контенту у цих WEB-серверів однакова в тому у випадку, коли Apache налаштована правильно, тобто це зайві виправлення конфігураційного файлу, яких при на будівництві Nginx можна, можливо уникнути.

Я не порівнював поштові сервери, тому що порівняв Apache та Nginx, тій причини, що Усе доступні безкоштовні поштові сервери мають ідентичну архітектуру та функціонал. З цієї причини мій вибір упав у бік iRedMail, який автоматично встановлює та налаштовує все необхідні компоненти поштового сервера.

Також у цьому розділі я описав заходи захисту WEB та поштових серверів. Захист WEB серверів, в основному, полягає у запобіганні витоку інформації, яка може бути «точкою входу» для зловмисника. Захист поштових серверів, в здебільшого, полягає в фільтрації спаму, забезпечення успішної та безпечної доставки легітимних вихідних листів, запобігання використання доменного імені зловмисниками для відправки спаму, використання антивірусів для сканування вхідна пошти на утримання шкідливих вкладених файлі.

РОЗДІЛ 3.
ПРАКТИЧНА ЧАСТИНА

У цьому розділі буде розглянуто процес розгортання та налаштування віртуальних серверів Amazon EC2, налаштування та забезпечення безпеки WEB та поштових серверів.

3.1. Розгортання і налаштування інстансу Amazon EC2

Першим справою при створенні інстансу Amazon EC2 потрібно вибрати операційну систему і тип інстансу. Я вибираю Ubuntu server 18,04 в якості ОС, і t2 micro в якості інстансу (рисунок 3.1):

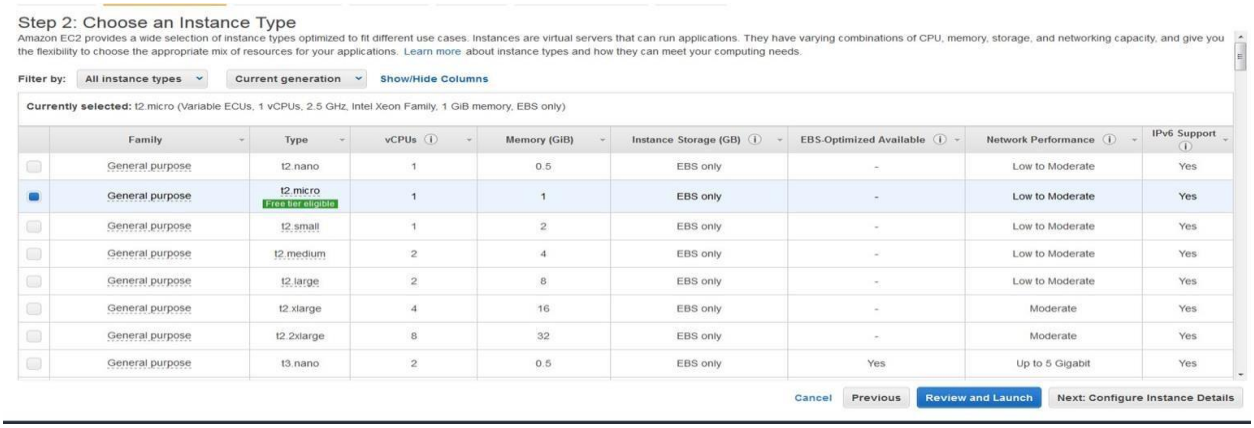


Рис.3.1.Доступні типи інстансів Amazon EC2

Далі необхідно налаштувати групи безпеки AWS таким чином, щоб вхідний трафік підходив для WEB та поштових серверів (рисунок 3.2).

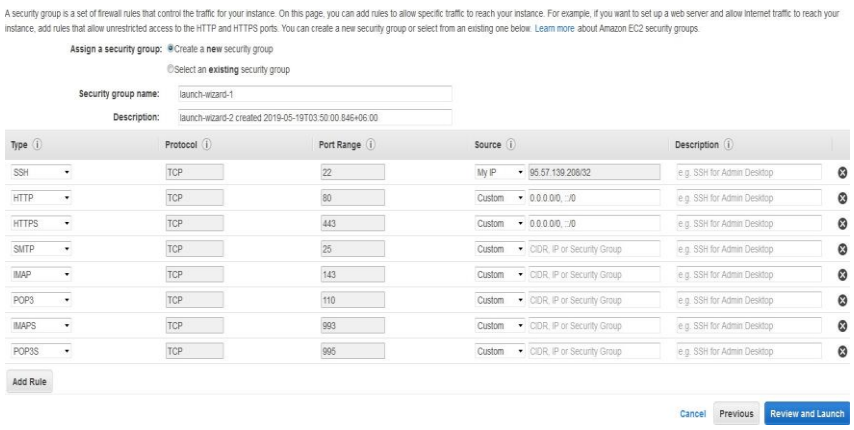


Рис.3.2.Налаштування групи безпеки AWS

На наступному кроці необхідно створити пару ключів (відкритого і закритого), завантажити собі закритий ключ, Котрий буде постійно використовуватися для підключення до інстансу по SSH (Рисунок 3.3).

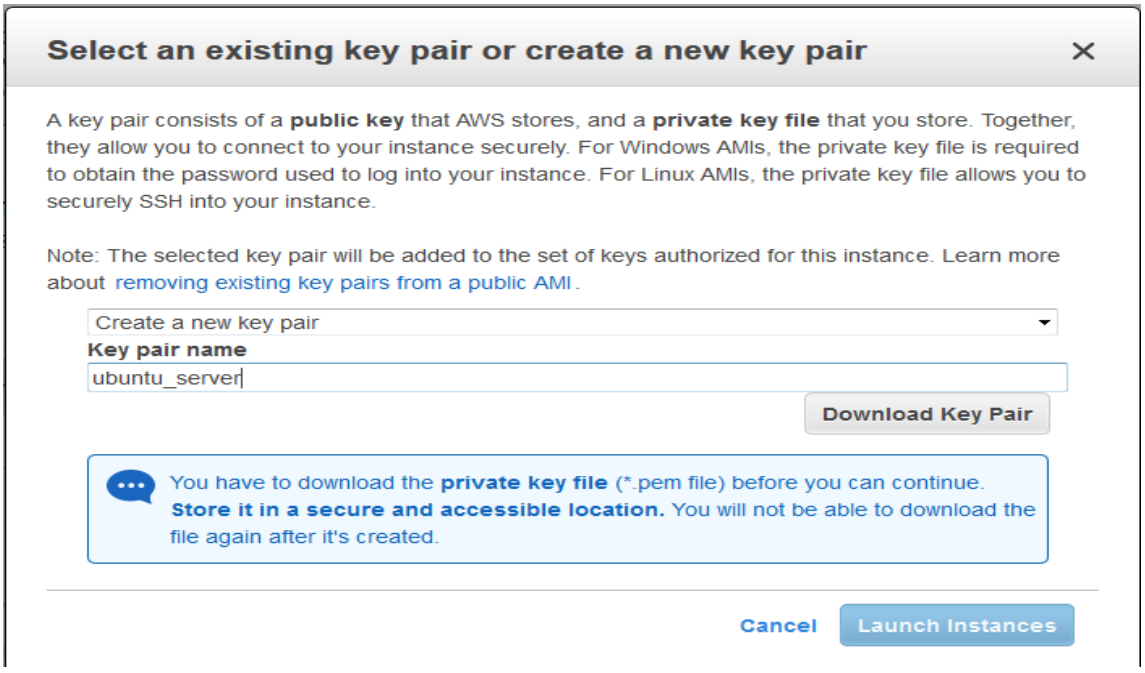


Рис. 3.3. створення пари ключів

3.2 Налаштування і забезпечення безпеки WEB-сервера Ngnix

Безпека WEB-сервера починається з етапу встановлення Nginx, так як ми можемо зібрати Ngnix з вихідного коду, виключивши непотрібні та потенційно

небезпечні модулі, і включивши корисні модулі, як WAF(рисунок) 3.4).

Рис. 3.4. Складання Nginx з вихідного коду

Після збирання встановлюємо правила блокування для захисного екрану рівня програми naxsi (рисунок 3.5), і включаємо ці правила в конфігураційному файлі nginx.conf (рисунок 3.6).

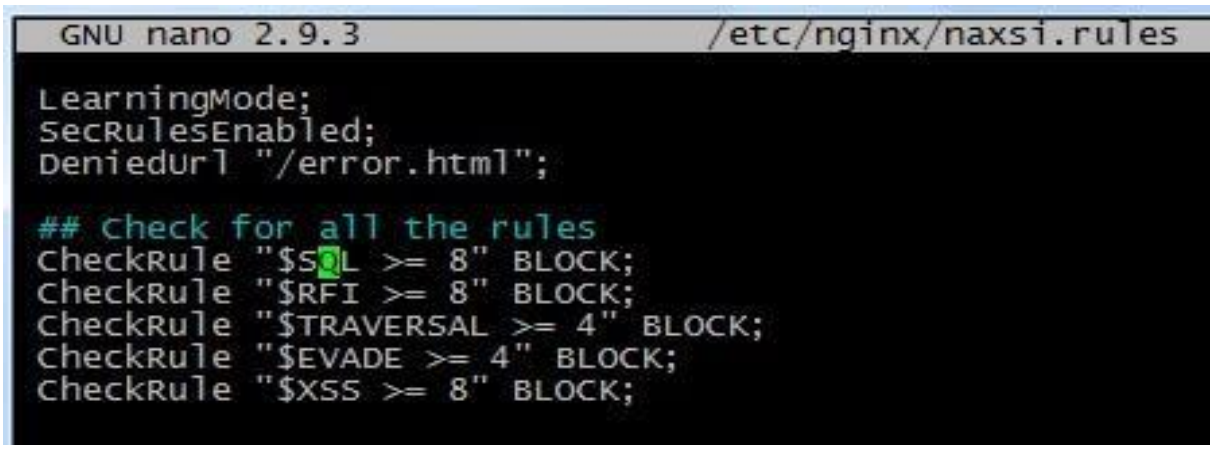


Рис. 3.5. перелік правил блокування naxsi

Рис. 3.6. Включаємо можливості naxsi

Відправивши серверу «підозрілий» запит, тестуємо працездатність naxsi (рис. 3.7). Логи naxsi можна побачити на рисунку 3.8.

Рис. 3.7.результат роботи naxsi

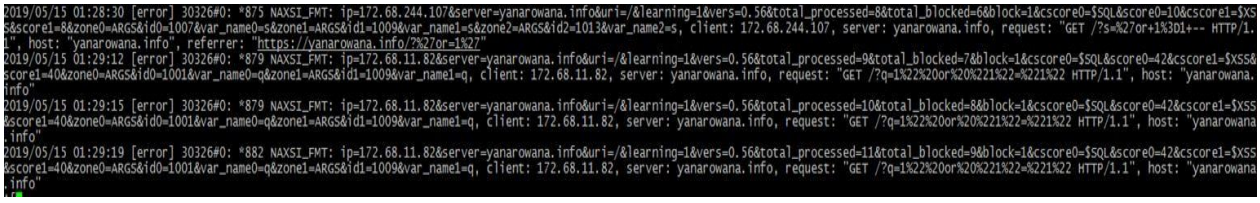


Рис. 3.8. Логи naxsi

Для використання безпечного підключення нам потрібен SSL/TLS сертифікат. Компанія Cloudflare дозволяє створити безкоштовний сертифікат TLS підписаний Cloudflare, для установки на сервер. Всі що від нас потрібно це встановлення відкритого та закритого ключа на сервері (рисунок 3.9, рисунок

3.10), увімкнути опцію, що дозволяє використовувати SSL і вказати шлях до ключам в файл конфігурації WEB-сервера (рисунок 3.11).

Рис. 3.9. Файл відкритого ключа

Рис. 3.10. Файл закритого ключа

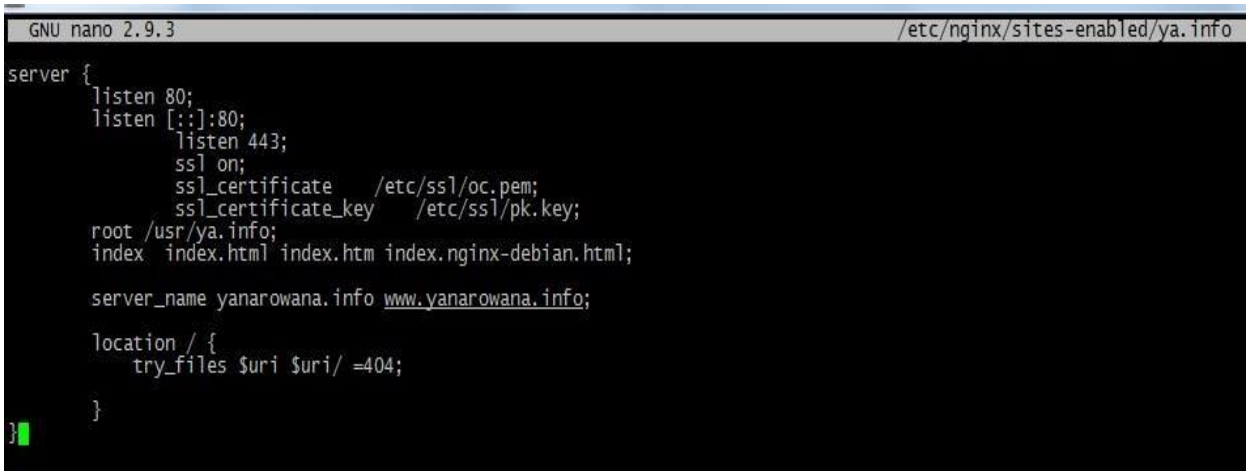


Рис. 3.11.Налаштування конфігураційного файлу для використання SSL

При відвідуванні сайту можна, можливо переконатися, що підключенню здійснюється за допомогою протоколу HTTPS. Параметри сертифіката представлені на рисунку 3.12.

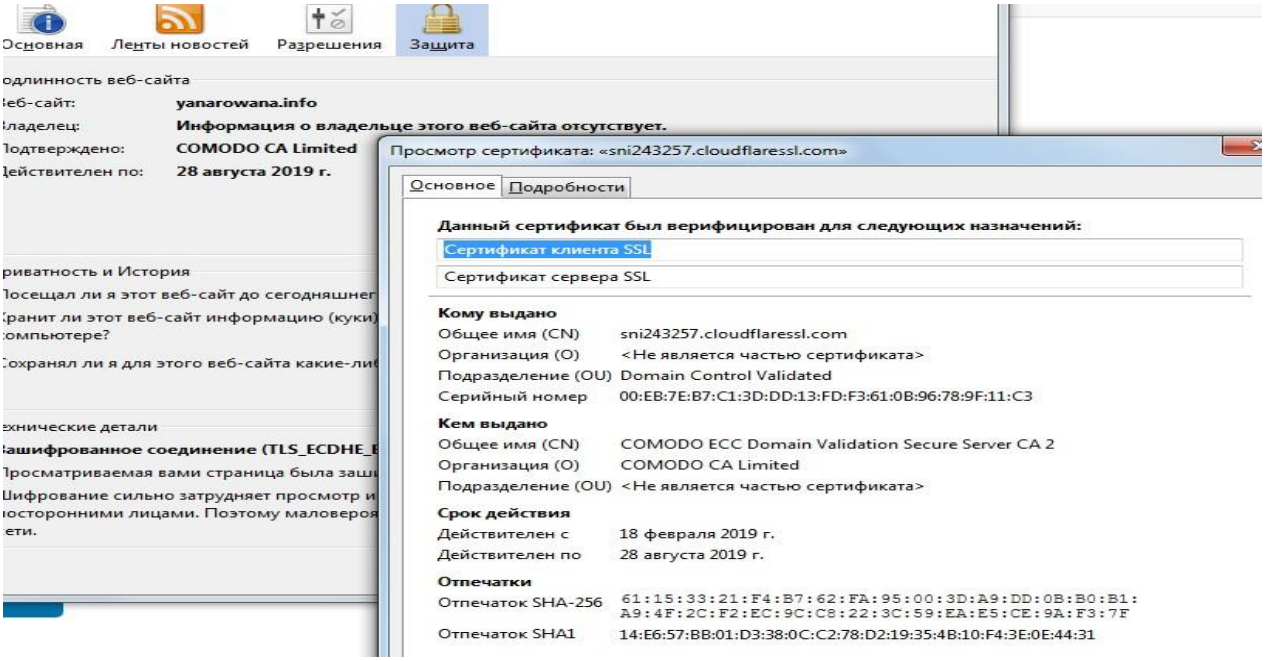


Рис. 3.12. Подробиці про сертифікат

Для захисту від атак типу Session Attack, у цьому випадку SlowLoris, можна закрити з'єднання через 5 секунд, якщо клієнт не відповідає (рисунок 3.13)

. Зазвичай боти просто перебирають діапазони IP-адрес в пошуках відкритих

80 портів і посилають запит HEAD для отримання інформації про WEB-сервер. Можна, можливо легко запобігти такий скан, заборонивши звернення до сервера за IP-адресою (рис. 3.14). Деякі боти використовують різні методи звернення до серверу для спроби визначення його типу або проникнення, по цією причини Усе методи,крім GET, HEAD і POST, можна, можливо безболісно вимкнути (рисунок 3.14).

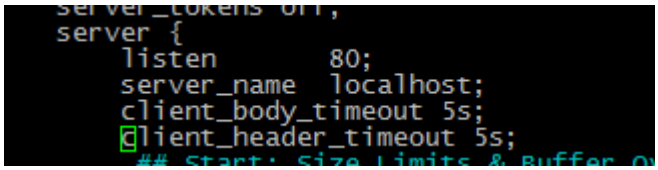


Рис. 3.13. Налаштування зниження тайм-ауту на очікування у відповідь пакету

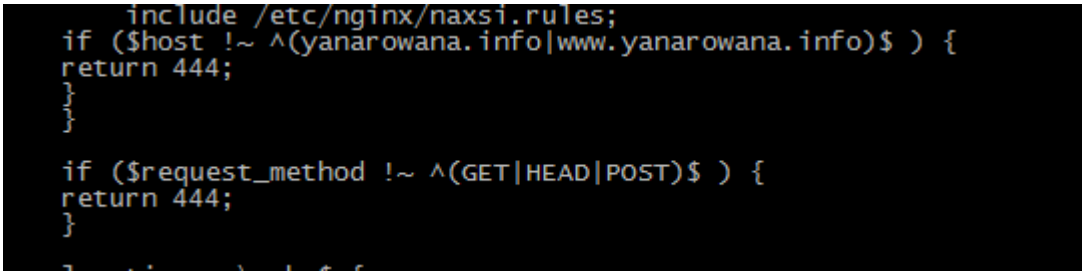


Рис. 3.14. Налаштування заборони звернення до сервера за ip-адресою та відключення необов'язкових методів

CDN Cloudflare допомагає поглинати потік трафіку, пов'язаний затаками DDoS. На додаток до цього вбудованого захисту від DDoS, Cloudflare забезпечує додатковий захист "Under Attack Mode" (рисунок 3.15). Це рівень безпеки, Котрий ви включаєте, коли ваш сайт піддається активною атаці. Коли цей режим включений, він додає додаткові засоби захисту, щоб запобігти влучення потенційно небезпечного HTTP-трафіку на сайт. Результат роботи цього режиму представлений рисунку 3.16.

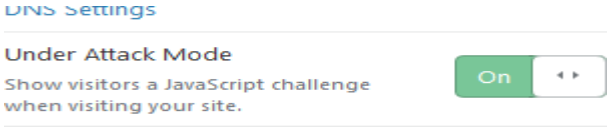


Рис. 3.15. Режим Under Attack Mode

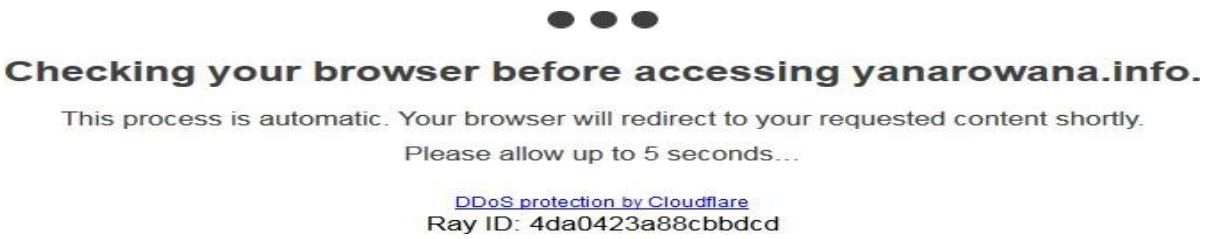


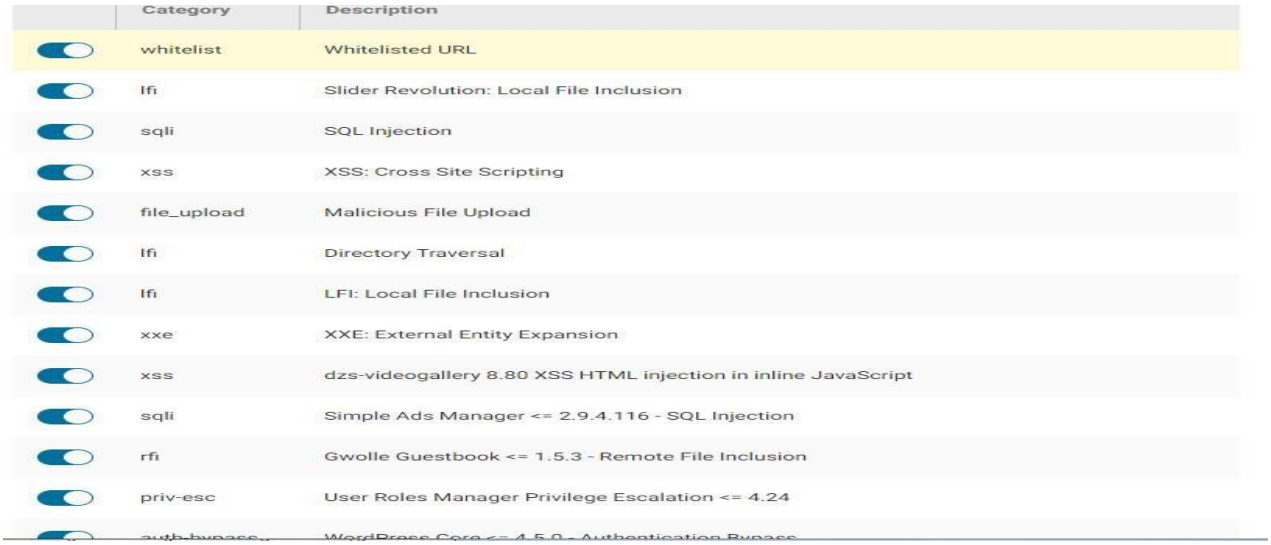
Рис. 3.16. Результат роботи режиму Under Attack Mode

Налаштування стека LEMP і CMS Wordpress представлена на рисунку 3.17:

```
server {
        listen 80;
        listen [::]:80;
                listen 443 ssl;
                ssl_certificate     /etc/ssl/oc.pem;
                ssl_certificate_key     /etc/ssl/pk.key;
        root /usr/ya.info;
        index index.php index.html index.htm index.nginx-debian.html;

        server_name yanarowana.info www.yanarowana.info;

        location / {
            #try_files $uri $uri/ =404;
            try_files $uri $uri/ /index.php$is_args$args;
            include /etc/nginx/naxsi.rules;
        }
        location ~ \.php$ {
                include fastcgi.conf;
                fastcgi_pass unix:/var/run/php/php7.2-fpm.sock;
        }

        location ~ /\.ht {
                deny all;
        }
       location = /favicon.ico { log_not_found off; access_log off; }
       location = /robots.txt { log_not_found off; access_log off; allow all; }
       location ~* \.(css|gif|ico|jpeg|jpg|js|png)$ {
        expires max;
        log_not_found off;
       }
}
```

Рис. 3.17. Налаштування LEMP і Wordpress

У Wordpress CMS необхідно встановити плагін web application firewall Wordfence. перелік правил Wordfence, налаштування захисту від перебору, блокування від ботів представлені на рисунках 3.18, 3.19, 3.20 відповідно.



	Category	Description
ON	whitelist	Whitelisted URL
ON	lfi	Slider Revolution: Local File Inclusion
ON	sqli	SQL Injection
ON	xss	XSS: Cross Site Scripting
ON	file_upload	Malicious File Upload
ON	lfi	Directory Traversal
ON	lfi	LFI: Local File Inclusion
ON	xxe	XXE: External Entity Expansion
ON	xss	dzs-videogallery 8.80 XSS HTML injection in inline JavaScript
ON	sqli	Simple Ads Manager <= 2.9.4.116 - SQL Injection
ON	rfi	Gwolle Guestbook <= 1.5.3 - Remote File Inclusion
ON	priv-esc	User Roles Manager Privilege Escalation <= 4.24
ON	auth-bypass	WordPress Core <= 4.5.0 - Authentication Bypass

Рис. 3.18. перелік правил Wordfence

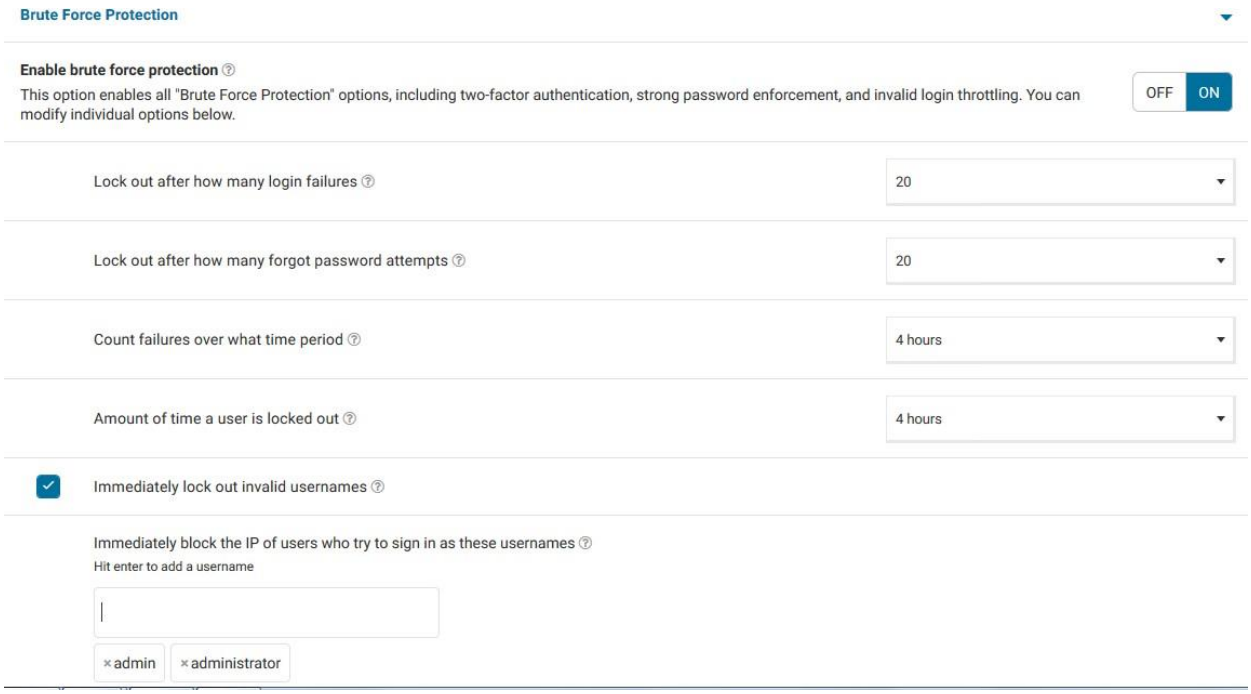


Рис. 3.19. Параметри захисту від перебору (brute force)

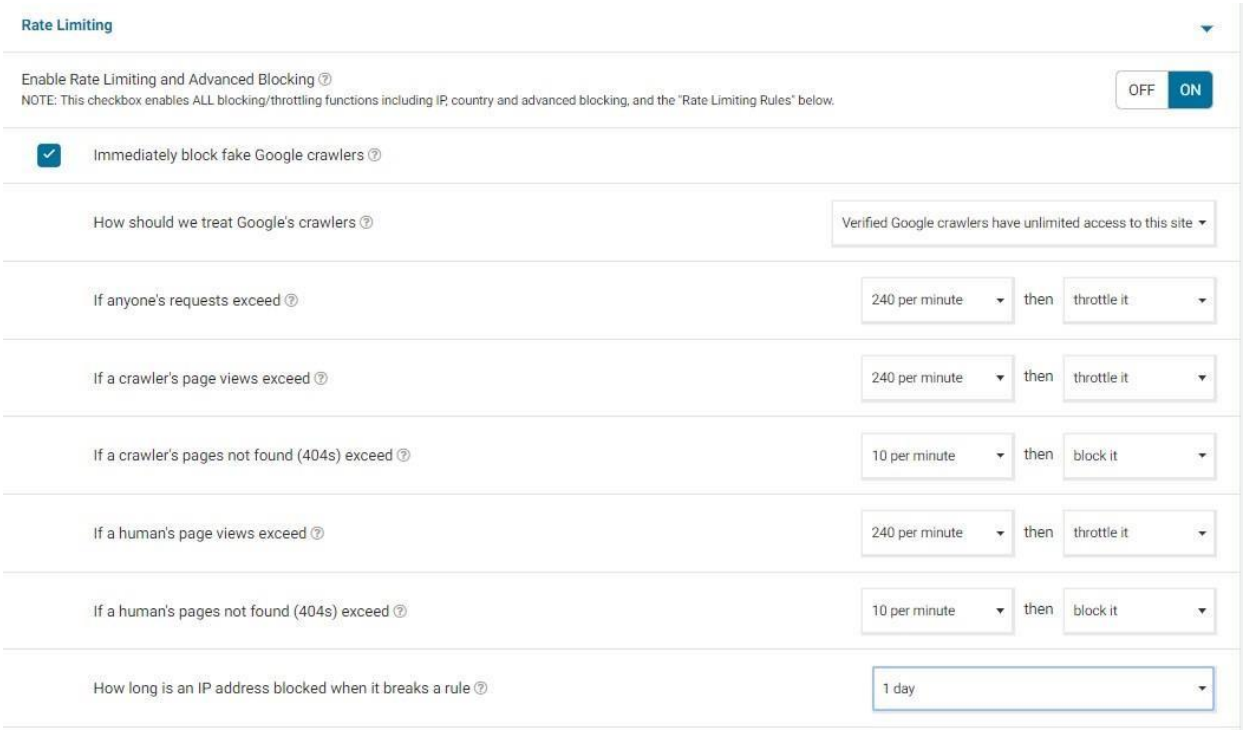


Рис. 3.20. Параметри блокування ботів

Тепер після 20 спроб введення невірного логіну та пароля протягом 4 годин ip-адреса користувача буде заблокована на 4 години, і користувач не матиме можливість увійти на сайт. Також якщо користувач спробує увійти до адмін-панелі використовуючи логін «admin» або «administrator» він буде заблоковано негайно (рисунок 3.21, рисунок 3.22).

Рис. 3.21. Спроба входу в систему, використовуючи логін "admin"

Рис. 3.22. Сторінка блокування wordfence

Ми можемо убезпечити наш сайт від brute force атак змінивши стандартний url авторизації Wordpress wp-admin і wp-login.php на якийсь інший, використовуючи плагін Wps-Hide-Login. Змінюємо стандартний url і зберігаємо налаштування (рисунок 3.23). Тепер при зверненні по стандартному URL/wp-admin або /wp-login.php ми отримуємо помилку 404 (рисунок 3.24).

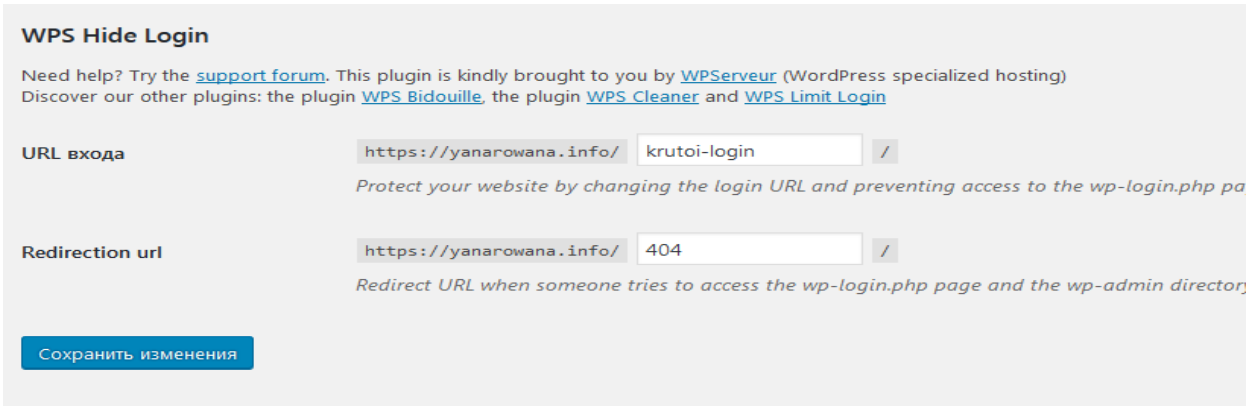


Рис. 3.23. Зміна стандартного URL сторінки авторизацій

Використовуючи різні утиліти для пошуку директорій і файлів на сервері, зловмисники здатні знайти точку входу (рисунок 3.25). Для того, щоб приховати вміст каталогів скористаємося сервісом Cloudfalre,що дозволяє проводити весь вхідний трафік через їх сервер (рисунок 3.26), завдяки також ховається зовнішня ip-адреса нашого сервера (рисунок 3.27). Повторно просканувавши наш сервер,

можна виявити, що реальні директорії і файли були приховані серверами cloudfalre (рисунок 3.28).

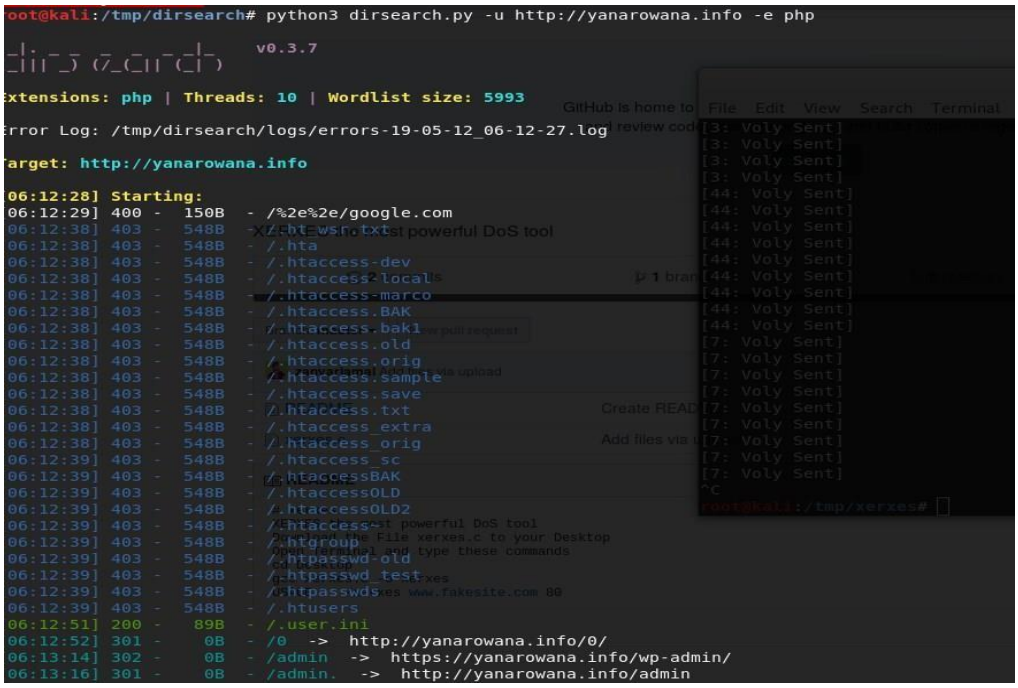
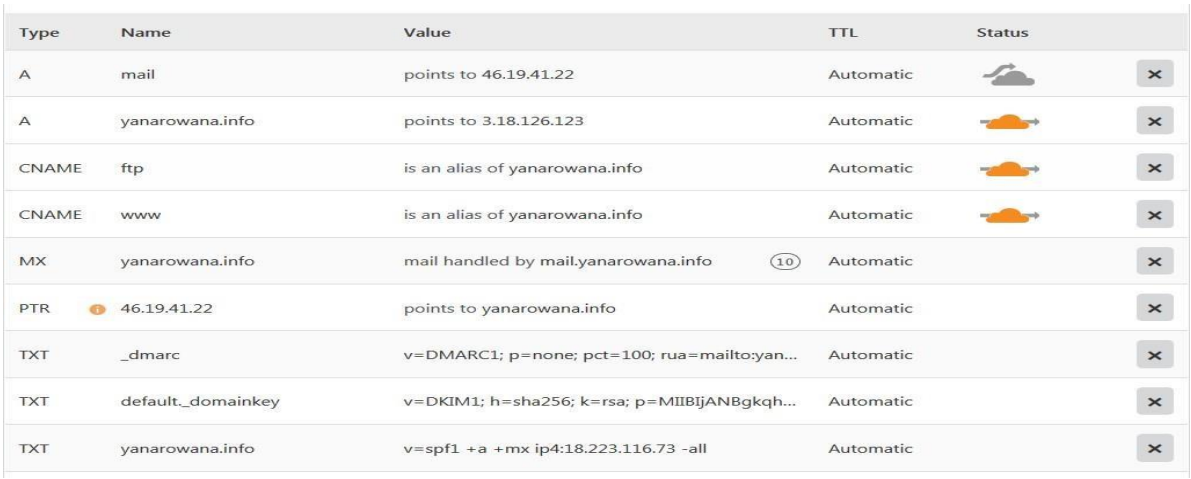


Рис. 3.24. Результат сканування директорій і файлів



Type	Name	Value	TTL	Status
A	mail	points to 46.19.41.22	Automatic	
A	yanarowana.info	points to 3.18.126.123	Automatic	
CNAME	ftp	is an alias of yanarowana.info	Automatic	
CNAME	www	is an alias of yanarowana.info	Automatic	
MX	yanarowana.info	mail handled by mail.yanarowana.info (10)	Automatic	
PTR	46.19.41.22	points to yanarowana.info	Automatic	
TXT	_dmarc	v=DMARC1; p=none; pct=100; rua=mailto:yan...	Automatic	
TXT	default._domainkey	v=DKIM1; h=sha256; k=rsa; p=MIIBIjANBgkqh...	Automatic	
TXT	yanarowana.info	v=spf1 +a +mx ip4:18.223.116.73 -all	Automatic	

Рис. 3.25. Налаштування cloudlfare dns-proxy

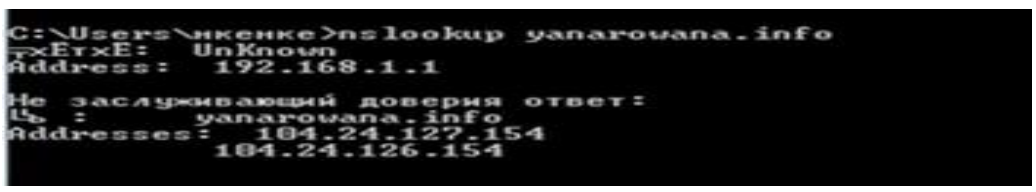


Рис. 3.26. Результат роботи nslookup

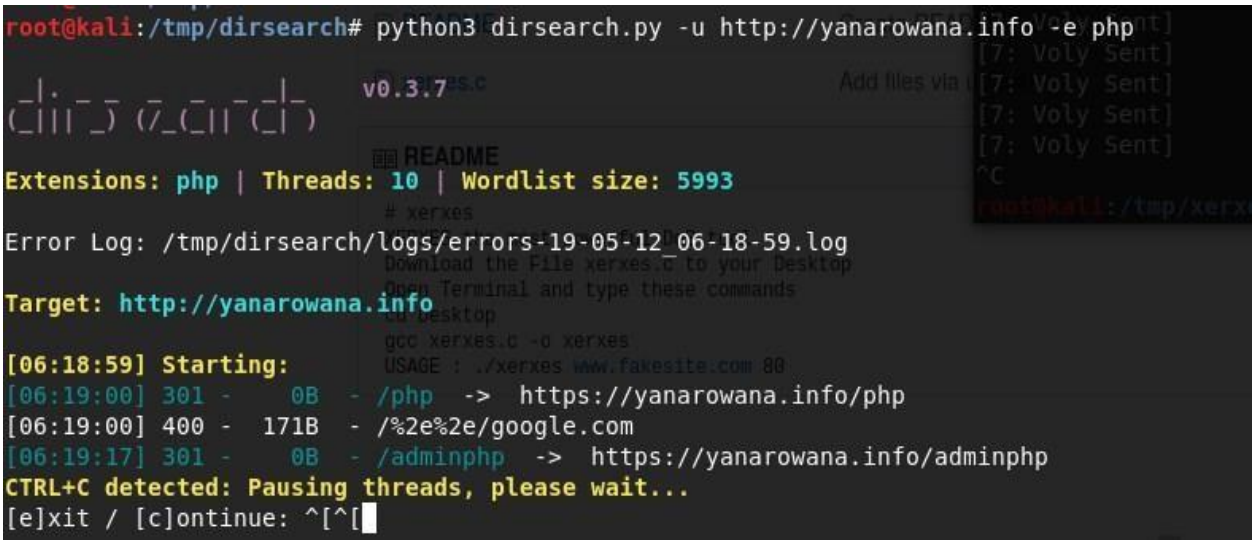


Рис. 3.27. Результат повторного сканування директорій та файлів

Для того щоб ускладнити завдання для потенційногозловмисника, який намагається атакувати наш сайт за допомогою sql-ін'єкцій, рекомендується змінити стандартний префікс таблиць Wordpress. Для цього в панелі управління базами даних phpmyadmin експортуємо копію бази даних “wordpress” у форматі .sql (рисунок 3.29), змінюємо префікс “wp” на будь-який інший (рисунок 3.30), зберігаємо результат і завантажуємо на сервер,

попередньо вилучивши струмінь базу даних (рисунок 3.31), змінюємо префіксу конфігураційному файлі wp-config.php (рисунок 3.32). Переконатись у тому, що префікс змінився, можна, можливо зробивши запит вибірки прямо на сервері (рисунок 3.33).

Рис. 3.28. Експорт бази даних wordpress

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `wordpress`
--

-- --------------------------------------------------------


--
-- Table structure for table `wp_commentmeta`
--

CREATE TABLE `krutoi_commentmeta` (
  `meta_id` bigint(20) UNSIGNED NOT NULL,
  `comment_id` bigint(20) UNSIGNED NOT NULL DEFAULT '0',
  `meta_key` varchar(255) COLLATE utf8mb4_unicode_520_ci DEFAULT NULL,
  `meta_value` longtext COLLATE utf8mb4_unicode_520_ci
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_520_ci;

-- --------------------------------------------------------
```

Рис. 3.29. Редагування скрипта sql

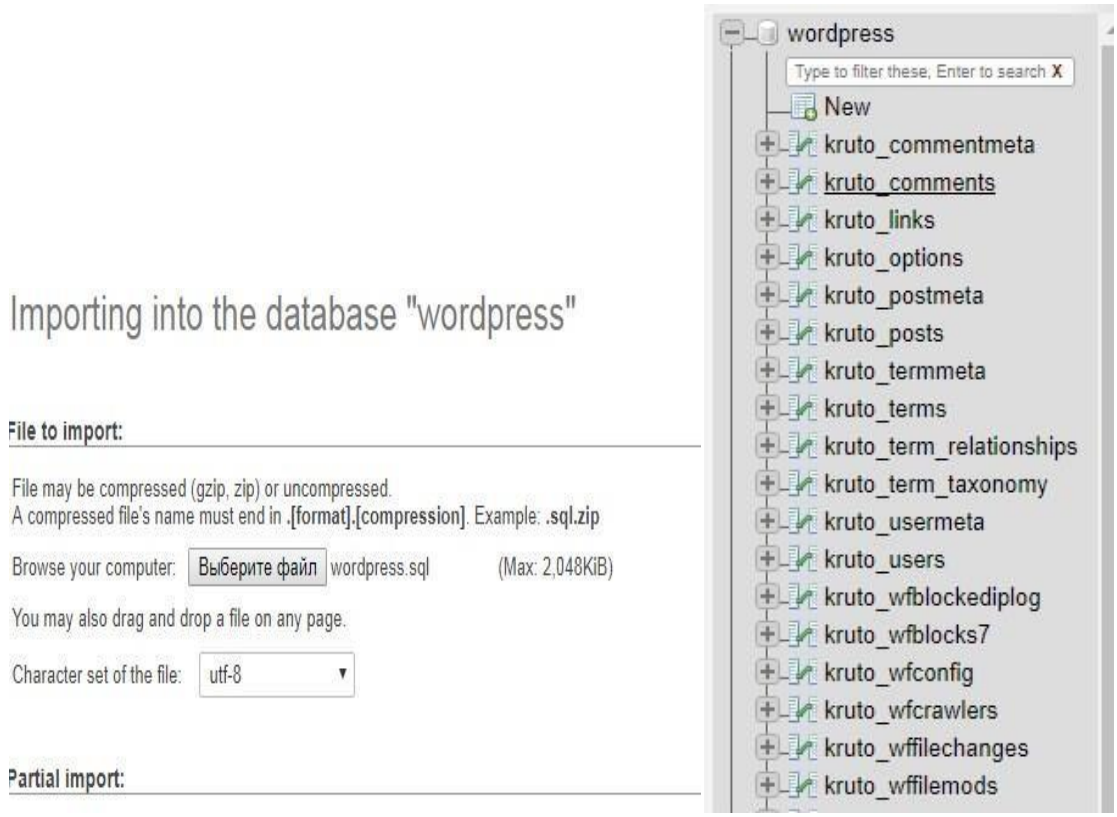


Рис. 3.30.Імпорт відредагованою бази

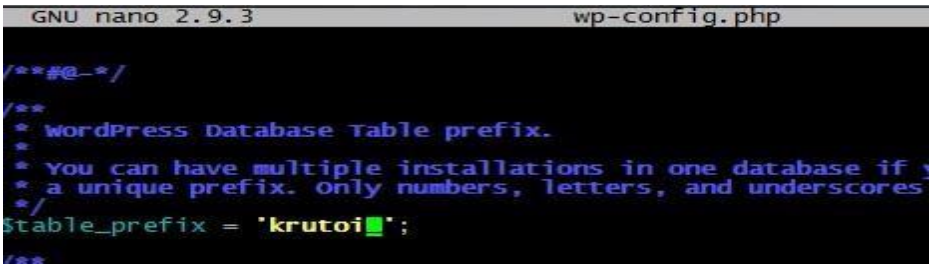


Рис. 3.31.Зміна префікса в конфігураційному файлі wordpress

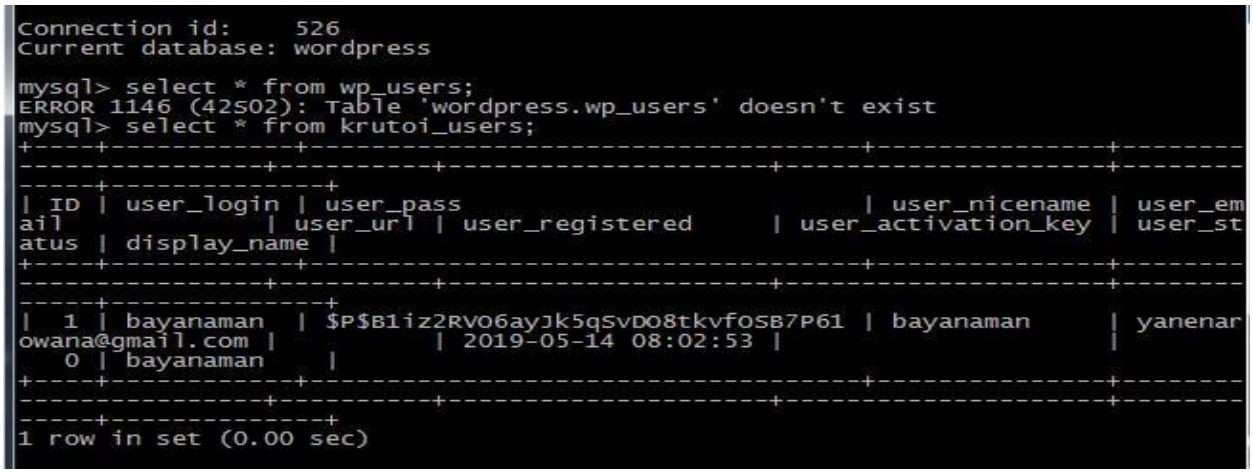
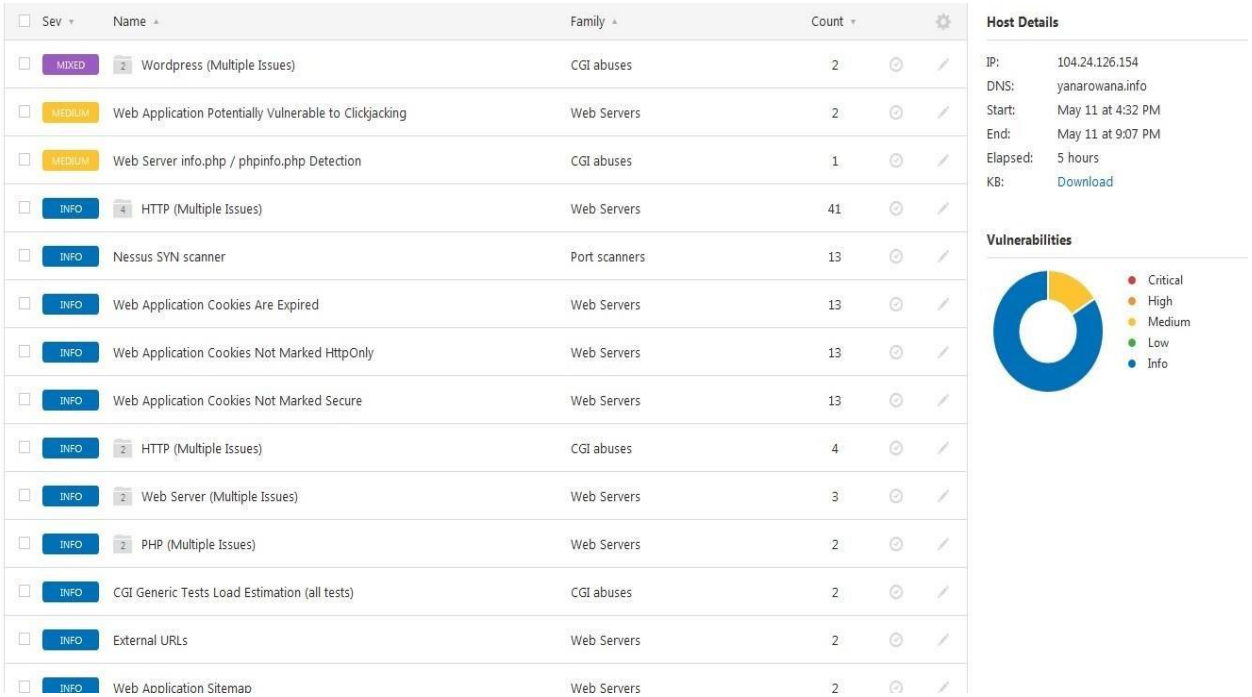


Рис. 3.32. Результат запиту вибірки на сервері

Використовуючи сканер вразливостей, проскануємо сайт на наявність вразливостей (рисунок 3.34).



Sev	Name	Family	Count
MIXED	Wordpress (Multiple Issues)	CGI abuses	2
MEDIUM	Web Application Potentially Vulnerable to Clickjacking	Web Servers	2
MEDIUM	Web Server info.php / phpinfo.php Detection	CGI abuses	1
INFO	HTTP (Multiple Issues)	Web Servers	41
INFO	Nessus SYN scanner	Port scanners	13
INFO	Web Application Cookies Are Expired	Web Servers	13
INFO	Web Application Cookies Not Marked HttpOnly	Web Servers	13
INFO	Web Application Cookies Not Marked Secure	Web Servers	13
INFO	HTTP (Multiple Issues)	CGI abuses	4
INFO	Web Server (Multiple Issues)	Web Servers	3
INFO	PHP (Multiple Issues)	Web Servers	2
INFO	CGI Generic Tests Load Estimation (all tests)	CGI abuses	2
INFO	External URLs	Web Servers	2
INFO	Web Application Sitemap	Web Servers	2

Рис. 3.33. Результат сканування Nessus

Сканер знайшов вразливість user enumeration, завдяки якої зловмисник здатний дізнатися імена користувачів Wordpress і використовувати їх для спроби злому сторінки авторизацій. Результат сканування користувачів представлено рисунку 3.35. Щоб усунути цю «пролом», встановлюємо і активуємо плагін Stop User Enumeration (рисунок 3.36). Після повторного сканування спроби знайти імена користувачів не увінчалися успіхом (рисунок 3.37).

```
[i] User(s) Identified:

[+] bayanaman
 | Detected By: Author Posts - Author Pattern (Passive Detection)
 | Confirmed By:
 |  Rss Generator (Passive Detection)
 |  Wp Json Api (Aggressive Detection)
 |   - https://yanarowana.info/wp-json/wp/v2/users/?per_page=100&page=1
 |  Rss Generator (Aggressive Detection)
 |  Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

Рис. 3.34. Результат сканування імен користувачів

Рис. 3.35. Встановлення плагіна stop user enumeration

Рис. 3.36. Результат повторного сканування імен користувачів

Наступна вразливість, яку виявив сканер називається:

Clickjacking (рисунок 3.38). Clickjacking - це атака, яка змушує користувача клікнути на елемент WEB-сторінки, Котрий невидимий або прихований під іншим елементом. Це може призвести до того, що користувачі мимоволі завантажуватимуть шкідливі програми, відвідуватимуть шкідливі програми WEB-сторінки, надавати облікові дані або конфіденційну інформацію, перекладати гроші або купувати продукти через Інтернет.

Для захисту від цією атаки використовуємо Заголовок X-Frame-Options. Заголовок відповіді X-Frame-Options передається як частина HTTP-відповіді WEB- сторінки, вказуючи, дозволено чи браузеру відображати сторінку всередині тега

<FRAME> або <IFRAME>. Я використовуватиму значення SAMEORIGIN,

що дозволяє відображати поточну сторінку в рамці на іншій сторінці, але лише у межах поточного домену. Для того, щоб увімкнути цю опцію змінюємо конфігураційний файл ngnix.conf (рисунок 3.39).

На рисунках 3.40 і 3.41 представлені заголовки відповіді до і після застосування зміні.

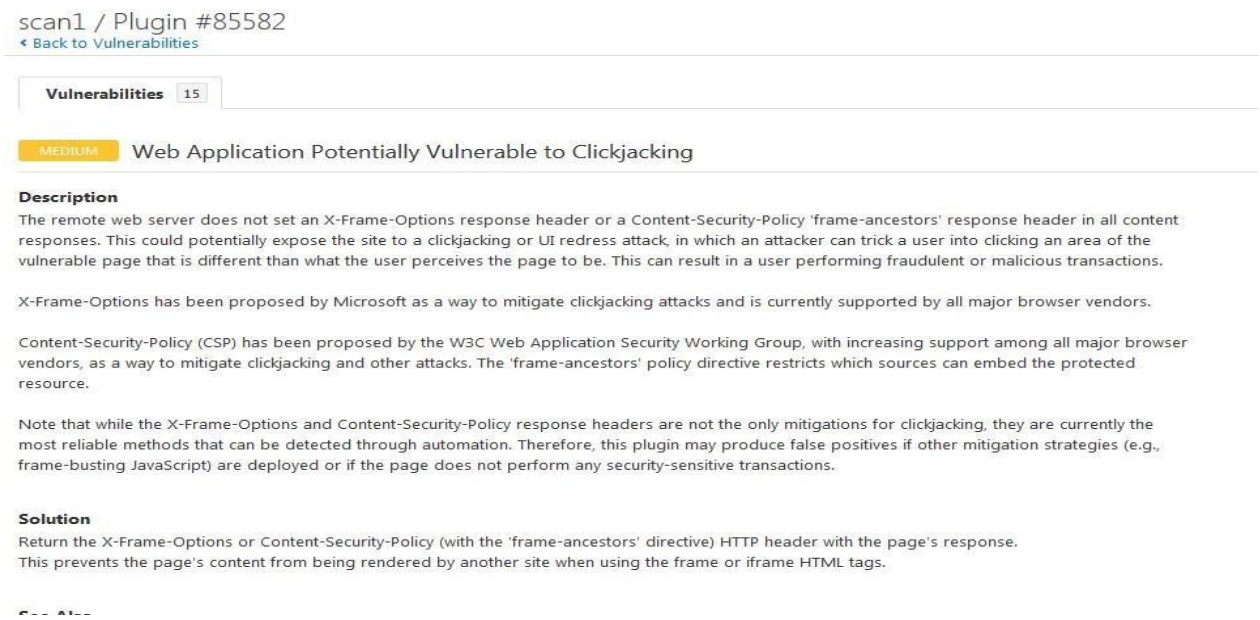


Рис. 3.37. Виявлена сканером вразливість Clickjacking

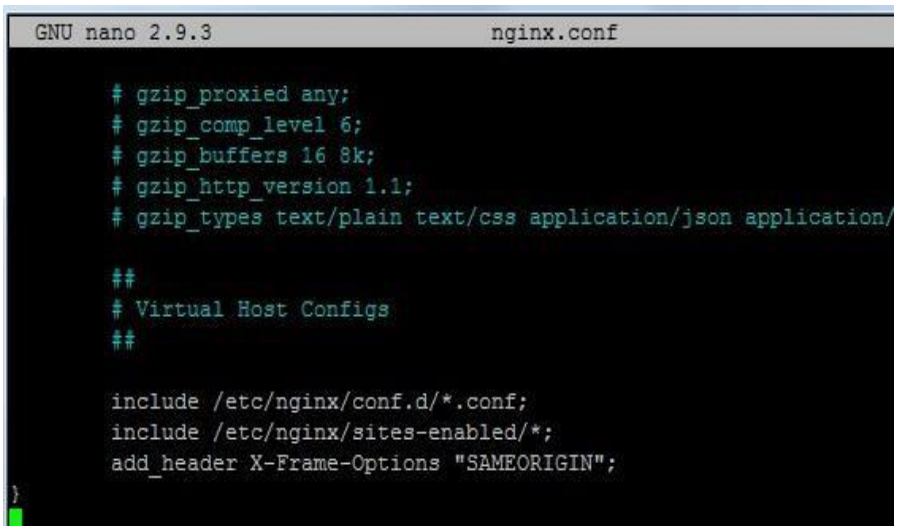


Рис. 3.38. Увімкнення опцій X-Frame-Options

Рис. 3.39. Заголовок відповіді до включення опцій

Рис. 3.40. Заголовок відповіді після включення опцій

Сканер виявив, включений XML-RPC (рисунок 3.42). XML-RPC – це протокол віддалених процедур, Котрий дозволяє віддалено взаємодіяти з вашим веб-сайтом WordPress. Іншими словами, це спосіб керувати сайтом без необхідності входу в систему вручну черезстандартну сторінку "wp-login.php". У спільноті безпеки WordPress було багато питань про XML-RPC. У здебільшого є дві проблеми:

1. xml-rpc можна, можливо використовувати для DDoS-атаки;
2. його можна використовувати для повторного використання комбінацій імені користувача та пароля.

Для відключення цього протоколу використовуємо плагін “disable xml rpc”

(рисунок 3.43) і заборонимо доступ до конфігураційного файлу (рисунок 3.44). Результат повторного сканування можна побачити на рисунку 3.45.

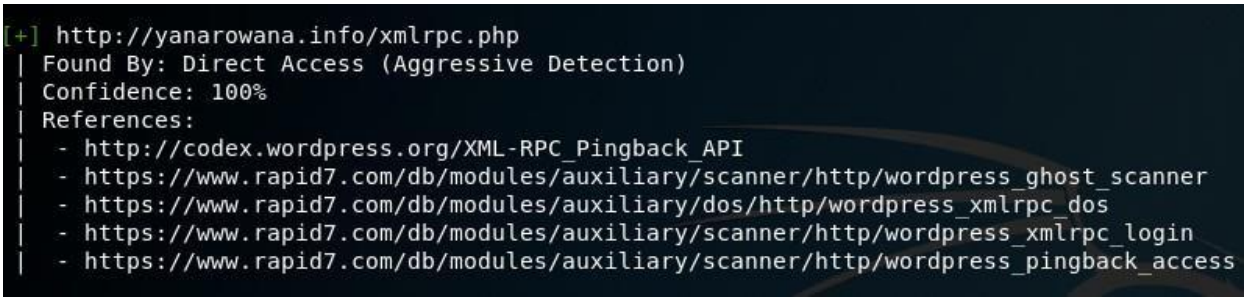


Рис. 3.41. Результат сканування протоколу xml-rpc

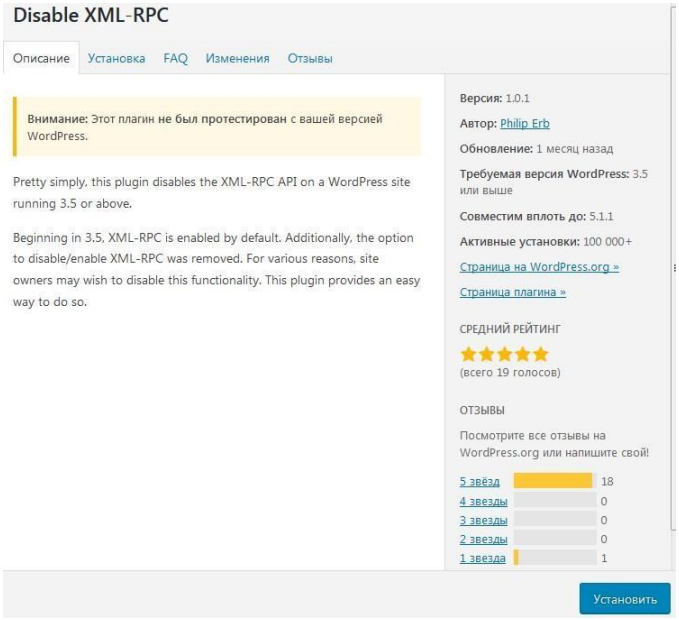


Рис. 3.42. Встановлення плагіна Disable XML-RPC

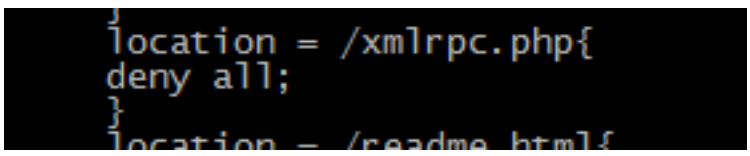


Рис. 3.43. Заборона зверненню до файлу xmlrpc.php

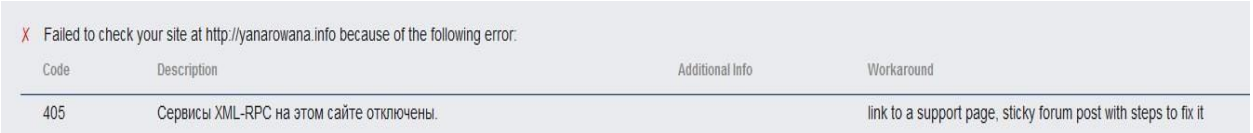


Рис. 3.44. Повторне сканування на наявність протоколу XML-RPC Далі

Змінимо деякі параметри php.ini. Першим справою змінимо значення cgi.fix_pathinfo на 0 (рисунок 3.46), адже налаштування за замовчуванням дуже небезпечна, тому що завдяки їй PHP спробує виконати найближчий файл, який зможе знайти у випадку, коли запитуваний PHP файл не можна знайти. Це дозволить користувачам сформувати PHP запити таким чином, щоб запускати скрипти, до яких вони не повиннібути доступу.

Параметр session.cookie_lifetime встановлюємо на 0 (рисунок 3.47). 0 має особливе значення. Він повідомляє браузеру не зберігати cookie в постійне сховище. Отже, коли браузер закривається, сесійні cookie відразу ж видаляються. Якщо задати значення відмінне від 0, це може дозволити іншим користувачам використовувати ці cookie.

Включаємо опцію session.use_strict_mode (рисунок 3.48). Це не дозволить сесійному модулю використовувати неініціалізовані ідентифікатори сесій. Іншими словами, сесійний модуль буде приймати тільки коректні ідентифікатори, згенеровані їм ж ібуде нехтувати ідентифікатори створені на боці користувача.

Через особливостей специфікації cookie, атакуючий може зробити cookie з ідентифікатором сесії, що не видаляється за допомогою локальної бази cookie або JavaScript-ін'єкцією. session.use_strict_mode може не дати атакуючому використовувати цей ідентифікатор.

Включаємо опцію session.cookie_httponly (рисунок 3.49). Ця опція забороняє доступ до сесійної cookie для JavaScript. Ця опція запобігає крадіжку cookie з допомогою JavaScript-ін'єкції.

Включаємо опцію session.cookie_secure (рисунок 3.50). Вона дозволяє отримувати доступ до cookie ідентифікатора сесії лише при використанні протоколу HTTPS. Якщо ваш сайт використовує лише протокол HTTPS, вам

необхідно увімкнути цю опцію.

В опції disable_function передаємо значення всіх потенційно небезпечних функції (рисунок 3.51).

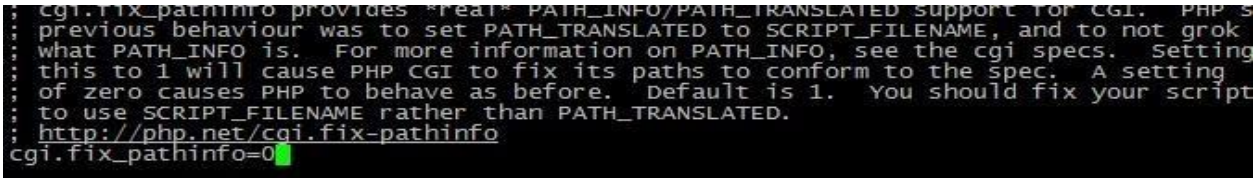


Рис. 3.45. Налаштування параметра cgi.fix_pathinfo

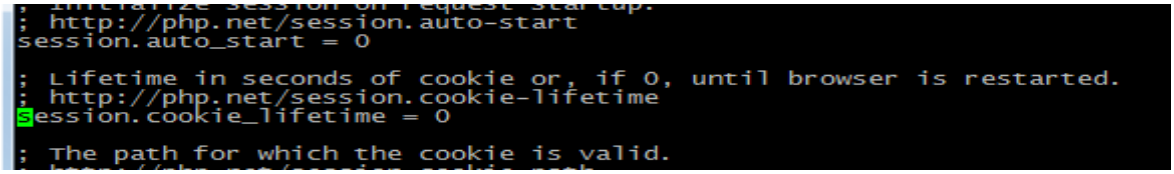


Рис. 3.46. Налаштування параметра session.cookie_lifetime

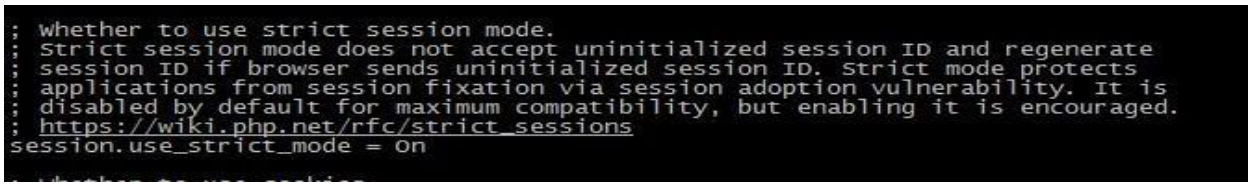


Рис. 3.47. Налаштування параметра session.use_strict_mode

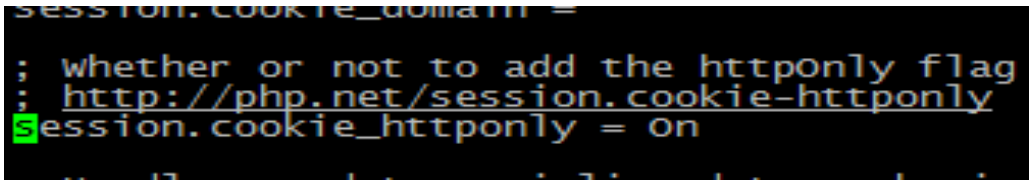


Рис. 3.48. Налаштування параметра session.cookie_httponly

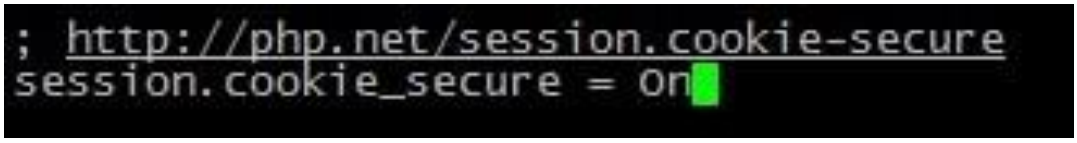


Рис. 3.49. Налаштування параметра session.cookie_secure

```
; http://php.net/open-basedir
open_basedir =

; This directive allows you to disable certain functions for security reasons.
; It receives a comma-delimited list of function names.
; http://php.net/disable-functions
disable_functions =phpinfo, system, mail, exec, pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,$

; This directive allows you to disable certain classes for security reasons.
; It receives a comma-delimited list of class names.
; http://php.net/disable-classes
```

Рис. 3.50. Налаштування параметра disable_functions

При повторному скануванні можна побачити, що вразливість усунені (рисунок 3.52):

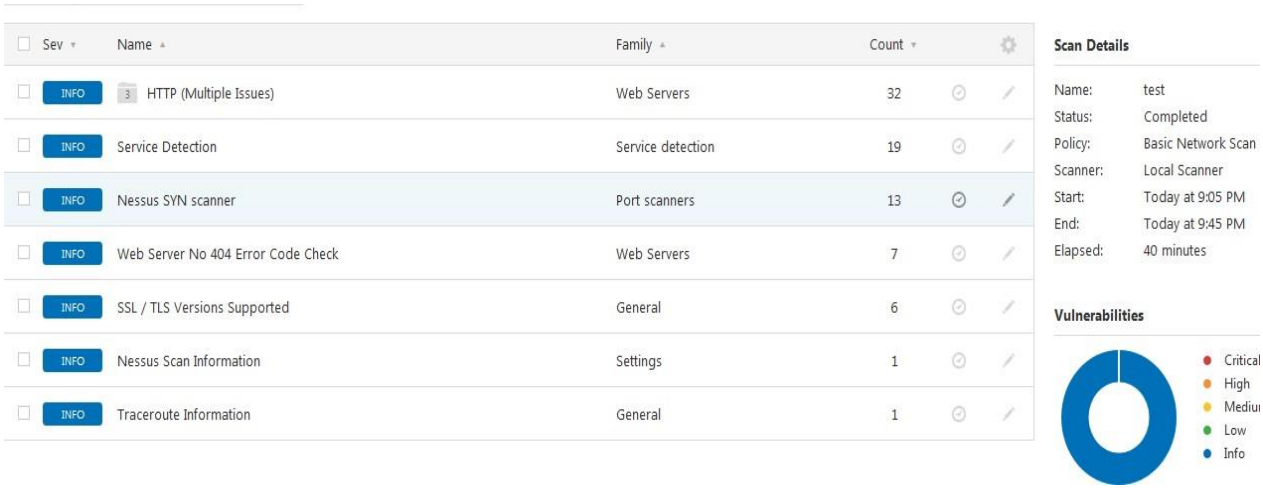


Рис. 3.51. Результат повторного сканування Nessus

3.3 Налаштування і забезпечення безпеки поштового сервера

Встановлюю iRedMail за допомогою оболонки bash. Підсумкові параметри установки представлені на рисунку 3.53.

```
* Storage base directory:                /var/vmail
* Mailboxes:
* Daily backup of SQL/LDAP databases:
* Store mail accounts in:                MySQL
* Web server:                            Nginx
* First mail domain name:                yanarowana.info
* Mail domain admin:                     postmaster@yanarowana.info
* Additional components:                 Roundcubemail netdata iRedAdmin

< Question > Continue? [y|N]
```

Рис. 3.52. Параметри установки iRedMail

Для успішною доставки листів додаю запис SPF в панелі керування DNS записами (рисунок 3.54). Копіюю відкритий ключ DKIM з файлу iRedMail.tips (рисунок 3.55). Додаю його у вміст запису DKIM в тій ж панелі (рисунок 3.56). Задаю політику DMARC (рисунок 3.57).Результат тестування SPF, DKIM, DMARC представлений на рисунку 3.58.

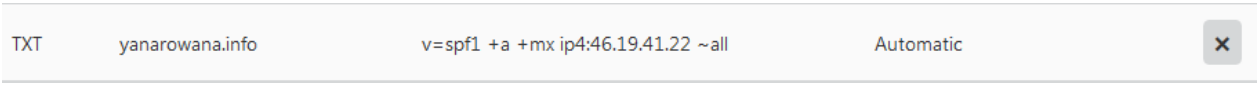


Рис. 3.53. Запис SPF на панелі управління DNS записами

Рис. 3.54. Зміст файлу iRedMail.tips

Рис. 3.55. Запис DKIM на панелі управління DNS записами

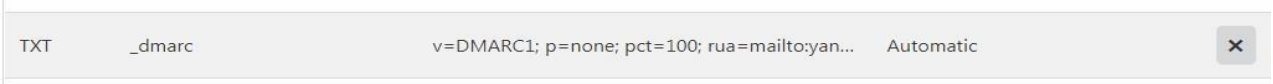


Рис. 3.56. Запис DMARC на панелі управління DNS записами

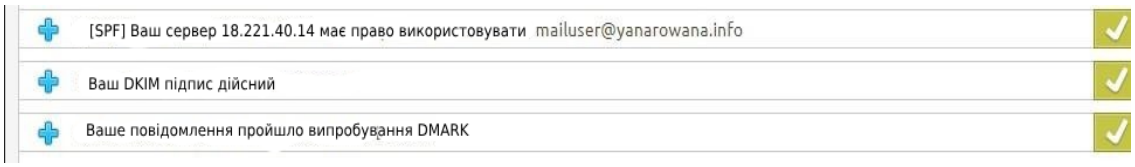


Рис. 3.57. Результат тестування SPF, DKIM, DMARC

Для боротьби зі спамом було відредаговано конфігураційний файл Postfix. На рисунку 3.59 представлені правила для блокування листів, якщо у клієнта

SMTP немає запису PTR, коли ім'я хоста HELO / EHLO не має A записи і не MX записи в DNS, ім'я хоста клієнта SMTP не має дійсного A запису. Були додані правила для блокування листів, перерахованих у публічних чорних списках (рисунок 3.60).

```
# HELO restriction
smtpd_helo_required = yes
smtpd_helo_restrictions =
    permit_mynetworks
    permit_sasl_authenticated
    check_helo_access pcre:/etc/postfix/helo_access.pcre
    reject_non_fqdn_helo_hostname
    reject_unknown_helo_hostname

# Sender restrictions
smtpd_sender_restrictions =
    reject_unknown_sender_domain
    reject_non_fqdn_sender
    reject_unlisted_sender
    reject_unknown_reverse_client_hostname
    reject_unknown_client_hostname
    reject_unknown_sender_domai
    permit_mynetworks
    permit_sasl_authenticated
    check_sender_access pcre:/etc/postfix/sender_access.pcre
```

Рис. 3.58. Правила блокування спаму

```
smtpd_recipient_restrictions =
    reject_non_fqdn_recipient
    reject_unlisted_recipient
    check_policy_service inet:127.0.0.1:7777
    permit_mynetworks
    permit_sasl_authenticated
    reject_unauth_destination
    reject_rhsbl_helo dbl.spamhaus.org,
    reject_rhsbl_reverse_client dbl.spamhaus.org,
    reject_rhsbl_sender dbl.spamhaus.org,
    reject_rbl_client zen.spamhaus.org
```

Рис. 3.59. Правила блокування листів, перелічених у публічнихчорних списках

Крім цього, для боротьби зі спамом, була налаштована перевірка заголовків і тел вхідних листів. Для цього, першим справою, я увімкнула параметри для перевірки заголовків і вмісту (рисунок 3.61). Були додані правила блокування листів на основі регулярних виразі (рисунок 3.62).

```
header_checks = pcre:/etc/postfix/header_checks
body_checks = pcre:/etc/postfix/body_checks
```

Рис. 3.60. Параметри для перевірки вмісту листи

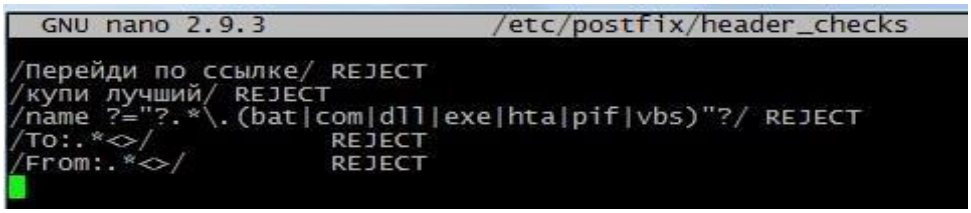


Рис. 3.61. Правила блокування листів на основі регулярних виразі

Для більш ефективної боротьби зі спамом було настроєно Spamassassin – були додані користувальницькі правила перевірки вхідних листів (рисунок 3.63), налаштовані правила для відхилення прийому листа, якщо то набирає 7 балів Spamassasin (рисунок 3.64).

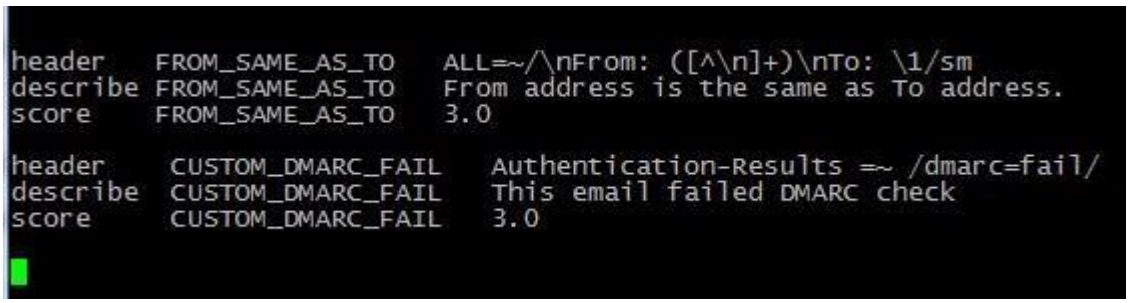


Рис. 3.62. Користувальницькі правила Spamassasin

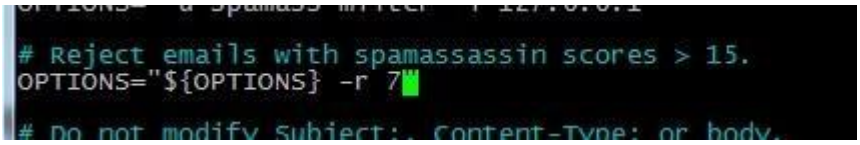


Рис. 3.63. Правила для відхилення листів

ВИСНОВКИ ДО РОЗДІЛУ 3

У даної главі було продемонстровано захист WEB і поштових серверів. Спочатку, був продемонстровано процес створення інстансу Amazon EC2 – вибір типу інстансу, налаштування групи безпеки AWS, створення пари ключів SSH для безпечної авторизації. Потім я безпечно розгорнув WEB-сервер Nginx, зібравши його з вихідного коду. Разом з Nginx був встановлений WAFnaxsi. Я активував його, поставивши деякі правила блокування. Завдяки Cloudflare CDN я вирішив безліч проблем безпеки – встановив TLS сертифікат, забезпечив захист від DDoS-атак та захист від сканерів каталогів. Для забезпечення зручного створення, редагування і управління вмістом WEB-додатки була встановлена та налаштована Wordpress CMS. Використовуючи плагіни Wordpress, я досяг ще одного рівня захисту. WAFWordfence, встановлений у вигляді плагіна на Wordpress допоміг позбутися поширених атак на WEB- програми та встановив надійний захист від перебору паролів. Після цього були виконані стандартні дії для забезпечення безпеки WEB- сервера – змінено стандартну адресу входу в адмін панель, змінено префікс таблиць Wordpress. Використовуючи сканер вразливостей, я знайшов решта вразливості і усунув їх.

Під час налаштування поштового сервера я показав, як досягти того, щоб вихідні листи були надіслані та не потрапляли до спаму. Для цього необхідно було увімкнути механізми SPF, DKIM, DMARC. Задав деякі правила блокування спаму в конфігураційному файлі postfix. Для більше надійною захисту від спаму використав параметри перевірки заголовків та тексту листів, а так ж Spamassassin для оцінки листи.

ВИСНОВКИ

Сьогодні важко заперечувати той факт, що технології, засновані на хмарних обчисленнях, надзвичайно популярні та активно розвиваються. Під технологією хмарних обчислень розуміється технологія, що дозволяє об'єднувати IT-ресурси різних апаратних платформ в єдине ціле та надавати користувачу доступ до них через мережу Інтернет. Хмарні сервіси надають користувачам доступ до своїх ресурсів через мережу Інтернет за допомогою безкоштовних або умовно безкоштовних хмарних додатків, програми та апаратні вимоги яких не потребують наявності у клієнта високопродуктивних комп'ютерів. Сьогодні багато компаній переносять свої проекти в хмару, використовуючи різні хмарні сервіси, та повністю довіряють захист проектів провайдерам хмарних послуг. Але це не є універсальним рішенням від усіх загроз, які можуть завдати шкоди інформаційним активам. Як WEB, так і поштові сервери, як цінні інформаційні активи, потребують захисту від атак. У своїй кваліфікаційній роботі я розгорнула та забезпечила безпеку WEB та поштових серверів на платформі хмарних обчислень Amazon Web Services.

Для забезпечення захисту WEB-сервера я вжила заходів, описаних у другому розділі цієї кваліфікаційної роботи. WEB-сервер повинен приймати підключення тільки за протоколами HTTP і HTTPS.

Тому першим кроком я встановила правила в групах безпеки AWS. Я встановила правила, які дозволяють потоку HTTP-трафіку проходити через порт 80 з вихідним IP-адресою 0.0.0.0/0, тобто з будь-якого місця, а потоку HTTPS-трафіку - через порт 443 з вихідним IP-адресою 0.0.0.0/0. Ці вхідні правила дозволяють трафіку з IPv4 адресами. Щоб дозволити трафіку IPv6, я додала вхідні правила на ті ж порти з адресою джерела ::/0. Потім створила пару ключів SSH для безпечного підключення до серверу. На вже створеній віртуальній машині був розгорнутий WEB-сервер Nginx, зібраний з вихідного коду, що включає в себе WAF Naxsi та заздалегідь вимкнені потенційно небезпечні модулі. Для захисту від поширених атак, таких як SQL-ін'єкція і XSS-атаки, були включені правила

блокування Naxsi. Для забезпечення безпечного підключення і використання протоколу HTTPS на сервері був встановлений TLS-сертифікат, наданий Cloudfalre CDN.

Також завдяки Cloudfalre CDN було включено захист від DDoS-атак і забезпечена захист від сканерів каталогів. Було змінено налаштування за замовчуванням конфігураційних файлів WEB-сервера для захисту від атаки SlowLoris. Для забезпечення зручного створення, редагування і управління вмістом WEB-додатків була встановлена і налаштована WordPress CMS. Для створення додаткового шару захисту у WordPress було встановлено Wordfence, брандмауер рівня програми. Так само, як і Naxsi, він захищає сайт від поширених атак, таких як SQL-ін'єкція і XSS-атаки. Важливою функцією Wordfence виявилася захист від перебору логінів і паролів. Функція була налаштована так, що після 20 спроб введення невірного логіну та пароля протягом 4 годин IP-адреса користувача блокується на 4 години. Після цього було виконано стандартні дії для забезпечення безпеки WEB-сервера: змінено стандартну адресу входу в адмін-панель, змінено префікс таблиць WordPress.

За допомогою сканера WEB-вразливостей були виявлені та усунені вразливості, такі як clickjacking, WordPress user enumeration, вразливості, пов'язані з xml-rpc.php. Відредагувавши файл php.ini, я забезпечила безпеку сесій і вимкнула виконання потенційно небезпечних PHP-функцій.

Після завершення роботи з WEB-сервером я розгорнула поштовий сервер. Я досягла успішної і безпечної доставки вихідної пошти, використовуючи кілька механізмів. Спочатку я додала і правильно налаштувала SPF-записи, DKIM-записи в панелі управління DNS-записами. Я отримала публічний ключ DKIM з файлу налаштувань iRedMail.tips. Для боротьби зі спамом були використані засоби, що надаються Postfix і SpamAssassin. У конфігураційному файлі Postfix були задані наступні правила:

- відхилити електронну пошту, якщо у клієнта SMTP немає записів PTR;
- увімкнути обмеження HELO/EHLO Hostname в Postfix;
- відхилити електронну пошту, якщо ім'я хоста клієнта SMTP не має

дійсного запису A;

- використовувати загальнодоступні антиспамові чорні списки;

- блокувати спам в електронній пошті, перевіряючи його з допомогою SpamAssassin.

Щоб забезпечити безпеку вхідної пошти, було встановлено fail2ban. Це дозволяє блокувати IP-адреси, з яких проводилися невдалі спроби аутентифікації. Крім того, для шифрування інтернет-трафіку використовувався TLS.

Це лише загальний опис заходів, прийнятих для забезпечення безпеки WEB- та поштових серверів в хмарному середовищі AWS. Існує багато інших аспектів безпеки, які можуть бути враховані та реалізовані залежно від потреб проекту та рівня захисту, який вимагається.

ПЕРЕЛІК ПОСИЛАНЬ

1. Cloud Data Center Trends To Watch For In 2015 [Електронний ресурс] / Sarah Tanksalvala Режим доступу : World Wide Web. – URL: https://www.forbes.com/sites/huawei/2015/04/16/cloud-data-center-trends-towatch-for-in-2015/ x
2. [Електронний ресурс] / Laura Pevehouse – Режим доступу : World Wide Web. – URL: https://www.delltechnologies.com/en-us/blog/what-companies-growingmore-than-50-percent-faster-are-investing-in/
3. Хмарні обчислення, Integrity Systems.[Електронний ресурс]. Доступно: http://integritysys.com.ua/solutions/pricatecloud-solution. Дата звернення: Січ. 27, 2020.
4. Таненбаум Е. Сучасні операційні системи / Таненбаум Е - СПб.: Изд. Пітер, 2002. – Р. 74-75.
5. Six Advantages of Cloud Computing [Електронний ресурс] – Режим доступу : WorldWideWeb.URL:https://docs.aws.amazon.com/whitepapers/latest/awsoverview/six-advantages-of-cloud-computing.html
6. Gartner Forecasts Worldwide Public Cloud End-User Spending to Grow 18% in 2021 [Електронний ресурс] – Режим доступу : World Wide Web. – URL: https://www.gartner.com/en/newsroom/press-releases/2020-11-17-gartnerforecasts-worldwide-public-cloud-end-user-spending-to-grow-18-percent-in-2021
7. Netflix Case Study [Електронний ресурс] – Режим доступу : World Wide Web. URL: https://aws.amazon.com/solutions/case-studies/netflix-case-study/
8. Using multi-factor authentication (MFA) in AWS [Електронний ресурс] –Режим доступу : World Wide Web. – URL:
9. https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa.html 13. OWASP is pleased to announce the release of the OWASP Top 10 – 2017
10. [Електронний ресурс] – 2017 – Режим доступу : World Wide Web. – URL: https://owasp.blogspot.com/2017/11/owasp-is-pleased-to-announce-releaseof.html