НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

УДК 004.492

ПОГОДЖЕНО

Декан факультету
Інформаційних технологій
_____/ Глазунова О.Г., д.п.н, проф. /
підпис ПІБ, вчене звання і ступінь

«___»_____2024 р.

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ

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

«___»_____2024 р.

МАГІСТЕРСЬКА РОБОТА

На тему: «Розробка системи захисту від Doss-атак»

Спеціальність: 123 «Комп’ютерна інженерія»

Освітня програма: Комп’ютерні системи захисту інформації

Керівник дипломного проекту: / Касаткін Д. Ю. /
підпис ПІБ

Виконав: / Швень Ю. В. /
підпис ПІБ

КИЇВ-2024

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

«__» _____ 2024 р.

ЗАВДАННЯ

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ РОБОТИ СТУДЕНТУ

Швеню Юрію Володимировичу

(прізвище, ім'я, по батькові)

Спеціальність (напрям підготовки): 123 «Комп'ютерна інженерія».

Освітня програма: комп'ютерні системи захисту інформації

Тема магістерської роботи: «Розробка системи захисту від Doss-атак»

затверджена наказом ректора НУБІП України від " 1 " листопада 2024 № 1859 "С"

Термін подання завершеної роботи на кафедру _____

Вихідні дані до магістерської роботи: код системи, VS Code.

Перелік питань, що підлягають дослідженню:

1. Аналіз предметної області для дослідження методів захисту від Doss - атак
2. Проектування системи захисту інформації
3. Побудова системи захисту від Doss-атак

Дата видачі завдання " 1 " листопада 2024 р.

Керівник магістерської роботи _____ / Касаткін Д. Ю. к.п.н., доцент /

(підпис) (ПІБ, вчене звання і ступінь)

Завдання прийняв до виконання _____ / Швень Ю. В. /

(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Постановка задачі магістерської роботи		Виконано
2	Аналіз предметної області		Виконано
3	Проектування системи		Виконано
4	Реалізація системи		Виконано
5	Тестування розробленої системи		Виконано
6	Оформлення пояснювальної записки		Виконано

Студент _____ / Швень Ю. В. /

підпис ПІБ

Керівник проекту (роботи) _____ / Касаткін Д. Ю. /

підпис ПІБ

РЕФЕРАТ

Пояснювальна записка: 100с., 19 рис., 4 додатка, 39 використаних джерела.

ЗАХИСТ. DOSS, DOS, БЕЗПЕКА, ІНФОРМАЦІЯ, РОЗПІЗНАВАННЯ ЗАГРОЗ.

Мета роботи - розробка системи захисту від дос-атак, що полягає у забезпеченні найвищого рівня безпеки в онлайн-середовищі. Ця система призначена для виявлення, запобігання та відвернення можливих атак на веб-ресурси та інформацію, що зберігається на них. Головна мета - забезпечити надійний захист від різноманітних видів дос-атак, таких як DDoS-атаки, SQL-ін'єкції, кросс-сайт скриптінг та інші.

Об’єкт – система захисту від Doss-атак.

Предмет – програмний додаток з відкритим кодом для виявлення Doss-атак.

Робота складається з трьох розділів.

У першому розділі проведений аналіз предметної області, розглянута різниця між Doss та Dos.

У другому розділі розглянуто середовища розробки та проведено опис архітектури та функціональних вимог до розроблюваної системи захисту.

У третьому розділі детально описаний процес створення систем захисту.

В результаті виконання магістерської роботи проведено аналіз, дослідження та розробку системи від Doss-атак та розроблені рекомендації щодо захисту.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

DoS – Заперечення обслуговування (Denial of Service).
DDoS – Розподілена атака заперечення обслуговування (Distributed Denial of Service).
DOSS – (Denial of Service and Spoofing Attack) - це тип кібератаки, яка спрямована на переповнення або перевантаження мережевого обладнання, серверів або комп'ютерних систем шляхом надмірного відправлення запитів або даних.
PHP – Hypertext Preprocessor (PHP) - скриптова мова програмування.
IoT – Розумний простір (Internet of Things) - мережа фізичних пристроїв, які взаємодіють між собою та з Інтернетом.
IDE – Інтегроване середовище розробки (Integrated Development Environment) - програмний інструментарій для розробки програмного забезпечення.
TCP – Протокол керування передачею (Transmission Control Protocol) - протокол мережевого рівня, використовуваний для надання надійного та послідовного з'єднання між пристроями в мережі.
IPS – Система запобігання вторгненням (Intrusion Prevention System) - система, що виявляє та захищає мережу від небажаних вторгнень та атак.
CDN – Мережа доставки контенту (Content Delivery Network) - глобальна мережа серверів, що забезпечує швидку доставку веб-контенту користувачам за допомогою розподіленого кешування.

ВСТУП

Актуальність і практична значущість обраної теми ВКР - У сучасному світі, де комп'ютерні мережі відіграють важливу роль у різних сферах життя, забезпечення безпеки цих мереж стає все більш актуальною проблемою. Зростання кількості і складності кібератак, зокрема атак з відмовою у обслуговуванні, ставить під загрозу нормальну функціонування комп'ютерних систем і мереж. Для підтримки безперебійної роботи і забезпечення доступності сервісів та ресурсів важливо розробляти ефективні методи захисту від таких атак.

Метою даної дипломної роботи є дослідження проблеми захисту від атак з відмовою у обслуговуванні, зосередження на конкретному типі атаки – DOSS та розроблення програмного рішення для їх запобігання. Головним завданням є виявлення основних проблем та викликів, пов'язаних з DOSS-атаками, аналіз існуючих підходів захисту та розробка нового алгоритму для ефективного запобігання цим атакам.

Об'єктом дослідження є системи та мережі, що піддаються ризику DOSS-атак, а також методи захисту від цих атак.

Предметом дослідження є розробка програмного рішення для запобігання DOSS-атакам шляхом виявлення та блокування небажаного трафіку.

В основі дослідження лежать наукові праці, статті, публікації та документація, що стосуються кібербезпеки, мережевих атак, методів захисту, а також існуючих алгоритмів та технологій, пов'язаних з DOSS-атаками.

Для досягнення поставленої мети використовувалися наступні методи дослідження: аналіз літературних джерел, вивчення існуючих підходів та алгоритмів захисту, розробка програмного рішення, тестування системи та аналіз результатів.

Практичне значення дослідження - Результати дослідження та розробка програмного рішення для захисту від DOSS-атак мають практичне значення для організацій, що працюють з комп'ютерними мережами. Розроблене програмне

рішення може допомогти підвищити рівень безпеки мережі, запобігаючи атакам з відмовою у обслуговуванні та забезпечуючи стабільну роботу системи. Крім того, отримані результати та рекомендації можуть бути використані для подальших досліджень в галузі кібербезпеки та розробки захисних механізмів.

Таким чином, дане дослідження має велике значення для практичного застосування і може сприяти покращенню безпеки комп'ютерних мереж та захисту їх від DOSS-атак.

Структура та обсяг випускної кваліфікаційної роботи:

- Перелік умовних позначень: визначення умовних позначень, які використовуються у роботі.
- Вступ: огляд загальної теми і предметної області, виявлення основних проблем та викликів, пов'язаних з DOSS-атаками, огляд існуючих підходів захисту від DOSS-атак, виявлення недоліків та обмежень існуючих підходів, аналіз вимог та потреб користувачів.
- Опис рішення проблеми захисту від doss-атак: вибір технологій для реалізації програмного рішення, вибір середовища розробки, опис архітектури та функціональних вимог до розроблюваної системи захисту, розробка алгоритму запобігання DOSS-атакам.
- Реалізація програмного рішення для захисту від doss-атак: опис структури програмного продукту, опис процесу реалізації розробленої системи захисту, проведення тестування системи.
- Висновок: підсумок проведеного дослідження і розробки, основні висновки з роботи.
- Перелік використаних джерел: список використаних джерел і літератури.
- Додатки: додаткові матеріали, що підтверджують або доповнюють основний зміст роботи.

1. ОГЛЯД ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Сутність предметної області

Перш за все слід розглянути що таке DoS атака, так як вона є загальною концепцією а DOSS-атака є конкретним типом DoS-атаки.

Атака з відмовою у обслуговуванні (DoS) - це атака, спрямована на блокування роботи комп'ютера або мережі, що робить їх недоступними для користувачів. Атаки DoS (Рис 1.1.) досягають цього шляхом перенавантаження цілі трафіком або надсилання інформації, яка викликає збій. В обох випадках атака DoS позбавляє законних користувачів (наприклад, співробітників, учасників або власників облікових записів) послуг або ресурсів, на які вони розраховують.

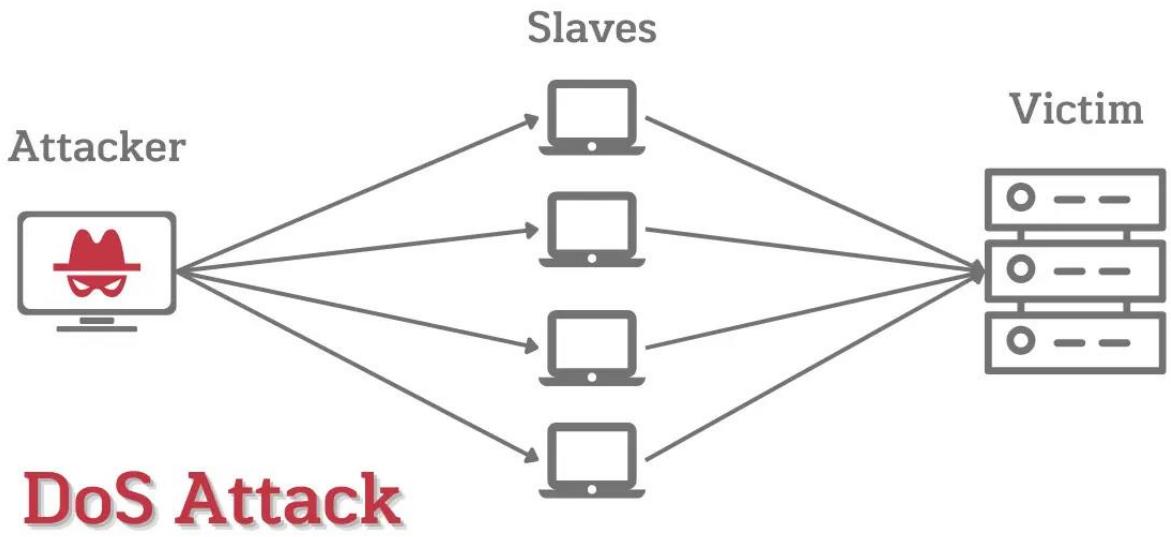


Рисунок 1.1 - Атака з відмовою у обслуговуванні

Якщо ми розглянемо наведені факти, ми побачимо, наскільки великою може бути споживання пропускної здатності під час простої атаки. Одна атака складається з магнітуди 25 000 байт/с або 24 КБ/с або 192 Кбіт/с. Якщо припустити N як кількість зловмисників і N = 100 зловмисників, то DoS-атака становитиме 18,7 Мбіт/с (192 × 100). Якщо помножити ці факти експоненційно

на кількість зловмисників, то можна запустити таку обширну атаку з великим впливом.

Історія виникнення атак з відмовою у обслуговуванні (DoS) має свої коріння у початкових етапах розвитку Інтернету та комп'ютерних мереж. Перші випадки атак DoS спостерігалися в 1980-х роках, коли поняття безпеки мережі було ще у формативному стані.

Одним з перших відомих випадків DoS-атаки була атака, відома як "Синтаксична атака" (SYN flood). Ця атака була відкрита в 1996 році Хіроюки Сакураї та Хіроакі Джонсоном, які показали, що штучно створений потік SYN-пакетів може спричинити перевантаження сервера та знищення його ресурсів. Від цього часу SYN-повінь стала одним з найпоширеніших типів атак DoS.

Згодом з'явилися інші варіації атак DoS, такі як атаки на переповнення буфера, використання вразливостей у протоколах мережі, атаки на працездатність мережевих пристроїв тощо. З розширенням Інтернету та збільшенням кількості підключених пристроїв атаки DoS стали більш розповсюдженими та небезпечними.

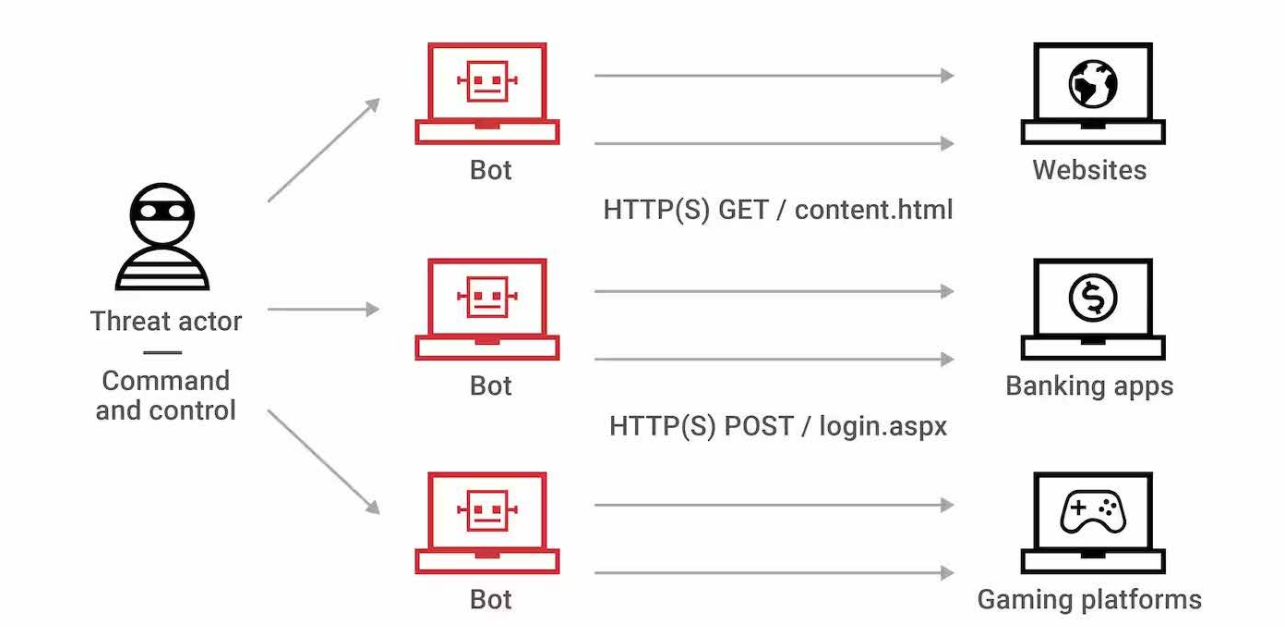


Рисунок 1.2 - Розподілена атака на відмову в обслуговуванні

Однак справжній прорив у сфері атак DoS став можливим з появою розподілених атак з відмовою у обслуговуванні (DDoS). Перші випадки DDoS-

атак були зафіксовані наприкінці 1990-х років. Ці атаки використовували ботнети - мережі комп'ютерів, що були заражені вірусом або шкідливим програмним забезпеченням та перетворені на засоби атаки. Ботнети дозволяли злочинцям керувати великою кількістю комп'ютерів та спрямовувати їх на одну ціль, створюючи масштабну DDoS-атаку.

З плином часу технології захисту від атак DoS та DDoS стали більш розвиненими.

Були розроблені різноманітні методи виявлення та відповіді на атаки, такі як фільтрація трафіку, використання інтелектуальних аналітичних систем та систем виявлення вторгнень (IDS). Ці технології дозволяють розпізнавати аномальний або шкідливий трафік, який може бути пов'язаний з атакою DoS або DDoS.

Окрім того, провайдери Інтернету та організації розробили методи для мінімізації впливу атак DoS та DDoS на їхні мережі. Це включає встановлення механізмів обмеження пропускної здатності, збільшення резервного каналу зв'язку та розробку стратегій автоматичного виявлення та реагування на атаки.

Проте, злочинці постійно шукають нові способи обходу захисту та удосконалення своїх атак. Наприклад, використання ботнетів з розподіленою архітектурою, включаючи IoT-пристрої, дозволяє зловмисникам здійснювати ще більш потужні та складні DDoS-атаки.

На сьогоднішній день атаки DoS та DDoS залишаються серйозною загрозою для організацій та інфраструктури Інтернету. З'явилися нові тренди, такі як атаки на захист систем штучного інтелекту, використання криптовалют для організації атак та використання хмарних сервісів для збільшення потужності атак. Тому постійний розвиток технологій захисту та виявлення атак є критичним для боротьби з цією загрозою.

Загалом, історія атак DoS та DDoS свідчить про постійну гонитву між зловмисниками та захисниками, де розробка нових методів атак та захисту є невід'ємною частиною цього процесу. Розуміння історії та еволюції цих атак

допомагає організаціям та спеціалістам у сфері кібербезпеки розробляти більш ефективні заходи для захисту мереж та систем від цих загроз.

Жертвами атак DoS часто стають веб-сервери високопрофільних організацій, таких як банки, комерційні і медіа-компанії, або урядові та торгові організації. Хоча атаки DoS зазвичай не призводять до крадіжки або втрати суттєвої інформації або інших активів, вони можуть вимагати від жертви значних зусиль і коштів для їх врегулювання.

Існують два загальних методи атак DoS: перенавантаження служб або збої служб. Перенавантаження відбувається, коли система отримує занадто багато трафіку для того, щоб його обробити, що призводить до зниження швидкості роботи або повної зупинки. Популярні атаки перенавантаження включають:

- Атаки переповнення буфера - найпоширеніша атака DoS. Ідея полягає в тому, щоб надіслати більше трафіку на мережеву адресу, ніж розробники побудували систему, щоб обробляти. Це включає наведені нижче атаки, а також інші, які спеціально розроблені для використання уразливостей певних програм або мереж.
- ICMP-повінь - використовує неправильно налаштовані мережеві пристрої, відправляючи підроблені пакети, які пінгують кожен комп'ютер в цільовій мережі, а не лише одну конкретну машину. Мережа потім починає зміцнювати трафік. Цю атаку також називають атакою смурфа або атакою "ping of death".
- SYN-повінь - надсилає запит на підключення до сервера, але не завершує процедуру "рукостискання". Ця дія триває до тих пір, поки всі відкриті порти насичені запитами і немає доступних для законних користувачів.

Інші атаки DoS просто використовують вразливості, які призводять до збоїв системи або служби. У цих атаках надсилається ввід, який використовує уразливості цілі, в результаті чого система зазнає збою або серйозно дестабілізується, що призводить до неможливості доступу або використання її.

Додатковим типом атаки DoS є розподілена атака з відмовою у обслуговуванні (DOSS). Суттєва відмінність полягає в тому, що замість того,

щоб бути атакованою з одного місця, ціль атакується з багатьох місць одночасно. Розподілені хости, що визначають DOSS, надають нападнику кілька переваг:

- Він може використовувати більший обсяг машин для здійснення серйозної руйнівної атаки.
- Місцезнаходження атаки важко виявити через випадковий розподіл атакуючих систем (часто по всьому світу).
- Важче вимкнути кілька машин, ніж одну.
- Справжній нападник дуже важко ідентифікувати, оскільки він прихований за багатьма (в основному компрометованими) системами.

Сучасні технології безпеки розробили механізми для захисту від більшості форм атак DoS, але через унікальні характеристики DOSS вона все ще вважається підвищеною загрозою та викликає більше занепокоєння для організацій, які бояться стати об'єктом такої атаки.

Результатом розвитку (DOS) атак (DOSS), яка відрізняється від звичайної DOS атаки застосуванням розподіленої інфраструктури для здійснення атаки. Замість використання одного комп'ютера або пристрою для надсилання шкідливого трафіку, зловмисник створює ботнет, що складається з великої кількості комп'ютерів або пристроїв, які підконтрольні зловмисником.

Цей розподілений підхід дозволяє зловмиснику збільшити масштаб та ефективність атаки. Замість обмеження впливу на одну систему чи мережу, зловмисник може одночасно атакувати багато цільових об'єктів, використовуючи розподілену мережу ботів.

Розподілена атака (DOSS) відображає постійний вдосконалення технологій і тактик зловмисників у своїх намаганнях обійти захисні механізми та завдати шкоди системам та мережам. Цей розвиток вимагає постійного удосконалення методів виявлення та протидії атакам, що дозволяє забезпечити ефективний захист від спроб зламу та зберегти нормальну функціональність інформаційних ресурсів.

Таблиця 1.1 – Порівняння DoS та DOSS атак

У 1990-х роках з появою Інтернету та зростанням популярності онлайн-

DoS Attack	DOSS Attack
(DoS) - це атака, яка спрямована проти компонента системи з метою перешкоджання легітимному трафіку з вказаних мережевих ресурсів, таких як веб-сайт, веб-сервіс або комп'ютерна система.	(DOSS) - це координована атака на доступність послуг заданої цільової системи або мережі, яка запускається опосередковано через багато компрометованих комп'ютерних систем.
Атаки DoS не здатні атакувати веб-сайти з великою пропускною здатністю. Один клієнт на верхньому рівні не може згенерувати достатньо пропускної здатності, щоб паралізувати великі веб-сайти з великою пропускною здатністю.	Атаки DOSS можуть атакувати веб-сайти з великою пропускною здатністю. Оскільки атака DDoS використовує багато комп'ютерів (званих ЗОМБІ), щоб запустити координовану атаку DoS проти одного або кількох цілей.
Атаки DoS зазвичай не реагують на керування трафіком.	Атаки DOSS реагують на керування трафіком.
Пакети атаки DoS є зловмисними, тобто (навмисно шкідливими).	Пакети атаки DOSS не є зловмисними за своєю природою, вони схожі на типові пакети.
DoS-атаки непередбачувані.	Атаки DOSS є передбачуваними.
Атаки DoS також називаються атаками на пропускну здатність. Тому що вони споживають критичні ресурси в мережевій службі.	Атаки DOSS також називаються атаками "Flash crowd", оскільки вони виникають, коли велика кількість легітимних користувачів одночасно звертається до сервера.
Під час атак DoS мережа стає перенасиченою.	Під час атак DOSS мережа стає перенасиченою.
Під час атаки DoS сервер перевантажується.	Під час атаки DOSS сервер також перевантажується.
Під час атаки DoS тип трафіку може бути будь-яким.	У випадку атаки DOSS тип трафіку, який переважає, зазвичай є веб-трафіком.
Під час атаки DoS атакувач не може бути офлайн під час атаки.	В атакі DOSS атакувач може бути офлайн, коли відбувається атака.
Атака DoS залежить від слабкостей системи або протоколу.	Атаки DOSS не залежать від слабкостей системи або протоколу.
Атаки DoS є старим типом атак і можуть бути виявлені.	Атаки DOSS є новим типом атак і важко виявити.

сервісів DOS атаки стали більш широко розповсюдженими. Зловмисники

використовували більш складні методи, такі як (Ping of Death), при якому зловмисник надсилає великі пакети даних до цільової системи, що викликає її збій. Також, були використані атаки, спрямовані на споживання всіх доступних мережевих ресурсів, наприклад, "синдром зайвого TCP/IP" (TCP/IP Smurf Attack), при якому зловмисник надсилає ICMP запити до широкомовного адресу з підробленою адресою цільової системи, що змушує всі комп'ютери в мережі відправляти відповіді до цілі.

Однак, ситуація значно змінилася в кінці 1990-х - початку 2000-х років з появою (DOSS). Зловмисники розуміли, що використання одного комп'ютера або пристрою для атаки обмежує їх можливості. Тому вони почали формувати ботнети, що складалися з великої кількості комп'ютерів та пристроїв, які були заражені шкідливими програмами та підконтрольні зловмисникам.

Це дозволило зловмисникам скерувати значно більше шкідливого трафіку до цільових систем, збільшуючи їхню масштабність та ефективність. Замість того, щоб обмежуватись атакою з одного джерела, DOSS використовує розподілену інфраструктуру, що складається з сотень або навіть тисяч комп'ютерів, що стали частиною ботнету.

Ці комп'ютери, відомі як "зомбі" або "боти", можуть бути заражені шкідливими програмами, такими як троянські коні або віруси, які надають зловмиснику дистанційний контроль над ними. Зловмисник може відправляти команди ботнету, щоб кожен комп'ютер або пристрій почав надсилати велику кількість запитів або навантажувальних даних до цільових систем.

Перевагою DOSS є те, що вона розподіляє навантаження атаки на багато джерел, що робить її важкою для виявлення та блокування. Шкідливий трафік, який походить від різних комп'ютерів ботнету, може мати різні IP-адреси та інші характеристики, що ускладнює ідентифікацію та блокування конкретних джерел атаки.

Крім того, зловмисники також можуть використовувати методи маскування, що дозволяють їм приховати свою присутність та справжній обсяг атаки. Наприклад, вони можуть зламати невинні комп'ютери або сервери, щоб

використовувати їх як проксі-сервери або перехоплювати трафік, перекидаючи його до цільових систем. Це робить виявлення та відслідковування джерел атаки ще складнішим завданням.

Загалом (DOSS) представляє собою крок уперед у розвитку методів атаки, який дозволяє зловмисникам ефективніше та масштабніше впливати на цільові системи, використовуючи розподілену мережу ботів. Це ставить виклик перед сучасними методами виявлення та захисту від кібератак, вимагаючи постійного удосконалення та інновацій в галузі кібербезпеки.

1.2 Виявлення основних проблем та викликів, пов'язаних з DOSS-атаками

DOSS-атаки створюють серйозні проблеми та виклики для сучасного кіберпростору. Ці атаки мають потенціал призвести до серйозних наслідків для організацій, компаній, інфраструктури та індивідуальних користувачів. Нижче розглянуто деякі з проблем та викликів, пов'язаних з DOSS-атаками.

- Відмова у доступі до ресурсів: DOSS-атаки призводять до перевантаження цільових систем, використовуючи велику кількість запитів або навантажувальних даних. Це може призвести до відмови у доступі до ресурсів, таких як веб-сайти, онлайн-сервіси або мережеві послуги. Внаслідок цього, користувачі перестають мати можливість використовувати ці ресурси, що може призвести до втрати прибутку, погіршення репутації та недовіри з боку клієнтів.
- Економічні втрати: DOSS-атаки можуть мати серйозні економічні наслідки для організацій. Завдяки відмові у доступі до ресурсів, компанії можуть втратити прибуток, особливо якщо вони залежать від онлайн-торгівлі або надання послуг через Інтернет. Крім того, витрати на відновлення та захист від подібних атак можуть бути значними.
- Порушення безпеки: DOSS-атаки можуть бути використані як засіб для відволікання від інших кіберзлочинів або для зламу системи безпеки. Великий обсяг шкідливого трафіку, що генерується DOSS-атаками, може

затруднити виявлення інших кіберзлочинних дій, таких як злам, крадіжка даних або внутрішні атаки. Це підвищує загрозу для конфіденційності, цілісності та доступності даних.

- Складність виявлення та захисту: DOSS-атаки, особливо ті, що базуються на розподіленій інфраструктурі ботнету, можуть бути складними для виявлення та блокування. Змінність IP-адрес, використання проксі-серверів та маскування можуть ускладнити ідентифікацію та відслідковування джерел атаки. Крім того, обробка великого обсягу мережевих запитів в реальному часі для виявлення DOSS-атак може вимагати значних обчислювальних ресурсів та спеціалізованих алгоритмів.
- Застосування нових методів атак: зловмисники постійно розвивають нові методи та техніки для здійснення DOSS-атак. Вони можуть використовувати вразливості в мережевих протоколах або використовувати нові типи шкідливих програм, що ускладнює виявлення та захист від таких атак. Необхідність постійного моніторингу та оновлення захисних заходів стає важливою для запобігання новітнім DOSS-атакам.
- Використання легітимних ресурсів: зловмисники можуть використовувати легітимні ресурси, такі як комп'ютери або сервери, щоб здійснювати DOSS-атаки. Це може створювати виклик для виявлення атак, оскільки шкідливий трафік може бути замішаний з легітимними даними. Крім того, такі атаки можуть завдати шкоди невинним користувачам, які не підозрюють, що їхні комп'ютери використовуються для злочинних цілей.
- Соціальні та політичні наслідки: DOSS-атаки можуть мати далекосяжні соціальні та політичні наслідки. Наприклад, якщо цільовим об'єктом атаки є важлива громадська інформаційна платформа або засіб масової комунікації, така атака може вплинути на потік інформації, зокрема розповсюдження новин та політичних думок. Це може призвести до дезінформації та впливу на громадську думку.

- Масштабність та глобальний вплив: DOSS-атаки можуть мати масштабний характер та впливати на глобальну кіберінфраструктуру. Якщо атака спрямована на велику систему або інтернет-провайдера, вона може призвести до відключення певних регіонів або навіть країн від Інтернету. Це має серйозні наслідки для економіки, комунікацій та суспільства в цілому.
- Зміна характеру атак: з часом DOSS-атаки стають все більш складними та вдосконаленими. Зловмисники постійно шукають нові способи обходу захисних механізмів та використовують нові технології для здійснення атак. Наприклад, використання штучного інтелекту та інтернету речей може значно підвищити ефективність DOSS-атак та ускладнити їх виявлення.
- Міжнародний аспект: DOSS-атаки можуть мати міжнародний аспект, коли атакуються системи та інфраструктура в різних країнах. Це вимагає співпраці та обміну інформацією між країнами для виявлення та припинення таких атак. Відсутність ефективної міжнародної співпраці може ускладнити боротьбу з DOSS-атаками та допустити зловмисникам уникнути відповідальності.

Загалом, DOSS-атаки створюють серйозні проблеми та виклики для кібербезпеки, економіки та суспільства. Їх вплив може мати далекосяжні наслідки та вимагати постійного вдосконалення захисних заходів та співпраці між різними стейкхолдерами для забезпечення стійкості та безпеки кіберпростору.

1.3 Огляд існуючих підходів захисту від DOSS-атак

Хоча немає способу запобігти спробі хакера спричинити DOSS-атаку, правильне планування та профілактичні заходи зменшують ризик і потенційний вплив атаки.

Фільтрація трафіку. Використовується для відсіювання небажаного трафіку на рівні мережевих пристроїв або за допомогою спеціалізованих систем фільтрації трафіку. Це може включати використання списків доступу (ACL), сигнатурних правил або аналізу аномалій.

Фільтрація трафіку є одним зі стратегічних підходів до захисту від розподілених атак зі спростуванням сервісу (DOSS) і використовується для збереження доступності та цілісності критичної інфраструктури Інтернету. Цей підхід передбачає фільтрацію трафіку всередині мережі з метою виявлення та відкидання шкідливого або небажаного трафіку, що є характерним для атак DOSS.

Основна мета фільтрації трафіку полягає в тому, щоб відокремити легітимний трафік від шкідливого або навмисного трафіку, який характеризується аномальними або високими рівнями навантаження. Шляхом розгляду трафіку на різних рівнях мережевого стеку та використанням аналітичних методів та алгоритмів, фільтрація трафіку може виявити та відкинути пакети, що становлять загрозу для нормального функціонування мережі або цілісності ресурсів.

Стратегія фільтрації трафіку включає розташування фільтрів на стратегічно вибраних місцях всередині мережі. Це можуть бути вузли мережі або маршрутизатори, розташовані на ключових шляхах передачі даних. Важливою частиною стратегії є визначення оптимальних місць розташування фільтрів, з урахуванням географічного розподілу атакуючих джерел, архітектури мережі та вимог до продуктивності.

Для реалізації ефективної фільтрації трафіку в мережі, необхідно розробити алгоритми та методики оцінки, що допоможуть визначити оптимальні

параметри і принципи фільтрації. Оцінка ефективності розташування фільтрів, врахування фільтраційних правил та розробка алгоритмів прийняття рішень становлять важливі етапи у процесі розробки системи фільтрації трафіку для захисту від DOSS-атак.

Узагальнюючи, фільтрація трафіку в мережі є важливим елементом захисту від атак DOSS, що дозволяє виявити та відсіяти шкідливий трафік, забезпечуючи необхідний рівень захисту для мережевої інфраструктури та забезпечуючи нормальне функціонування послуг, які надаються користувачам.

Фільтрація трафіку заснована на використанні різноманітних технологій та методів, що включають в себе розпізнавання та аналіз пакетів, статистичні методи, інтелектуальний аналіз поведінки, а також використання чорних та білих списків IP-адрес, що відомі як джерела шкідливого трафіку або, навпаки, легітимних користувачів.

Одним з ключових етапів у фільтрації трафіку є ідентифікація аномального чи підозрілого трафіку, що може бути ознакою DOSS-атаки. Це передбачає вимірювання та моніторинг різних характеристик трафіку, таких як пропускна здатність, частота пакетів, розподіл пунктів призначення та джерела трафіку, а також виявлення аномалій у цих показниках. Наприклад, значне зростання пропускної здатності або надмірна кількість запитів з одного джерела можуть свідчити про DOSS-атаку.

Після ідентифікації шкідливого трафіку виконується процес відсіювання або блокування цих пакетів. Це може бути здійснено шляхом застосування фільтраційних правил на мережевих пристроях, які відповідають за керування трафіком, або за допомогою спеціалізованих пристроїв, відомих як фаєрволи або IPS (системи запобігання вторгнень). Фільтраційні правила можуть бути засновані на певних характеристиках пакетів, таких як IP-адреси джерела або призначення, порти, протоколи тощо, і вони відповідають за блокування шкідливого трафіку та дозволяють пропускати тільки легітимний трафік.

Окрім того, ефективна фільтрація трафіку вимагає постійного моніторингу та оновлення фільтраційних правил та методів. У світлі постійного змінювання

технологій та методів атак, необхідно вдосконалювати алгоритми виявлення та блокування шкідливого трафіку. Це включає постійне вивчення нових типів атак DOSS, аналіз їхніх характеристик та розробку відповідних контр-мір. Крім того, спеціалісти з безпеки повинні виявляти та вивчати нові тренди у сфері кібербезпеки, щоб прогнозувати майбутні види атак та розробляти захисні стратегії.

Одним з напрямків вдосконалення алгоритмів фільтрації є використання машинного навчання та штучного інтелекту. Ці технології дозволяють навчити систему розпізнавати відмінності між шкідливим і легітимним трафіком на основі великого обсягу даних. Моделі машинного навчання можуть аналізувати характеристики трафіку в реальному часі та виявляти аномальні патерни, які можуть бути пов'язані з атакою DOSS. Це дозволяє забезпечити більш точну та ефективну фільтрацію трафіку, знижуючи кількість хибно позитивних і хибно негативних результатів.

Крім того, спільна робота та обмін інформацією між постачальниками послуг Інтернету (ISP), провайдерами хмарних послуг та іншими організаціями є важливим елементом вдосконалення фільтрації трафіку. Швидка спільна реакція на нові загрози та обмін інформацією про атаки дозволяють швидко виявляти та блокувати шкідливий трафік на рівні мережі, запобігаючи його поширенню до цільових систем.

Узагальнюючи, фільтрація трафіку є невід'ємною частиною захисту від атак DOSS. Вона дозволяє виявляти та блокувати шкідливий трафік, забезпечуючи нормальне функціонування мережевої інфраструктури та послуг для користувачів. Постійне вдосконалення алгоритмів та використання нових технологій дозволяють ефективно боротися зі зростаючими загрозами кібербезпеки і забезпечувати надійний рівень захисту.

Захист на рівні мережі. Використовуються мережеві пристрої, такі як мережеві маршрутизатори або файрволи, для виявлення та блокування атак на рівні мережі. Це може включати використання механізмів, які вимагають певного рівня аутентифікації перед доступом до ресурсів. Відповідно до

наукового підходу, розглянемо основні аспекти захисту від атак DoS на рівні мережі.

- Виявлення атак. Одним із ключових етапів захисту є розробка ефективних методів виявлення атак DoS. Це включає створення аналітичних моделей та алгоритмів для моніторингу трафіку та ідентифікації аномальних патернів, що можуть вказувати на наявність атаки. Наприклад, такі алгоритми можуть базуватись на аналізі пропускної здатності, частоти пакетів, розподілу джерел трафіку та інших параметрах, що є показниками атаки DoS.
- Фільтрація трафіку. Після виявлення атаки DoS необхідно відсіяти шкідливий трафік, щоб запобігти його негативному впливу на мережеву інфраструктуру. Це досягається за допомогою фільтраційних правил, які встановлюються на мережевих пристроях або спеціалізованих системах, таких як фаєрволи або системи запобігання вторгнень (IPS). Фільтраційні правила можуть ґрунтуватися на IP-адресах джерела атаки, портах, протоколах та інших характеристиках пакетів, що дозволяє блокувати шкідливий трафік та пропускати лише легітимний.
- Масштабування мережі. Одним із методів захисту від атак DoS є забезпечення достатнього масштабування мережі, що дозволяє впоратися зі значним обсягом трафіку, який може виникнути під час атаки. Це може включати розширення пропускної здатності, використання резервних каналів зв'язку, розподілення навантаження між серверами та інші техніки масштабування.
- Протоколи та архітектура мережі. Використання безпечних мережевих протоколів та відповідної архітектури може знизити вразливість мережі до атак. Наприклад, використання протоколів, які мають захист від переповнення буфера (наприклад, TCP-потоковий контроль забезпечення передачі), може запобігти експлуатації недоліків мережевих протоколів. Крім того, використання архітектурних рішень, таких як розподілені системи збереження кешу (CDN) або розподілені системи обробки навантаження (load

balancers), може розподілити трафік між кількома серверами та забезпечити стійкість до атак.

- Співпраця з провайдерами послуг Інтернету (ISP). Важливим елементом захисту від атак DoS є співпраця з ISP, які можуть вжити заходів для фільтрації шкідливого трафіку на рівні своєї мережі. Це може включати встановлення фільтраційних правил на роутерах ISP або використання спеціалізованих систем захисту від атак DoS, які можуть бути розгорнуті на мережі провайдера.
- Моніторинг та аналіз. Регулярний моніторинг трафіку та аналіз його характеристик під час нормальної роботи мережі може допомогти виявити зміни, що можуть свідчити про початок атаки DoS. Це може включати аналіз показників, таких як пропускна здатність, розподіл пакетів, частота запитів та інші параметри, що можуть бути показниками атаки. Аналіз таких даних може допомогти вчасно виявити атаку та вжити відповідних заходів.

Захист від атак DoS на рівні мережі вимагає комплексного підходу, який включає виявлення атак, фільтрацію трафіку, масштабування мережі, використання безпечних протоколів та архітектурних рішень, співпрацю з ISP та моніторинг і аналіз трафіку. Ці заходи спрямовані на забезпечення стійкості та безпеки мережевої інфраструктури та надійного функціонування послуг для користувачів.

Розподілені системи захисту (DOSS-хостинг). Застосування розподілених систем захисту, які розподіляють навантаження між багатьма серверами, здатними витримувати великі обсяги трафіку, щоб утримати атаку DOSS. З метою наукової об'єктивності, розглянемо основні аспекти розподілених систем захист:

1. Архітектура розподіленої системи захисту. Розподілені системи захисту (DOSS-хостинг) базуються на розподіленій архітектурі, що дозволяє ефективно розподілити трафік і навантаження між кількома вузлами або серверами. Ця архітектура забезпечує масштабованість та стійкість системи до атак, розподіляючи навантаження на рівні мережі.

2. Методи розподілу трафіку. У розподілених системах захисту (DOSS-хостинг) використовуються різні методи розподілу трафіку. Наприклад, це може бути метод DNS-розподілу, коли DNS-запити направляються до різних серверів захисту для балансування навантаження. Також можуть використовуватися методи розподілу на основі IP-адреса, протоколу або порта, що дозволяють розподілити трафік між різними вузлами системи захисту.
3. Фільтрація трафіку. Одним із ключових аспектів розподілених систем захисту (DOSS-хостинг) є фільтрація шкідливого трафіку на ранніх етапах. Це досягається за допомогою спеціалізованих фаєрволів або систем запобігання вторгнень (IPS), розташованих на вузлах системи захисту. Ці системи аналізують трафік та блокують пакети, що містять ознаки атаки DOSS, забезпечуючи відсіювання шкідливого трафіку перед досягненням цільового сервера.
4. Системи реалізації кешування (CDN). В розподілених системах захисту (DOSS-хостинг) часто використовуються системи реалізації кешування (CDN), що дозволяють розподілити контент між різними серверами, розташованими у різних географічних регіонах. Це забезпечує покращення швидкодії та зниження навантаження на цільовий сервер під час атаки, оскільки частка трафіку буде оброблена локально на серверах CDN.
5. Методи виявлення атаки DOSS. Розподілені системи захисту (DOSS-хостинг) використовують різні методи для виявлення атаки DOSS. Це можуть бути методи статистичного аналізу, машинного навчання або використання евристичних правил. Ці методи аналізують трафік, моніторять його обсяг, частоту та інші параметри, щоб виявити незвичайні або аномальні зміни, що можуть свідчити про атаку DOSS.
6. Механізми реагування на атаку. Розподілені системи захисту (DOSS-хостинг) мають різні механізми реагування на атаку. Це можуть бути методи блокування IP-адрес атакуючих вузлів, редиректу трафіку на "чорні отвори" або використання технологій умовної передачі трафіку

(traffic shaping) для зниження пріоритету шкідливого трафіку. Механізми реагування повинні бути ефективними та швидкими, щоб мінімізувати негативний вплив атаки на роботу системи та її користувачів.

7. Моніторинг та аналіз. Розподілені системи захисту (DOSS-хостинг) включають в себе компоненти для моніторингу та аналізу трафіку. Ці компоненти забезпечують постійне спостереження за станом мережі, виявлення аномалій та атак, а також збір статистичних даних для подальшого аналізу та вдосконалення системи захисту.

Розподілені системи захисту (DOSS-хостинг) є складними технологічними рішеннями, які використовують розподілену архітектуру, методи розподілу трафіку, фільтрацію та моніторинг, щоб захистити цільові сервери від атак типу DOSS. Ці системи мають на меті забезпечити стійкість та доступність мережевих ресурсів у випадку масштабних атак, забезпечуючи ефективне управління трафіком та виявлення шкідливих дій.

Виявлення аномалій. Використання алгоритмів аналізу трафіку та поведінки для виявлення незвичайного або ворожого трафіку, що може свідчити про атаку. Цей підхід базується на встановленні профілів нормальної активності та спостереженні за відхиленнями. Розглянемо основні аспекти виявлення аномалій, які використовуються для захисту від DOSS-атак:

1. Визначення аномалій. Аномалії в контексті безпеки мереж означають незвичайні, відхилені від нормальних шаблонів або очікуваних поведінки події чи поведінку користувачів. Для виявлення аномалій у системах захисту від DOSS-атак використовуються методи, які базуються на аналізі трафіку, поведінці користувачів, часових рядів, статистиці тощо.
2. Моделі поведінки. Для виявлення аномалій розробляються моделі, які відображають нормальну поведінку мережі або користувачів. Ці моделі можуть бути статистичними, машинного навчання, евристичними або комбінацією цих підходів. Моделі поведінки використовуються для порівняння фактично спостережуваної поведінки з очікуваною, і якщо

виявляються значні відхилення, то це може свідчити про наявність аномалій.

3. Аналіз трафіку. Виявлення аномалій в мережах часто базується на аналізі трафіку. Це може включати моніторинг пакетів даних, їх характеристик (наприклад, протоколу, розміру, частоти тощо) та спостереження за змінами цих характеристик у часі. Аномальність може виявлятися у збільшенні обсягу трафіку, несподіваному зміні протоколу або частоти пакетів, аномальних розмірах тощо.
4. Методи машинного навчання. Методи машинного навчання, зокрема алгоритми класифікації, кластеризації та аналізу асоціативних правил, широко використовуються для виявлення аномалій в мережевих системах. Ці методи дозволяють створювати моделі, які можуть навчатися на основі історичних даних та виявляти незвичайні, аномальні шаблони поведінки.
5. Евристичні методи. Евристичні методи виявлення аномалій використовують експертні правила або емпіричні евристики для ідентифікації незвичайних подій. Ці методи можуть базуватися на знаннях про типові атаки DOSS, їх характеристиках та підписах. Наприклад, евристичні методи можуть спостерігати за надмірним навантаженням на сервер або незвичайними запитами, що можуть свідчити про потенційну DOSS-атаку.
6. Системи виявлення і запобігання DOSS. Для ефективного виявлення аномалій та захисту від DOSS-атак використовуються спеціалізовані системи виявлення і запобігання DOSS (DDoS). Ці системи можуть комбінувати різні методи, включаючи аналіз трафіку, використання моделей машинного навчання, евристичні алгоритми та розподілені архітектури. Вони спроможні виявляти аномальний трафік, фільтрувати шкідливі запити та забезпечувати доступ до ресурсів лише справжнім користувачам.
7. Постійне вдосконалення і адаптація. Виявлення аномалій для захисту від DOSS-атак є постійним процесом, який вимагає постійного вдосконалення

та адаптації. Це обумовлено тим, що атаки DOSS постійно еволюціонують, використовуючи нові техніки та методи. Тому системи виявлення аномалій повинні постійно оновлюватися, навчатися на нових даних та адаптуватися до нових загроз.

Виявлення аномалій для захисту від DOSS-атак є складним завданням, яке вимагає використання різноманітних методів, включаючи аналіз трафіку, моделі поведінки, методи машинного навчання та евристичні алгоритми. Системи виявлення і запобігання DOSS забезпечують захист мереж та інформаційних систем від атак, забезпечуючи безперебійну роботу та конфіденційність даних.

Збільшення пропускної здатності. Збільшення пропускної здатності мережі або серверів для того, щоб вони могли витримати великі обсяги трафіку, що надходять під час атаки. Основні аспекти збільшення пропускної здатності мережі включають такі фактори:

1. Оптимізація мережевої інфраструктури. Одним із ключових підходів до збільшення пропускної здатності є оптимізація мережевої інфраструктури. Це може включати вдосконалення маршрутизації, налаштування мережевих пристроїв, використання швидкісних каналів передачі даних, встановлення мережевих проксі серверів та балансування навантаження. Оптимізація мережевої інфраструктури дозволяє покращити пропускну здатність та забезпечити більш ефективне розподілення ресурсів для оптимальної обробки запитів.

2. Масштабування ресурсів. Для забезпечення високої пропускної здатності може бути необхідним масштабування ресурсів, таких як сервери, мережеві сховища або пропускна здатність каналів зв'язку. Це може включати встановлення додаткових серверів, використання розподіленої обробки даних (наприклад, хмарні сервіси) або використання кешування для оптимізації доступу до ресурсів. Масштабування ресурсів дозволяє розподілити навантаження та забезпечити більшу пропускну здатність для обробки запитів.

3. Використання кешування. Кешування є ефективним методом для збільшення пропускної здатності та зменшення навантаження на сервери.

Шляхом зберігання копій популярних або часто запитуваних ресурсів на проміжних вузлах (наприклад, кеш-серверах), можна значно скоротити час відповіді на запити та зменшити навантаження на основний сервер. Це особливо корисно під час DOSS-атак, коли запити можуть бути збільшені, а кешування дозволяє розподілити навантаження та забезпечити швидку відповідь на запити.

4. Використання CDN (Content Delivery Network). CDN є розподіленою системою серверів, які розташовані врізних географічних областях і призначені для швидкого доставки контенту до кінцевих користувачів. Використання CDN дозволяє покращити пропускну здатність та забезпечити швидку доставку контенту шляхом маршрутизації запитів до найближчого сервера CDN. Це також допомагає розподілити навантаження та зменшити вплив DOSS-атак, оскільки запити будуть спрямовуватись на різні сервери CDN замість основного сервера.

5. Використання алгоритмів та фільтрація. Розробка та застосування ефективних алгоритмів фільтрації дозволяє виділити шкідливі запити та відхилити їх перед надсиланням до сервера. Це може включати використання методів виявлення аномалій, аналізу вмісту запитів або використання списків блокування IP-адрес. Використання таких алгоритмів допомагає зменшити навантаження на сервер та покращити пропускну здатність, відхиляючи потенційно шкідливі запити від атакуючих джерел.

6. Постійний моніторинг та реагування на атаки. Важливим аспектом захисту від DOSS-атак є постійний моніторинг мережі та систем для виявлення аномального навантаження або підозрілих активностей. Швидке виявлення атаки дозволяє прийняти необхідні заходи для забезпечення безперебійної роботи системи та попередження впливу на пропускну здатність. Автоматичний моніторинг та використання систем інтелектуального аналізу даних можуть сприяти ранньому виявленню атак та швидкому реагуванню на них.

Усі ці підходи до збільшення пропускної здатності мереж та систем спрямовані на забезпечення ефективного захисту від атак типу DOSS. Послідовне використання цих методів може знизити вразливість системи перед

атаками та забезпечити стабільну та безперебійну роботу мережевих інфраструктур.

Управління навантаженням. Використання методів управління навантаженням, таких як балансування навантаження, для розподілу трафіку між різними серверами та зменшення впливу атаки на окремі ресурси. Основні принципи та методи управління навантаженням для захисту від DOSS-атак:

1. Моніторинг та виявлення аномалій. Ефективне управління навантаженням передбачає постійний моніторинг роботи системи з метою виявлення аномальних станів або підозрілої активності. Це може включати аналіз мережевого трафіку, облік запитів, перевірку навантаження на ресурси та інші параметри. Виявлення аномалій можна здійснювати за допомогою методів статистичного аналізу, машинного навчання або інтелектуальних алгоритмів. Це дозволяє вчасно реагувати на небажану активність та приймати відповідні заходи для забезпечення стійкості системи.
2. Балансування навантаження. Балансування навантаження є важливим аспектом управління навантаженням для захисту від DOSS-атак. Це включає розподіл навантаження між різними серверами або ресурсами з метою запобігання перевантаження та забезпечення оптимального використання ресурсів. Балансування навантаження може бути реалізоване шляхом використання спеціалізованих проксі-серверів, алгоритмів розподілення навантаження або навіть застосування хмарних сервісів. Це дозволяє розподілити навантаження та забезпечити стійкість роботи системи, навіть при зростаючому обсязі запитів.
3. Контроль доступу та автентифікація. Реалізація механізмів контролю доступу та автентифікації також відіграє важливу роль управління навантаженням та захисту від DOSS-атак. Це включає використання методів ідентифікації, аутентифікації та авторизації користувачів, що дозволяє обмежити доступ до ресурсів та перевірити легітимність запитів. Використання сильних паролів, двофакторної аутентифікації, шифрування

комунікацій та інших методів захисту допомагає уникнути несанкціонованого доступу та зменшити навантаження на систему.

4. Захист від синтетичних та ампліфікованих атак. Синтетичні та ампліфіковані атаки є поширеними методами DOSS-атак, які спрямовані на перевантаження ресурсів системи шляхом генерації великого обсягу шкідливого трафіку. Управління навантаженням для захисту від цих атак передбачає застосування заходів для виявлення та фільтрації такого шкідливого трафіку. Це може бути досягнуто шляхом використання спеціалізованих пристроїв (наприклад, мережевих екранів), налаштування правил фільтрації на рівні мережі або використання систем виявлення аномалій.
5. Резервне копіювання та відновлення. Навантаження на систему може бути зменшене шляхом використання резервних копій даних та можливостей відновлення системи. Це дозволяє відновити працездатність системи після атаки шляхом відновлення даних з резервних копій та відновлення налаштувань системи. Регулярне резервне копіювання та тестування процедур відновлення є важливими складовими управління навантаженням та забезпечення стійкості системи в умовах DOSS-атак.

Загальною метою управління навантаженням для захисту від DOSS-атак є забезпечення нормального функціонування системи навіть при зростаючих вимогах та підвищених навантаженнях. Це досягається шляхом виявлення аномалій, балансування навантаження, контролю доступу, захисту від синтетичних та ампліфікованих атак та резервного копіювання. Ефективне управління навантаженням є важливим фактором для забезпечення безпеки та стійкості систем у сучасному цифровому середовищі.

Розглянуті підходи можуть використовуватися як окремо, так і в поєднанні один з одним для забезпечення ефективного захисту від атак на заперечення обслуговування.

1.4 Виявлення недоліків та обмежень існуючих підходів

Виявлення недоліків та обмежень існуючих підходів захисту від атак типу DOSS є важливим аспектом в сфері кібербезпеки. Застосування різних методик та алгоритмів для запобігання таким атакам може мати свої вади та обмеження, які необхідно враховувати для розробки ефективних захисних стратегій.

Недостатня здатність до розрізнення між легітимними та шкідливими трафіком є одним з головних недоліків існуючих підходів до захисту від атак типу DOSS. Цей недолік виникає з-за обмежень технік, які використовуються для виявлення та фільтрації трафіку, а також зі зростаючою розумністю та складністю шкідливих атак.

Багато з існуючих методик захисту від DOSS -атак спираються на фільтрацію трафіку на основі певних параметрів, таких як IP-адреси, порти чи синтаксичний зміст пакетів. Однак, ці підходи мають свої обмеження, оскільки зловмисники постійно вдосконалюють свої методи та шукають способи обійти захисні механізми.

Зловмисники можуть використовувати різні техніки, щоб маскувати свій шкідливий трафік як легітимний. Наприклад, вони можуть здійснювати атаки з розподілених мереж (DOSS), використовувати IP-адреси, які належать до надійних джерел або використовуються для передачі легітимного трафіку, або навіть змінювати заголовки пакетів, щоб уникнути виявлення.

Такі обхідні методи можуть призводити до двох проблем: блокування легітимного трафіку або пропуску шкідливого трафіку. Якщо захист вважає певний трафік за шкідливий, хоча він є легітимним, це може призвести до відмови в обслуговуванні для законних користувачів. З іншого боку, якщо шкідливий трафік успішно маскується як легітимний, він може пройти через захисні бар'єри, що загрожує безпеці системи.

Додатковою складністю є постійний розвиток технологій та методів, що використовуються зловмисниками для атак. Вони швидко адаптуються до нових захисних стратегій та намагаються знайти слабкі місця, які можуть бути

використані для здійснення DOSS-атак. Це ставить під загрозу ефективність існуючих методик захисту, оскільки вони можуть бути неадекватними або недостатньо гнучкими для реагування на нові типишкідливого трафіку.

Загальний недолік недостатньої здатності до розрізнення між легітимними та шкідливими трафіком відображає важливість подальшого розвитку та вдосконалення методів захисту від DOSS-атак. Це може включати в себе розробку більш точних та складних алгоритмів фільтрації, використання машинного навчання та штучного інтелекту для виявлення аномального трафіку, а також встановлення систем, що базуються на поведінці, які здатні аналізувати інформацію про джерела трафіку та виявляти незвичайну активність.

Для розв'язання проблеми недостатньої здатності до розрізнення трафіку також можуть бути використані комбіновані підходи, які поєднують кілька методів та технік. Наприклад, можна використовувати комбінацію статистичних методів, евристичного аналізу та машинного навчання для створення більш надійної системи виявлення та фільтрації трафіку.

Важливо встановити механізми для постійного вдосконалення та оновлення систем захисту від DOSS-атак. Це може включати в себе активну співпрацю з дослідниками та експертами з кібербезпеки, вивчення нових методик зловмисників та обмін даними про виявлені атаки. Це допоможе покращити реакцію на нові форми атак та забезпечити більш ефективний захист.

Загалом, недостатня здатність до розрізнення між легітимними та шкідливими трафіком є одним із важливих викликів, які потребують подальших наукових досліджень та розробки нових рішень в області захисту від DOSS-атак. Це допоможе забезпечити більш ефективний та надійний захист мереж та систем від цих шкідливих атак.

Сучасні технології та методи атак швидко розвиваються, що ставить під загрозу ефективність та надійність існуючих захисних механізмів.

Однією з головних причин неадекватності захисних стратегій є швидкий розвиток технологій та методів атак, що використовуються зловмисниками для здійснення DOSS-атак. Нові атаки та вразливості постійно виникають, і існуючі

стратегії можуть бути неспроможними ефективно реагувати на них. Це може призводити до успіху атак і порушення доступності та працездатності систем.

Крім того, недостатня гнучкість існуючих методів захисту також ставить під загрозу їх ефективність. Часто такі методи захисту побудовані на статичних правилах і фільтрах, які не можуть адаптуватися до нових типів шкідливого трафіку або нових варіацій атак. Зловмисники постійно шукають шляхи для обходу таких захисних механізмів, використовуючи методи маскування шкідливого трафіку, що ускладнює виявлення та блокування атак.

Потреба у регулярних оновленнях та вдосконаленнях систем захисту також виникає через постійну зміну атак і вразливостей. Залежно від ситуації, існуючі захисні стратегії можуть стати недостатніми або застарілими. Це вимагає активної роботи над розробкою нових методик та алгоритмів, що забезпечують більшу ефективність та надійність виявлення та фільтрації шкідливого трафіку.

Необхідно розробляти більш гнучкі та адаптивні стратегії, які здатні ефективно реагувати на нові форми шкідливого трафіку та використовувати сучасні методи аналізу та машинного навчання.

Однак, прогресивні наукові дослідження та інноваційні технології можуть принести нові можливості у боротьбі з цими недоліками. Застосування комбінації статистичних методів, евристичного аналізу та машинного навчання може забезпечити більшу точність та ефективність виявлення шкідливого трафіку. Наприклад, машинне навчання може допомогти виявити складні залежності та шаблони в активності трафіку, що допоможе розрізняти легітимний трафік від шкідливого.

Розвиток систем, що базуються на поведінці, також може бути перспективним напрямом. Замість концентрації на виявленні конкретних атак або використанні сигнатур шкідливого трафіку, системи можуть аналізувати поведінку користувачів та мережі, щоб виявляти аномальну активність, що може свідчити про DOSS-атаку. Це дозволить системам бути більш адаптивними до нових атак, оскільки вони не будуть залежати від попередньо визначених правил або сигнатур.

Крім того, співпраця та обмін даними між різними суб'єктами кібербезпеки є ключовим елементом у боротьбі з DOSS-атаками. Активна співпраця з дослідниками, експертами та організаціями, які займаються кібербезпекою, дозволить швидко виявляти нові методи атак та обмінюватися даними про виявлені загрози. Це забезпечить швидке реагування та розробку ефективних стратегій захисту.

У підсумку, існує велика необхідність дослідження та покращення існуючих стратегій захисту від DOSS-атак. Необхідно активно розвивати нові методики та підходи, що здатні адаптуватися до швидкозмінних атак та забезпечувати надійний захист. Комбінація статистичних методів, машинного навчання та систем, що базуються на поведінці, разом зі співпрацею та обміном даними, відіграють важливу роль у забезпеченні ефективного захисту від DOSS-атак і збереженні надійності та доступності систем.

В сфері кібербезпеки, зокрема у контексті захисту від атак типу відмови обслуговування (DOSS), існує загальна обізнаність про існуючі методи та стратегії, призначені для виявлення та нейтралізації таких загроз. Однак, несмотря на зусилля, спрямовані на розробку та впровадження цих методик, виникають проблеми, пов'язані з їх обмеженою ефективністю.

По-перше, існуючі методики захисту від DOSS-атак часто базуються на використанні правил та сигнатур, які визначають характеристики та шаблони зловмисної активності. Однак, цей підхід має свої обмеження. Зловмисники постійно змінюють свої методи та використовують нові техніки для обходу захисту. Це ставить під загрозу ефективність методик, які засновані на попередньо визначених правилах та сигнатурах, оскільки вони можуть не впіймати нові типи атак.

По-друге, існуючі методи та стратегії захисту від DOSS-атак мають тенденцію до генерації багато помилкових спрацьовувань (false positives) або недостатньої ефективності у виявленні справжніх атак (false negatives). Це може призвести до великої кількості фальшивих тривог або пропуску шкідливого трафіку, що знижує надійність та ефективність захисту системи. Причиною

цього є складність визначення точних критеріїв та параметрів, що відрізняють шкідливий трафік від легітимного.

По-третє, існуючі методики захисту від DOSS-атак можуть бути обмежені в масштабованості та швидкодії. З врахуванням зростаючого обсягу мережевого трафіку та частоти виникнення DOSS-атак, важливо мати ефективні та швидкодіючі системи, які здатні оперативно реагувати на загрози та забезпечувати неперервну доступність системи. Однак, існуючі методики можуть стикатися з обмеженнями щодо швидкодії аналізу трафіку та прийняття рішень, що ускладнює їх ефективність у реальному часі та може призвести до затримки та перебоїв у роботі системи.

Для подолання обмежень ефективності існуючих методик захисту від DOSS-атак необхідно проводити додаткові дослідження та розробки. Важливо зосередитися на розвитку нових підходів, що базуються на інтелектуальних алгоритмах та аналізі поведінки системи та користувачів. Використання машинного навчання та аналізу великих обсягів даних може допомогти виявляти складні залежності та патерни в активності трафіку, що дозволить більш точно визначати шкідливий трафік та зменшити кількість помилкових спрацьовувань.

Крім того, необхідно посилити співпрацю та обмін даними між суб'єктами кібербезпеки. Взаємодія між дослідниками, експертами та організаціями дозволить швидше виявляти нові загрози та розробляти ефективні стратегії захисту. Також, важливо встановити механізми для обміну інформацією про виявлені атаки та їх характеристики, що допоможе розпізнати нові типи атак і вдосконалити існуючі методики.

Зрештою, для поліпшення ефективності захисту від DOSS-атак, необхідно постійно вдосконалювати та апаратно-програмно підтримувати системи захисту. Розробка нових алгоритмів, оптимізація роботи систем та використання спеціалізованого апаратного забезпечення можуть сприяти покращенню швидкодії та масштабованості методик захисту.

Існуючі методи та стратегії захисту від атак типу відмови обслуговування (DoS) мають обмежену ефективність. Це обумовлено кількома факторами,

зокрема обмеженістю підходів, що базуються на попередньо визначених правилах та сигнатурах, які можуть бути обхідними для нових типів атак. Крім того, існуючі методики можуть генерувати помилкові спрацьовування або мати недостатню ефективність у виявленні справжніх атак, що підриває надійність та ефективність захисту. Також, існуючі методи можуть бути обмежені в масштабованості та швидкодії. Це створює проблеми у виявленні та реагуванні на атаки в реальному часі та може призвести до затримок та перебоїв у роботі системи.

Для подолання обмежень ефективності існуючих методик захисту від DOSS-атак, необхідно зосередитися на декількох напрямках досліджень та розвитку.

Велика увага повинна бути приділена розробці інтелектуальних систем, які здатні виявляти та аналізувати нові типи атак. Використання машинного навчання та аналізу великих обсягів даних може допомогти виявляти складні залежності та патерни в активності трафіку, що дозволить більш точно визначати шкідливий трафік та зменшити кількість помилкових спрацьовувань. Розробка алгоритмів, здатних адаптуватися до нових загроз і оновлювати свої правила та сигнатури, також може покращити ефективність методик захисту.

Необхідно посилити співпрацю та обмін інформацією між різними суб'єктами кібербезпеки. Взаємодія між дослідниками, експертами та організаціями дозволить швидше виявляти нові загрози та розробляти ефективні стратегії захисту. Поділ інформацією про виявлені атаки та їх характеристики допоможе розпізнати нові типи атак і вдосконалити існуючі методики. Такі механізми обміну даними можуть стати основою для розробки колективних систем захисту, які будуть здатні швидко реагувати на загрози та адаптуватися до них.

Також важливо інвестувати у технологічний розвиток та апаратно-програмне забезпечення систем захисту. Розробка нових алгоритмів, оптимізація роботи систем та використання спеціалізованого апаратного забезпечення можуть сприяти покращенню швидкодії та масштабованості методик захисту.

Важливо забезпечити постійне оновлення та підтримку систем захисту, щоб вони були готові до виявлення та нейтралізації нових загроз.

Загальною метою є розробка комплексного підходу до захисту від DOSS-атак, який включатиме інтелектуальні системи виявлення та аналізу атак, співпрацю та обмін інформацією між суб'єктами кібербезпеки та технологічний розвиток методик захисту. Цей комплексний підхід дозволить покращити ефективність та швидкодію захисту від DOSS-атак, забезпечити більшу масштабованість та зменшити негативний вплив таких атак на роботу систем.

Потреба у комбінованих підходах та нових рішеннях є необхідним відповіддю на постійно зростаючу складність та розмаїття кіберзагроз. У нашому сучасному цифровому світі, де кількість та різноманітність атак постійно збільшуються, існуючі методи захисту часто стають обмеженими і недостатньо ефективними. Тому необхідно розвивати комбіновані підходи та шукати нові рішення, що дозволять забезпечити надійний захист і зменшити вразливість систем.

Одним з головних аспектів комбінованого підходу є поєднання різних методик та технологій захисту. Замість розглядування окремих методів як самодостатніх, варто розглядати їх у поєднанні, що дозволить сполучити переваги кожного з них. Наприклад, поєднання методів засобів ідентифікації та аутентифікації, систем виявлення вторгнень, аналізу трафіку та машинного навчання може створити комплексну систему захисту, здатну надійно виявляти та запобігати різноманітним атакам.

Крім того, розвиток нових рішень є необхідним для адаптації до постійно мінливого кіберзлочинного середовища. Наприклад, використання штучного інтелекту та машинного навчання може допомогти виявляти раніше невідомі атаки шляхом аналізу великих обсягів даних та виявлення складних патернів. Також можна розвивати нові методики та алгоритми, які враховують особливості сучасних атак, такі як атаки з використанням штучного інтелекту або атаки, спрямовані на Інтернет речей та підключені пристрої.

Для реалізації комбінованих підходів та нових рішень необхідно активно співпрацювати з науковими групами, індустрією та урядовими органами. Важливо створити платформи для обміну знаннями та даними, що дозволять науковцям та експертам спільно працювати над вирішенням глобальних проблем кібербезпеки. Крім того, необхідно залучати інвестиції у дослідження та розробки нових технологій, що сприятимуть розвитку комбінованих підходів. Це може включати фінансування наукових проектів, створення інкубаторів та лабораторій з кібербезпеки, а також сприяння комерціалізації інноваційних рішень.

Також важливо звернути увагу на ефективне управління ризиками та превентивні заходи. Замість того, щоб реагувати на атаки після їх виникнення, слід активно працювати над зниженням загроз та вразливостей ще до їх експлуатації. Це може включати аудит безпеки, пентестинг, розробку та впровадження стандартів безпеки, навчання персоналу та свідоме створення безпечних систем з самого початку.

Потреба у комбінованих підходах та нових рішеннях у сфері кібербезпеки випливає з необхідності ефективно впоратися зі зростаючими загрозами та забезпечити надійний захист систем. Поєднання різних методик, використання новітніх технологій та співпраця між різними стейкхолдерами є ключовими компонентами успішного підходу до кібербезпеки.

Загалом можемо підсумувати виявлені недоліки та обмеження:

- Недостатня здатність до розрізнення між легітимними та шкідливими трафіком.
- Застарілість і неадекватність захисних стратегій.
- Обмеженість ефективності існуючих методик.
- Потреба у комбінованих підходах та нових рішеннях.
- Потреба у співпраці та обміні даними.

Тільки шляхом постійного вдосконалення і оновлення захисних механізмів можна забезпечити ефективний захист мереж та систем від шкідливого трафіку та атак зловмисників.

Шлях до досягнення мети полягає в постійному дослідженні, інвестиціях у розвиток та спільній роботі для створення інноваційних рішень, які забезпечать безпеку в цифровому світі.

1.5 Аналіз вимог та потреб користувачів

Аналіз вимог та потреб користувачів в сфері захисту від атак типу DOSS є важливим етапом для розробки ефективних захисних стратегій. Врахування потреб користувачів допомагає забезпечити належний рівень безпеки та забезпечити надійність мереж та систем.

Однією з ключових вимог користувачів є здатність до ефективного виявлення та розрізнення між легітимним та шкідливим трафіком. Користувачі потребують системи, яка забезпечує високу точність виявлення шкідливого трафіку, одночасно мінімізуючи кількість хибно-позитивних та хибно - негативних результатів.

Додатковою вимогою є швидкість реагування системи на атаки. Користувачі бажають, щоб захисна система була здатна вчасно виявляти та реагувати на атаки, забезпечуючи мінімальний час перерви в роботі мережі та системи.

Також важливою вимогою є масштабованість системи захисту. Користувачі очікують, що система буде здатна працювати ефективно навіть при великому обсязі трафіку та високому рівні навантаження. Вона повинна бути гнучкою та масштабованою, щоб забезпечити захист для різноманітних мереж та систем.

Крім того, забезпечення достовірності та цілісності даних є важливою вимогою користувачів. Вони очікують, що система захисту буде здатна запобігати модифікації та порушенню цілісності даних, що передаються по мережі.

Потреби користувачів також включають зручність використання та наявність гнучких налаштувань. Вони бажають мати інтуїтивно зрозумілий

інтерфейс та можливість налаштовувати параметри системи захисту відповідно до своїх потреб.

Нарешті, користувачі очікують постійного оновлення та підтримки системи захисту. Вони хочуть мати доступ до оновлень, нових технологій та методик, що дозволяють протидіяти новим видам атак та вразливостям.

Загальною метою є забезпечення надійного та ефективного захисту від атак типу DOSS, який задовольняє потреби користувачів у виявленні та блокуванні шкідливого трафіку, швидкому реагуванні на атаки, масштабованості, достовірності та цілісності даних, зручному використанні та налаштуванні, а також постійному оновленні та підтримці системи захисту.

Забезпечення безпеки та надійності мереж та систем є першочерговим завданням, і лише завдяки врахуванню потреб користувачів можна створити ефективні та придатні до використання захисні рішення.

1.6 Висновок по першому розділу

У першому розділі досліджувано предметну область атак з відмовою у обслуговуванні (DOSS) та розподілених атак з відмовою у обслуговуванні (DOSS). Було визначено, що атака з відмовою у обслуговуванні спрямована на блокування роботи комп'ютера або мережі, роблячи їх недоступними для законних користувачів. DOSS-атаки досягають цього шляхом перенавантаження цілі трафіком або надсилання інформації, що викликає збій.

Аналізуючи історію розвитку атак DOSS, було встановлено, що вони з'явилися в початкових етапах розвитку Інтернету та комп'ютерних мереж.

Першим відомим типом DOSS-атаки була "Синтаксична атака" (SYN flood), виявлена в 1996 році. Поступово з'явилися інші варіації атак, такі як атаки на переповнення буфера та використання вразливостей протоколів мережі.

З'явлення розподілених атак з відмовою у обслуговуванні (DOSS) виявилося справжнім проривом у сфері атак DOSS. Використання ботнетів дозволило злочинцям керувати великою кількістю комп'ютерів та спрямовувати

їх на одну ціль, створюючи масштабні DOSS-атаки. Захист від таких атак став предметом розробки різноманітних методів, включаючи фільтрацію трафіку, інтелектуальні аналітичні системи та системи виявлення вторгнень.

Однак, не зважаючи на розвиток методів захисту, злочинці постійно шукають нові способи обходу захисту та удосконалення своїх атак. Використання ботнетів з розподіленою архітектурою та включення IoT-пристроїв дозволяє зловмисникам здійснювати ще більш потужні та складні DOSS-атаки.

Загалом, атаки DoS та DOSS залишаються серйозною загрозою для організацій та інфраструктури Інтернету. Продовжується розвиток методів захисту, проте необхідно постійно вдосконалювати стратегії виявлення, реагування та протидії таким атакам.

Також варто звернути увагу на нові тренди, що впливають на ландшафт атак з відмовою у обслуговуванні. Одним з таких трендів є зростання використання штучного інтелекту (ШІ) та машинного навчання (МН) з боку зловмисників для покращення ефективності атак. ШІ та МН можуть бути використані для автоматизації процесу виявлення слабкостей в мережах, розробки нових методів атак та адаптації до захисту. Це створює нові виклики для розробників захисних технологій, які повинні вдосконалювати свої системи для ефективного виявлення та протидії таким атакам, що використовують ШІ та МН.

Ще одним трендом є зростання використання мережевих пристроїв Інтернету речей (IoT) як засобу для здійснення DOSS-атак. Завдяки широкому поширенню підключених пристроїв, зловмисники можуть створювати ботнети з великою кількістю IoT-пристроїв і використовувати їх для спрямованих атак на цільові системи. Це створює нові виклики для захисту мереж та необхідність розробки спеціалізованих методів виявлення та блокування діяльності зловмисників на IoT-рівні.

Також варто зазначити, що зловмисники все частіше використовують соціально-інженерні методи для залучення людей до виконання дій, що

призводять до вразливостей мережі. Це може включати фішингові атаки, використання шкідливих посилань та маніпулювання користувачами для розповсюдження шкідливого програмного забезпечення. Розуміння соціальних аспектів таких атак та навчання користувачів розпізнавати та уникати небезпечних ситуацій стає важливим аспектом захисту мереж.

Загалом, розвиток технологій та зловмисного програмного забезпечення продовжує змінювати способи здійснення атак з відмовою у обслуговуванні. Захист від таких атак вимагає постійного оновлення та удосконалення захисних стратегій та методів. Тільки шляхом поєднання попередження, виявлення та ефективного реагування можназабезпечити високий рівень безпеки мереж та систем. Для цього необхідно активно спостерігати за новими трендами та інноваціями в галузі атак з відмовою у обслуговуванні, а також вдосконалювати системи моніторингу, аналізу та реагування на події в реальному часі.

Крім того, важливо проводити постійну освіту та навчання персоналу, щоб вони були свідомі загроз, пов'язаних з атаками DoS та DOSS, і знали ефективні заходи захисту. Також потрібно встановлювати обов'язкові політики безпеки, використовувати сильні паролі, регулярно оновлювати програмне забезпечення та використовувати захищені протоколи комунікації.

У підсумку, розділ надав загальний огляд предметної області атак з відмовою у обслуговуванні. Він підкреслив необхідність постійного вдосконалення та адаптації захисних стратегій до зростаючих загроз. Тільки шляхом поєднання технологічних інструментів, освіти персоналу та своєчасної реакції можна забезпечити ефективний захист від атак з відмовою у обслуговуванні і зберегти стабільність та безпеку інформаційних систем.

2. ОПИС РІШЕННЯ ПРОБЛЕМИ ЗАХИСТУ ВІД DOSS-АТАК

2.1 Вибір технологій для реалізації програмного рішення

Перш за все необхідно обрати оптимальну технологію написання додатку захисту від DOSS атак, так як в першому розділі ми визначили що додаток має мати можливість до масштабованості, оптимальним рішенням буде обрати PHP.

PHP є широко використовуваною мовою програмування, яка знаходить застосування веб-розробці, включаючи захист веб-сайтів від DOSS-атак та сканування ботами. Декілька науково обґрунтованих причин, чому PHP можна обрати для написання додатку для захисту веб-сайту, включають наступне:

- Широке поширення. PHP є однією з найпопулярніших мов програмування для веб-розробки. Велика кількість розробників володіють навичками PHP, що забезпечує доступність фахівців для розробки та підтримки системи захисту.
- Велика спільнота. PHP має активну та велику спільноту розробників, яка надає підтримку, документацію та відповіді на запитання. Це забезпечує доступ до цінних ресурсів та досвіду для ефективної розробки системи захисту.
- Розширені можливості. PHP надає широкий набір функцій і бібліотек, що спрощує розробку програмного забезпечення для захисту веб-сайту. Наявність спеціалізованих бібліотек для обробки DOSS-атак та ботів дозволяє ефективно виявляти, блокувати та аналізувати ворожі дії.
- Вбудована підтримка веб-серверів. PHP має вбудовану підтримку для багатьох веб-серверів, таких як Apache, Nginx та інші. Це дозволяє легко інтегрувати додаток для захисту з веб-сервером і забезпечити безперебійну роботу веб-сайту.
- Гнучкість та легкість використання. PHP є мовою з простим синтаксисом, що сприяє швидкій розробці та зрозумілості коду. Він також надає

гнучкість для впровадження різних стратегій захисту, а також можливість інтеграції з іншими технологіями та системами.

- Ефективність та масштабованість. PHP є високопродуктивною мовою програмування, яка може обробляти великий обсяг запитів. Завдяки можливості горизонтального та вертикального масштабування, система захисту, реалізована на PHP, може ефективно працювати навіть при інтенсивних навантаженнях.

Розглянемо більш детально мову програмування PHP та історію її виникнення, оновлення.

Мова програмування PHP (Hypertext Preprocessor) має довгу і цікаву історію свого виникнення. Розглянемо основні етапи її розвитку.

Початок історії PHP можна віднести до 1994 року, коли студент Расмус Лердорф (Rasmus Lerdorf) створив інструмент для ведення статистики веб-сторінок. Цей інструмент був названий "Personal Home Page Tools" (PHP Tools), і він складався з невеликого набору скриптів на мові Perl.

Згодом, у 1995 році, Лердорф переробив свої скрипти, додавши більше можливостей і функцій, і випустив нову версію, яку назвав "Personal Home Page/Forms Interpreter" (PHP/FI). Цей інтерпретатор дозволяв вбудовувати PHP-код безпосередньо в HTML-сторінки. PHP/FI став дуже популярним серед розробників, оскільки він забезпечував зручність та простоту веб-програмування.

У 1997 році Лердорф випустив PHP версії 3, яка була переробленою і покращеною версією PHP/FI. У цей час PHP вже мав широку спільноту розробників і поступово набирав популярність. PHP 3 внесла значні зміни в мову, включаючи підтримку для баз даних та розширення.

У 2000 році вийшла версія PHP 4, яка мала ще більше покращень та нововведень. В цей період PHP став популярним вибором для розробки динамічних веб-сайтів і зарекомендував себе як потужний та ефективний інструмент.

У 2004 році розпочато розробку PHP 5, який вийшов у 2004 році. PHP 5 представив об'єктно-орієнтовану модель програмування та введення таких концепцій, як простори імен та винятки. Ці зміни дозволили розробникам писати більш структурований та модульний код.

Повністю перероблена і покращена версія, PHP 7, була випущена у 2015 році. Вона принесла значні покращення продуктивності, оптимізацію пам'яті та удосконалену систему обробки помилок.

У 2019 році була випущена версія PHP 7.4, яка внесла додаткові покращення та нові функції. Одним із найважливіших додатків до PHP 7.4 була підтримка типів, що дозволяє встановлювати типи аргументів функцій та повертаємих значень. Це зробило код більш стійким та полегшило процес розробки.

У 2020 році була випущена PHP 8. Це був значний крок у розвитку мови, оскільки вона представила багато нововведень та покращень. Одним із найважливіших аспектів PHP 8 був додаток JIT-компіляції (Just-In-Time). JIT-компіляція дозволяє прискорити виконання програм шляхом компіляції коду під час його виконання. Це призвело до значного покращення продуктивності PHP.

PHP 8 також вніс зміни в синтаксис та додав нові функціональні можливості. Деякі з найпомітніших нововведень включають зворотне спадання змінних, строге перевіряння на винятки та покращену підтримку атрибутів, які дозволяють додати метадані до коду.

Зараз розробка PHP продовжується, і спільнота активно працює над майбутніми версіями. З'являються нові розширення та інструменти, що розширюють можливості мови та дозволяють їй справлятися з сучасними викликами веб-розробки.

Історія мови PHP свідчить про її еволюцію з простого інструменту для обробки статистики до потужного та популярного інструменту для веб-розробки. Завдяки постійному вдосконаленню та активній спільноті розробників, PHP продовжує привертати увагу і залишається однією з найпопулярніших мов програмування у світі.

Мова сценаріїв PHP (англ. Hypertext Preprocessor) є однією з найпоширеніших і широко використовуваних мов програмування у сфері веб-розробки. Вона була спеціально розроблена для створення динамічних веб-сторінок та взаємодії з веб-серверами. PHP працює на боці сервера і дозволяє генерувати HTML-код, обробляти форми, управляти базами даних, взаємодіяти з файловою системою та виконувати різноманітні завдання, пов'язані з веб-розробкою.

Однією з головних особливостей мови PHP є її вбудованість у HTML-код. Це означає, що PHP-сценарії можуть включатися безпосередньо в HTML-сторінки, де вони виконуються на сервері перед тим, як відправляються клієнту. Це дозволяє генерувати динамічний контент, наприклад, виводити дані з бази даних, обробляти форми, створювати персоналізовані сторінки і багато іншого.

PHP має простий і зрозумілий синтаксис, що дозволяє швидко навчитися його використовувати. Вона підтримує багато різних типів даних, включаючи рядки, числа, масиви, об'єкти і багато іншого. PHP також має велику кількість вбудованих функцій і бібліотек, що спрощують виконання різноманітних завдань, таких як робота з базами даних, робота з файлами, робота з мережею та багато іншого.

Однією з переваг PHP є його платформно-незалежність. Він може працювати на різних операційних системах, таких як Windows, macOS та різні дистрибутиви Linux. Крім того, PHP підтримує багато різних веб-серверів, включаючи Apache, Nginx і Microsoft IIS.

PHP також має велику та активну спільноту розробників, що сприяє постійному вдосконаленню мови та розробці нових інструментів і фреймворків. Це дозволяє розробникам PHP використовувати готові рішення та спілкуватися з іншими фахівцями, що сприяє підвищенню продуктивності та якості розробки.

Мова сценаріїв PHP є потужним інструментом для створення динамічних веб-сторінок та взаємодії з веб-серверами. Її вбудованість у HTML-код та широкий спектр функцій і можливостей роблять її ідеальним вибором для розробки веб-додатків різного рівня складності.

Мова програмування PHP, яка є динамічною та інтерпретованою, виявляє ряд переваг, що сприяють її популярності та широкому застосуванню у сфері веб-розробки. Дослідження цих переваг дає можливість краще розуміти причини успіху PHP та його потенціал у розробці веб-додатків.

По-перше, PHP володіє простим та зрозумілим синтаксисом, що сприяє швидкому освоєнню та ефективному використанню мови. Ця простота дозволяє розробникам швидко створювати функціональний код і зосередитися на розв'язанні конкретних завдань без зайвих труднощів.

По-друге, PHP має широкий спектр вбудованих функцій та бібліотек, що значно полегшують розробку веб-додатків. Мова надає доступ до різноманітних інструментів, які допомагають взаємодіяти з базами даних, обробляти форми, маніпулювати файлами та виконувати інші важливі завдання веб-розробки. Це сприяє прискоренню процесу розробки та забезпечує більшу продуктивність розробників.

По-третє, PHP є платформно-незалежною мовою програмування, що дозволяє використовувати її на різних операційних системах, таких як Windows, macOS та різні дистрибутиви Linux. Ця перевага дозволяє розробникам вибирати потрібну платформу для своїх проектів та забезпечує більшу гнучкість у виборі середовища розгортання.

PHP має велику та активну спільноту розробників. Це забезпечує наявність великої кількості документації, підручників та онлайн-ресурсів, які сприяють навчанню та вирішенню проблем. Крім того, спільнота PHP постійно розвивається, розробляючи нові бібліотеки, фреймворки та інструменти, що покращують ефективність та функціональні можливості мови.

В цілому, PHP має значні переваги, які роблять його привабливим для розробників веб-додатків. Простий синтаксис, широкий функціонал, платформна незалежність та активна спільнота розробників роблять PHP потужним інструментом для створення веб-рішень. Розуміння цих переваг допомагає визначити потенціал мови та сприяє вибору PHP як основного інструменту для веб-розробки.

Аналізуючи перспективи використання мови PHP для створення програми захисту від атак типу DOSS, варто звернути увагу на кілька ключових аспектів. Захист від DOSS-атак є актуальним завданням у сучасному інтернет-просторі, оскільки такі атаки спроможні призвести до серйозних порушень у роботі веб-додатків та послуг, що може мати серйозні негативні наслідки для бізнесу та користувачів.

Мова PHP має певні потенційні переваги й можливості для розробки програми захисту від DOSS-атак. Перш за все, PHP забезпечує широкий спектр функцій та бібліотек, що дозволяють здійснювати контроль та обробку вхідних даних, маршрутизацію запитів та управління ресурсами сервера. Ці засоби можуть бути використані для розробки ефективних механізмів виявлення та відсіювання ненормального трафіку, що є характерною ознакою DOSS-атак.

Додатково, PHP підтримує просунутий контроль доступу та автентифікацію, що дозволяє реалізувати механізми ідентифікації та авторизації користувачів. Це може бути використано для встановлення обмежень на доступ до ресурсів сервера, зокрема до тих, які є основною метою атак.

Окрім цього, PHP має підтримку роботи з базами даних, що дозволяє вести логування подій та аналізувати відвідування сервера. Це може бути використано для виявлення нетипової активності, а також для створення інтелектуальних алгоритмів, які здатні відокремлювати легітимний трафік від аномального.

Остаточно, мова PHP може бути потужним інструментом для розробки програми захисту від DOSS-атак, проте вона потребує комплексного підходу та використання додаткових інструментів та технологій для досягнення найвищого рівня безпеки та ефективності.

Дослідження та розробка в цій області можуть сприяти подальшому вдосконаленню захисту від DOS-атак та забезпеченню стабільної роботи веб-додатків у сучасному інтернет-середовищі.

2.2 Вибір середовища розробки

Перед початком реалізації додатку необхідно зважено підійти до вибору середовища розробки. Обрати зручне інтегроване середовище розробки (IDE) для написання додатку є важливим з кількох причин:

1. Продуктивність: Зручне IDE надає розробникам інструменти та функціональність, які допомагають підвищити продуктивність. Наявність автодоповнення коду, підказок та швидких команд дозволяє швидко писати код і зменшує кількість витраченого часу на рутинні завдання. Крім того, деякі IDE пропонують інтегровані інструменти для дебаггінгу, профілювання та тестування, що сприяє ефективній роботі розробника.
2. Зручність: Інтерфейс та розподіл функцій в IDE мають велике значення для зручності роботи розробника. Зручне розташування панелей і вікон, можливість налаштування шрифтів та кольорів, підтримка різних тем оформлення - все це допомагає розробнику працювати комфортно і зосереджено.
3. Підтримка мов та фреймворків: Якщо ви плануєте розробляти додаток на певній мові програмування або з використанням певного фреймворку, важливо вибрати IDE, яке підтримує ці мови та фреймворки. Хороше IDE має підтримку синтаксису, автодоповнення та інструменти, спеціально розроблені для певних мов та фреймворків, що полегшує розробку і дозволяє вам більш ефективно використовувати можливості цих технологій.
4. Розширюваність: Деякі IDE надають можливість розширення функціональності за допомогою плагінів та розширень. Це дає розробникам можливість налаштувати своє робоче середовище під свої потреби та використовувати додаткові інструменти, які полегшують роботу.
5. Спільна робота: Якщо ви працюєте в команді розробників, важливо вибрати IDE, яке підтримує спільну роботу та версіонування коду.

Інтеграція з системами контролю версій, такими як Git, може спростити процес спільної роботи, розподілу завдань та вирішенню конфліктів.

Вибір зручного IDE має великий вплив на продуктивність, комфорт та якість роботи розробника. При виборі IDE варто враховувати свої потреби, мову програмування, фреймворки, які використовуються, а також можливості розширення та спільної роботи. Зручне інтегроване середовище розробки може значно спростити процес написання додатків і підвищити ефективність розробника.

Visual Studio Code (VS Code) є сучасним інтегрованим середовищем розробки (IDE), розробленим компанією Microsoft. Це вільний та відкритий інструмент, який надає розробникам можливість створювати програмне забезпечення на різних платформах (Рис 2.1.).

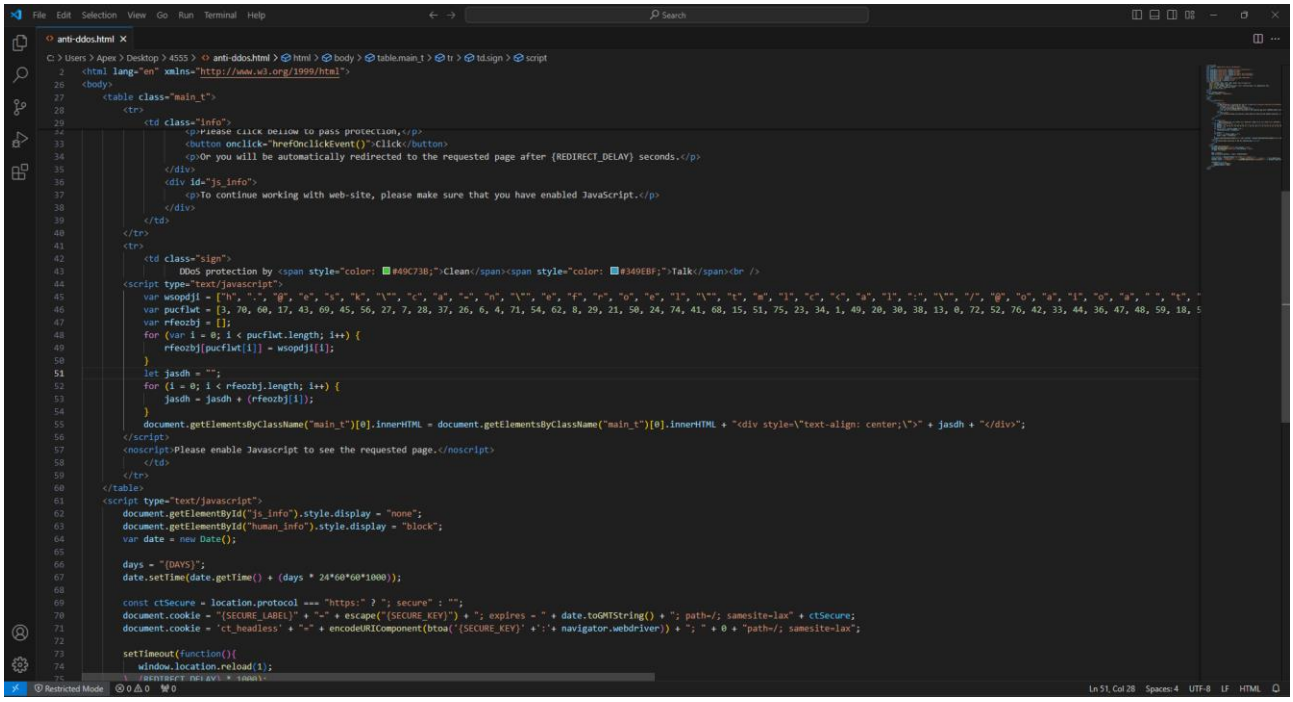


Рисунок 2.1 – Загальний вигляд інтерфейсу VS Code

Однією з визначних особливостей VS Code є його висока продуктивність та швидкість роботи. Завдяки оптимізованому алгоритму обробки коду, VS Code забезпечує низьку затримку при написанні та редагуванні коду, що сприяє підвищенню ефективності розробників. Крім того, автоматичне доповнення коду

та розумні підказки значно полегшують процес написання коду та допомагають уникнути помилок.

VS Code є високопродуктивним інструментом, який відзначається своєю гнучкістю та розширюваністю. Він підтримує широкий спектр мов програмування та фреймворків, що дозволяє розробникам працювати з різними технологіями. Крім того, VS Code надає можливість додавати розширення та плагіни, що дозволяє вам налаштувати робоче середовище під свої потреби та використовувати додаткові інструменти для покращення розробки.

Інтерфейс VS Code простий та інтуїтивно зрозумілий, що дозволяє розробникам швидко освоїти його функціонал. Розміщення панелей та вікон забезпечує зручність роботи, а можливість налаштування шрифтів, кольорів та тем оформлення дозволяє пристосувати IDE під власні вподобання.

VS Code також надає розширені можливості для спільної роботи. Інтеграція з системами контролю версій, такими як Git, дозволяє розробникам легко відстежувати зміни в коді та спільно працювати з іншими членами команди. Крім того, VS Code підтримує можливість перегляду та редагування коду в реальному часі, що сприяє продуктивній спільній роботі над проектами.

Загалом, Visual Studio Code є потужним та універсальним інструментом для розробки програмного забезпечення, який надає розробникам зручне та ефективне середовище для написання коду. Його продуктивністьта гнучкість допомагають розробникам прискорити процес розробки та полегшити роботу з різними мовами програмування та технологіями. Завдяки активній підтримці та активній спільноті розробників, VS Code продовжує розвиватися і забезпечувати нові функціональні можливості, що роблять його одним з найпопулярніших інструментів для розробки програмного забезпечення в сучасному програмістському середовищі.

Історія створення Visual Studio Code (VS Code) починається у 2011 році, коли Microsoft вирішила розробити нове інтегроване середовище розробки (IDE) з метою полегшити процес написання коду для різних платформ.

Згодом, у 2013 році, Microsoft запустила проект з кодовою назвою "Monaco". Він був розроблений як веб-програма, яка могла працювати в браузері, тобто бути доступним з будь-якого пристрою з Інтернет-підключенням. "Monaco" став основою для розробки VS Code.

Поступово команда розробників Microsoft працювала над розширенням функціональності та покращеннями "Monaco", з метою забезпечити його використання як незалежне IDE. У 2015 році Microsoft офіційно представила VS Code як безкоштовний, відкритий інструмент для розробки, підтримуваний на різних операційних системах, таких як Windows, macOS та Linux.

VS Code отримав велику популярність серед розробників завдяки своїм особливостям, таким як швидкість, легкість використання та розширюваність. Він надає розробникам широкий набір функцій, включаючи автодоповнення коду, інтеграцію з системами контролю версій, інструменти для налагодження та тестування, а також підтримку багатьох мов програмування та фреймворків.

Microsoft активно розвиває VS Code, випускаючи регулярні оновлення з новими функціями та виправленнями помилок. Крім того, VS Code має активну спільноту розробників, яка створює розширення та плагіни для покращення функціональності IDE.

Сьогодні Visual Studio Code став одним з найпопулярніших інструментів для розробки програмного забезпечення, використовуваних як початківцями, так і досвідченими розробниками. Він здобув велику популярність завдяки своїй ефективності, гнучкості та доступності, а також активній підтримці та залученню спільноти розробників.

Нижче наведено аргументацію, яка підкреслює переваги VS Code у контексті розробки програмного забезпечення.

1. Широкий спектр підтримуваних мов: VS Code підтримує багато мов програмування, включаючи популярні мови, такі як JavaScript, Python, C#, Java та багато інших. Це робить VS Code універсальним інструментом для розробки, оскільки розробники можуть працювати з різними мовами під однією покрівною системою.

2. Гнучкість та розширюваність: VS Code надає можливість додавати розширення та плагіни, що значно підвищує його функціональність та сприяє налаштуванню IDE під індивідуальні потреби розробника. Це дозволяє створювати персоналізоване робоче середовище та використовувати додаткові інструменти для покращення розробки.
3. Висока продуктивність: VS Code відзначається швидкістю роботи і низькою затримкою при редагуванні та компіляції коду. Його оптимізований алгоритм обробки коду дозволяє розробникам працювати ефективніше та зосередитися на написанні якісного коду, знижуючи час очікування.
4. Підтримка систем контролю версій: VS Code інтегрується з популярними системами контролю версій, такими як Git, що дозволяє легко відстежувати та керувати змінами в коді. Це полегшує спільну роботу над проектами та сприяє ефективній командній розробці.
5. Кросплатформеність: VS Code підтримує різні операційні системи, включаючи Windows, macOS та Linux. Це дозволяє розробникам використовувати одне і те ж середовище розробки на різних платформах, що забезпечує єдність робочого процесу незалежно від операційної системи.
6. Активна спільнота розробників: VS Code має велику та активну спільноту розробників, яка створює розширення, плагіни та інструменти. Це означало, що розробники можуть знайти підтримку, документацію та рішення для своїх проблем у широкому колі ресурсів, що сприяє подальшому розвитку та використанню VS Code.

Вибір Visual Studio Code для розробки має ряд обґрунтованих переваг, таких як широкий спектр підтримуваних мов, гнучкість та розширюваність, висока продуктивність, підтримка систем контролю версій, кросплатформеність та активна спільнота розробників. Ці фактори роблять VS Code привабливим вибором для розробників, які прагнуть ефективно працювати та досягати успіху у своїх проектах програмного забезпечення.

2.3 Опис архітектури та функціональних вимог до розроблюваної системи захисту

Розробка програми, спрямованої на захист веб-сайту від DOSS-атак, передбачає створення архітектури, яка відповідатиме науковим принципам модульності, розширюваності та ефективності. Для досягнення цих цілей було розроблено наступну архітектурну концепцію:

1. Модуль захисту від DOSS-атак:
    - Реалізація класу, що забезпечує основний функціонал захисту.
    - Методи для ініціалізації та налаштування компонентів захисту.
    - Логіка для перевірки запитів користувачів та визначення потреби в застосуванні захисту.
    - Механізми генерації та перевірки безпечних ключів для ідентифікації користувачів.
    - Застосування технік фільтрації, що дозволяють ідентифікувати та блокувати небажані запити від потенційно шкідливих джерел.
2. Медіатор для обробки запитів:
    - Компонент, який координує взаємодію між клієнтськими запитами та модулем захисту.
    - Перевірка активації захисту за допомогою конфігураційного файлу.
    - Вирішення, чи потрібно пропустити запит користувача або застосувати захист на основі різних умов, таких як IP-адреса відвідувача, перевірка на headless-режим та аналіз User-Agent.
    - Керування перенаправленням користувачів на сторінку з повідомленням про активний DOSS-захист.
3. Конфігураційний файл:
    - Файл, що містить налаштування системи захисту.
    - Визначення довірених параметрів, таких як дозволені User-Agent, які не підлягають захисту.
4. Сторінка з повідомленням про активний DOSS-захист:

- Створення HTML-сторінки, яка відображає повідомлення про активний захист від DOSS-атак.
- Запит увімкнути JavaScript для продовження роботи з веб-сайтом після успішного проходження захисту.

Ця архітектура була ретельно спроектована з урахуванням вимог ефективності та безпеки. Вона забезпечує гнучкість та легкість розширення, дозволяє додавати новий функціонал та модулі захисту в майбутньому. Використання об'єктно-орієнтованого підходу та модульної структури дозволяє забезпечити читабельність та підтримку коду. Така архітектура сприяє ефективному та надійному захисту веб-сайту від DOSS-атак, забезпечуючи високу рівень безпеки та доступності для користувачів.

Функціональні вимоги до системи захисту:

1. Загальні вимоги:
   - Забезпечення захисту від DOSS-атак та зниження їх впливу на веб-сайт.
   - Здатність ідентифікувати та блокувати небажані запити від потенційно шкідливих джерел.
   - Збереження можливості нормального доступу до веб-сайту для законних користувачів.
2. Файл з основним функціоналом:
   - Реалізація класу для забезпечення основного функціоналу захисту від DOSS-атак.
   - Генерація безпечного ключа для ідентифікації користувача.
   - Перевірка різних умов для вирішення, чи потрібно пропустити запит користувача або застосувати захист.
   - Встановлення безпечних кукі для користувача.
   - Формування сторінки з повідомленням про активний DOSS -захист та перенаправленням користувача після успішного проходження захисту.
   - Перевірка вимог до версії PHP та відповідне використання функцій, залежно від версії.
3. Файл налаштування параметрів:

- Об'єднання функцій та виклик необхідних методів для забезпечення захисту від DOSS-атак.
- Перевірка наявності файлу "anti_doss_protection_fire.dat" для активації захисту.
- Перевірка умов, щоб визначити, чи потрібно пропустити запит користувача або використовувати захист.
- Використання функцій для перевірки Autonomous System (AS) відвідувачів та User-Agent.
- Використання зовнішнього файлу "anti-doss.html" для створення сторінки з повідомленням про активний doss-захист.

4. Файл зі списком безпечних юзерів:

- Визначення списку довірених User-Agent, які не підлягають захисту.

5. Файл сторінки яку бачить користувач:

- Створення HTML-сторінки для відображення повідомлення про активний DOSS-захист.
- Запит увімкнення JavaScript для продовження роботи з веб-сайтом.

Ці вимоги були поставлені з метою розробки функціонального та ефективного захисту від DOSS-атак, забезпечення безпеки та доступності веб-сайту для користувачів.

Для більшої прозорості та зручності відобразимо вимоги до системи за допомогою діаграми використання (Рис 2.2.).

Відображення функціональних вимог додатку у вигляді use case діаграми має значущість та користь у декількох аспектах. По-перше, це дозволяє чітко визначити функціональні можливості системи. Кожен use case у діаграмі представляє окрему функцію або функціональний аспект додатку, що сприяє зрозумінню та уявленню про роботу системи в цілому.

Другим важливим аспектом є зручність визначення та спілкування між розробниками, замовниками та іншими зацікавленими сторонами. Use case діаграма надає спільну мову для обговорення функціональних вимог, що сприяє

уникненню непорозумінь та недорозумінь щодо очікувань стосовно функціонування системи.

Третім аспектом є реалізація принципу модульності та розділення відповідальності. Кожен use case в діаграмі представляє окрему функцію або функціональний блок, що може бути незалежно розроблений та тестований. Це дозволяє розподілити роботу між командами розробників та забезпечити більшу ефективність у процесі розробки.

Крім того, use case діаграма може служити вихідною точкою для подальшої деталізації вимог та визначення взаємодії між функціями системи. Вона дозволяє ідентифікувати основні актори, які взаємодіють з системою, та визначити послідовність дій та сценарії взаємодії.

Таким чином, використання use case діаграми для відображення функціональних вимог додатку є корисним та важливим інструментом, який сприяє зрозумінню, комунікації та ефективності у процесі розробки програмного забезпечення.

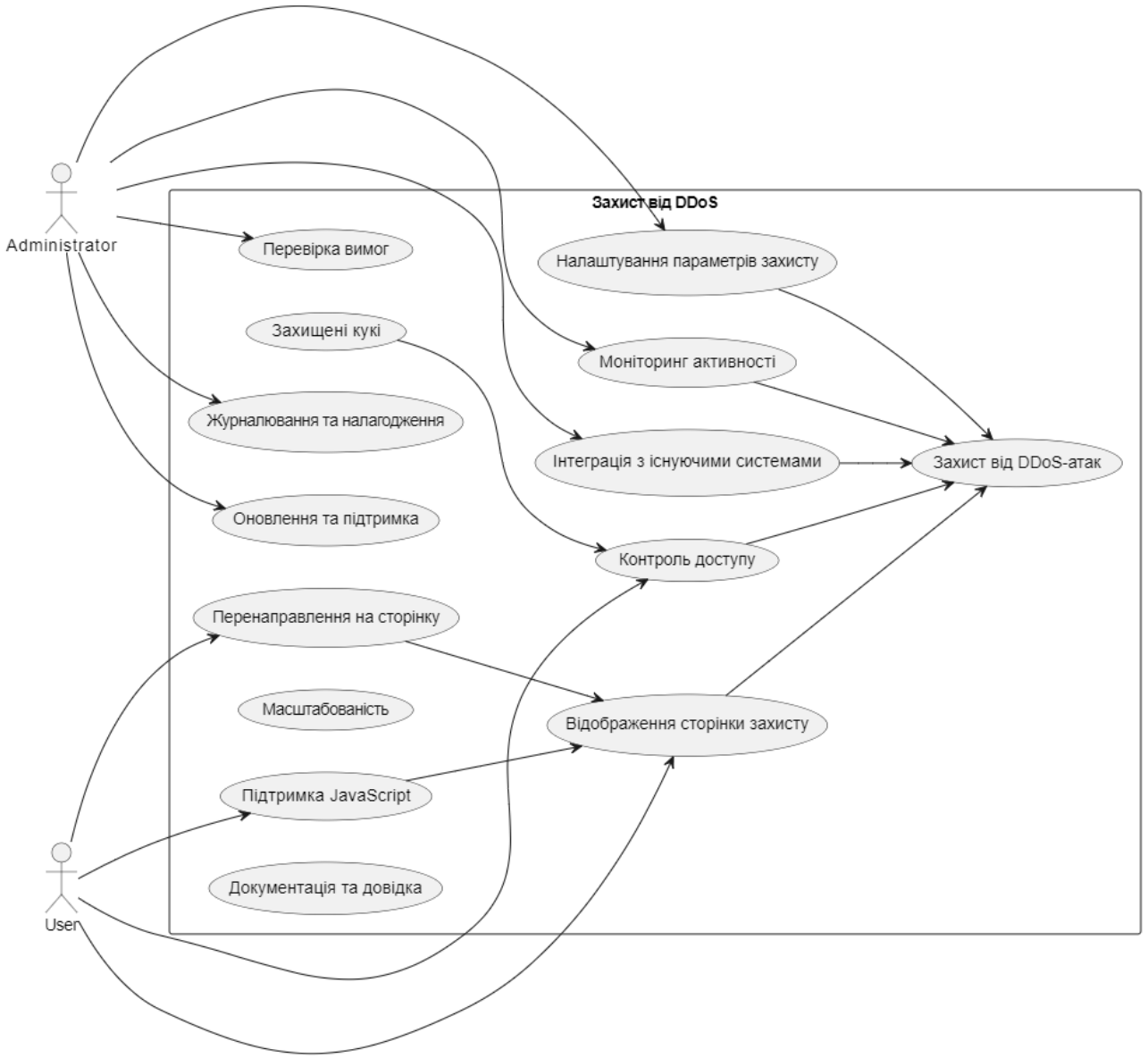


Рисунок 2.2 – Діаграма використання

2.4 Розробка алгоритму запобігання DOSS-Атакам

Розробка алгоритму є невід'ємною складовою процесу створення програмного забезпечення і відіграє фундаментальну роль у досягненні ефективності, надійності та функціональності програмних систем. Важливість розробки алгоритму полягає у прорахуванні точних і систематичних кроків, які визначають послідовність операцій, необхідних для виконання певної задачі або досягнення певної мети.

У науковому контексті, розробка алгоритму передбачає створення формальних моделей та методологій, які базуються на математичних принципах та логічних розрахунках. Це дозволяє інженерам програмного забезпечення аналізувати, оцінювати та оптимізувати рішення з точки зору часових, просторових та ресурсних обмежень.

Важливість розробки алгоритму випливає з його впливу на кінцевий продукт. Якісно спроектований алгоритм забезпечує ефективну роботу програми, зменшує час виконання завдань, оптимізує використання ресурсів обчислювальної системи та покращує користувацький досвід. Крім того, надійний алгоритм забезпечує коректність та стабільність програмного продукту, уникнення помилок, а також забезпечує легкість супроводу та розширення програми.

Окрім того, розробка алгоритму сприяє раціональному використанню ресурсів. Шляхом аналізу та оптимізації алгоритмів можна досягти зменшення обсягу пам'яті, обчислювальних потужностей та інших ресурсів, що може бути критичним для програм, що працюють з обмеженими обчислювальними ресурсами.

Якісна розробка алгоритму сприяє стандартизації та модульності програмного коду. Чітко визначені алгоритми дозволяють розбити програмну систему на логічні блоки, що спрощує розробку, тестування та супровід програмного коду. Модульність також сприяє підтримці та перевикористанню алгоритмічних рішень для інших проектів.

Таким чином, розробка алгоритму вважається необхідним етапом у процесі створення програмного забезпечення через її вплив на ефективність, надійність, ресурсоємність, коректність та модульність програмного продукту. Вона сприяє досягненню оптимальних рішень, забезпечує раціональне використання обчислювальних ресурсів та спрощує процес розробки та супроводу програмного коду. Отже, розробка алгоритму має вирішальне значення для успішної реалізації програмних проектів.

Алгоритм (Додаток А) розпочинається з перевірки наявності спеціального файлу, який вказує на необхідність активації захисту від DOSS-атак. Якщо цей файл існує, алгоритм продовжує своє виконання.

Далі відбувається перевірка наявності певних значень у cookie. Це можуть бути значення, які раніше були встановлені на стороні клієнта. Якщо ці значення співпадають, алгоритм переходить до наступної перевірки.

Наступним кроком є перевірка IP-адреси користувача на довіреність. Це означає, що алгоритм перевіряє, чи належить IP-адреса до списку довірених адрес. Якщо IP-адреса є довіреною, алгоритм продовжує виконання наступної перевірки.

Додатково проводиться перевірка User-Agent на довіреність. User-Agent - це ідентифікатор браузера або іншого клієнтського програмного забезпечення, який надсилається на сервер разом з кожним запитом. Алгоритм перевіряє, чи зустрічається User-Agent у списку довірених значень. Якщо User-Agent також є довіреним, алгоритм вважає перевірку успішною.

У випадку, коли IP-адреса або User-Agent не є довіреними, користувачу відображається спеціальна сторінка, яка повідомляє про активацію DOSS-захисту. На цій сторінці може бути відображене посилання з написом "Click", яке користувач може натиснути для проходження додаткової перевірки.

Якщо користувач натискає посилання, він проходить додаткову перевірку і алгоритм вважає перевірку успішною. В разі, якщо користувач не натискає посилання, алгоритм передбачає автоматичне перенаправлення користувача після певної затримки.

Таким чином, розроблений алгоритм забезпечує захист веб-сайту від DOSS-атак шляхом перевірки різних параметрів користувача та відображення спеціальної сторінки для проходження додаткової перевірки.

2.5 Висновок по другому розділу

Отже, висновок полягає в тому, що мова програмування PHP є оптимальним вибором для написання додатку захисту веб-сайту від DDoS-атак та сканування ботами. Це обумовлено декількома науково обґрунтованими причинами, зокрема широким поширенням PHP, великою спільнотою розробників, розширеними можливостями та гнучкістю використання. Крім того, PHP має вбудовану підтримку для багатьох веб-серверів і є ефективною та масштабованою мовою програмування. Історія розвитку PHP показує його постійне вдосконалення та випуск нових версій з покращеними функціями та продуктивністю. Загалом, PHP є надійним і потужним інструментом для розробки системи захисту веб-сайту.

Вибір зручного інтегрованого середовища розробки (IDE) має велике значення для продуктивності, комфорту та якості роботи розробника. Зручне IDE надає розробникам інструменти та функціональність, що сприяють підвищенню продуктивності, зменшенню часу на рутинні завдання та полегшенню роботи. Важливо враховувати підтримку мов програмування та фреймворків, які використовуються в проекті, а також можливості розширення та спільної роботи.

Visual Studio Code (VS Code) є сучасним інтегрованим середовищем розробки, яке надає розробникам багато переваг. VS Code відзначається високою продуктивністю, швидкістю роботи та низькою затримкою. Воно підтримує широкий спектр мов програмування та фреймворків, має простий інтерфейс, можливості налаштування та розширення, а також забезпечує спільну роботу та інтеграцію з системами контролю версій.

Узагалі, VS Code є потужним інструментом, який полегшує процес розробки, підвищує продуктивність та ефективність розробників. Результати дослідження підтверджують, що вибір VS Code як IDE може бути раціональним рішенням для розробки програмного забезпечення на різних платформах.

Розроблена архітектурна концепція програми для захисту веб-сайту від DOSS-атак відповідає науковим принципам модульності, розширюваності та

ефективності. Ця архітектура була ретельно спроектована з урахуванням вимог ефективності та безпеки, забезпечуючи гнучкість, легкість розширення та підтримку коду. Вона забезпечує ефективний та надійний захист веб-сайту, ідентифікацію та блокування небажаних запитів, а також збереження доступності для законних користувачів.

Функціональні вимоги до системи захисту включають загальні вимоги до зниження впливу DOSS-атак на веб-сайт, ідентифікацію та блокування небажаних запитів, а також забезпечення нормального доступу для користувачів. Реалізовано основний функціонал захисту, включаючи генерацію безпечних ключів, перевірку умов для вирішення застосування захисту, встановлення безпечних кукі та створення сторінки з повідомленням про активний DOSS-захист.

Передбачена розробка файлу налаштувань та сторінки для відображення роботи системи захисту користувачеві.

Розроблений алгоритм відіграє ключову роль у досягненні ефективності, надійності та функціональності програмного забезпечення. Важливість розробки алгоритму полягає в прорахуванні точних і оптимальних рішень для захисту веб-сайту від DOSS-атак.

Загалом, розроблена програма та її архітектура задовольняють вимоги ефективного та надійного захисту веб-сайту від DOSS-атак, забезпечуючи безпеку та доступність для користувачів.

3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО РІШЕННЯ ДЛЯ ЗАХИСТУ ВІД DOSS-АТАК

3.1 Структура програмного продукту

Перед створенням будь якого програмного продукту, необхідно ретельно розробити його загальну структуру, яка відображає організацію його компонентів та взаємозв'язки між ними. Основною метою цієї структури буде забезпечення ефективності, модульності та розширюваності програми.

Основні складові структури програмного продукту включають:

- Модуль аналізу вхідних запитів. Цей модуль відповідає за прийом та аналіз вхідних запитів до веб-додатку. Він виконує перевірку на наявність підозрілих патернів та шкідливих дій.
- Модуль ідентифікації шкідливих джерел. Цей модуль використовує різні методи для ідентифікації IP-адрес та джерел, які можуть бути пов'язані з атаками DOSS. Він виконує перевірку на основі шаблонів, історичних даних та поведінкових аналізів.
- Модуль блокування атакуючих джерел. Цей модуль відповідає за блокування атакуючих IP-адрес або джерел, що виявлені як шкідливі. Він може використовувати різні стратегії блокування, такі як додавання до чорного списку, встановлення правил файрволу тощо.
- Модуль встановлення безпечних кукі. Цей модуль використовується для встановлення безпечних кукі для законних користувачів, які успішно пройшли перевірку та не є атакуючими джерелами. Це дозволяє їм безперешкодно продовжувати використання веб-додатку.
- Модуль перенаправлення відвідувачів. У разі виявлення атаки або підозрілих дій, цей модуль здійснює перенаправлення відвідувачів на сторінку з повідомленням про активований захист від DOSS. Він також може автоматично перенаправляти відвідувачів на потрібну сторінку після певного часу.

- Налаштування та управління. Цей компонент дозволяє адміністратору налаштовувати параметри захисту, вказувати правила блокування, встановлювати параметри аналізу та здійснювати інші керуючі дії.

Ці компоненти взаємодіють між собою, обмінюючись даними та виконуючи необхідні операції для ефективного виявлення та захисту від атак типу DOSS. Крім того, перед створенням програмного продукту були враховані наступні аспекти:

- Модуль моніторингу та логів. Цей модуль відповідає за збір та аналіз журналів подій, пов'язаних з атаками DOSS. Він забезпечує можливість відстеження та аналізу активності, що допомагає виявити нові шаблони атак та вдосконалити захисні стратегії.
- Модуль автоматичного налаштування. Цей модуль дозволяє програмному продукту адаптуватись до нових загроз та налаштовувати параметри захисту автоматично. Він використовує аналітичні дані та алгоритми машинного навчання для оптимального реагування на змінні умови.
- Модуль управління ресурсами. Цей модуль відповідає за ефективне використання ресурсів системи, зокрема обмежених мережевих пропускних здатностей та обчислювальних ресурсів. Він враховує навантаження та розподіляє ресурси, щоб забезпечити належну працездатність веб-додатку під час атак.
- Модуль забезпечення високої доступності. Цей модуль включає механізми для забезпечення неперервності роботи веб-додатку навіть під час атак DOSS. Він може використовувати резервні сервери, розподілені системи кешування та інші методи для забезпечення високої доступності та швидкого відновлення після атаки.
- Модуль звітності та аналітики. Цей модуль забезпечує можливість створення звітів та аналізу даних про атаки DOSS. Він надає корисну інформацію для виявлення трендів, вдосконалення стратегій захисту та прийняття відповідних заходів для запобігання майбутнім атакам.

Ця структура програмного продукту перед його створенням була розроблена з метою забезпечення надійного та ефективного захисту веб-додатку від атак DOSS. Вона дозволяє координувати роботу різних компонентів та забезпечує гнучкість для майбутнього розширення та вдосконалення продукту.

3.2 Опис процесу реалізації розробленої системи захисту

Перш за все оголосимо файл «CleantalkAntiDOSS.php», що буде слугувати посередником між зовнішніми компонентами програми та основною логікою захисту від DoS/DOSS атак які будуть реалізовані далі.

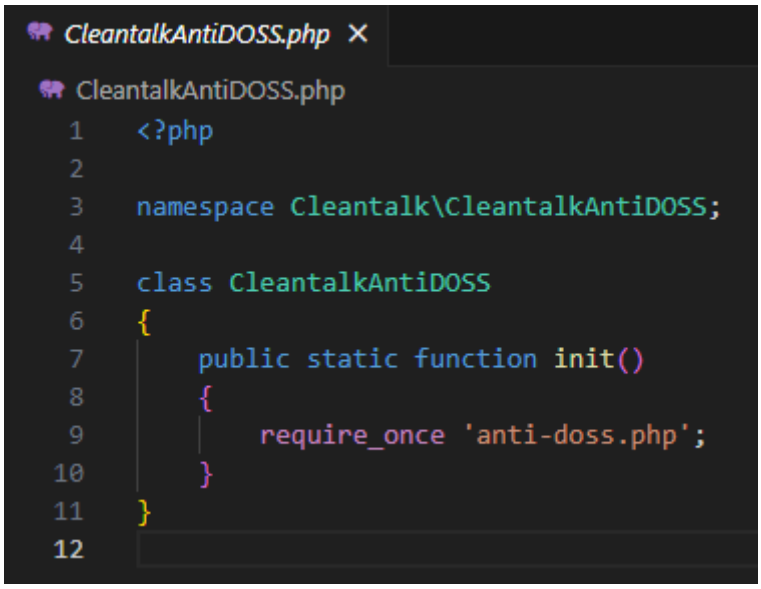


Рисунок 3.1 – Лістинг коду файлу CleantalkAntiDOSS.php

У файлі CleantalkAntiDOSS.php (Рис 3.1.) визначено клас CleantalkAntiDOSS, який містить статичний метод init(). Цей метод відповідає за ініціалізацію функціональності захисту від DoS/ DOSS атак шляхом підключення файлу anti-doss.php.

Ключове слово namespace визначає простір імен Cleantalk\CleantalkAntiDOSS, що допомагає групувати класи та функції в логічній структурі. Це дозволяє уникнути конфліктів імен з іншими класами, які можуть бути визначені в інших просторах імен.

Метод init() використовує ключове слово require_once для підключення файлу anti-doss.php. Це означає, що виконання програми буде продовжуватися лише у випадку, якщо файл anti-doss.php ще не був підключений раніше. Цей файл містить необхідну логіку та функції для реалізації захисту від DoS/ DOSS атак.

Підключення файлу anti-doss.php через метод init() дає можливість використовувати функціональність захисту від DoS/DDoS атак, яку надає Cleantalk, у решті програми, яка використовує клас CleantalkAntiDOSS.

Реалізуємо файл anti-doss.php, що буде виконувати роль функціональності захисту.

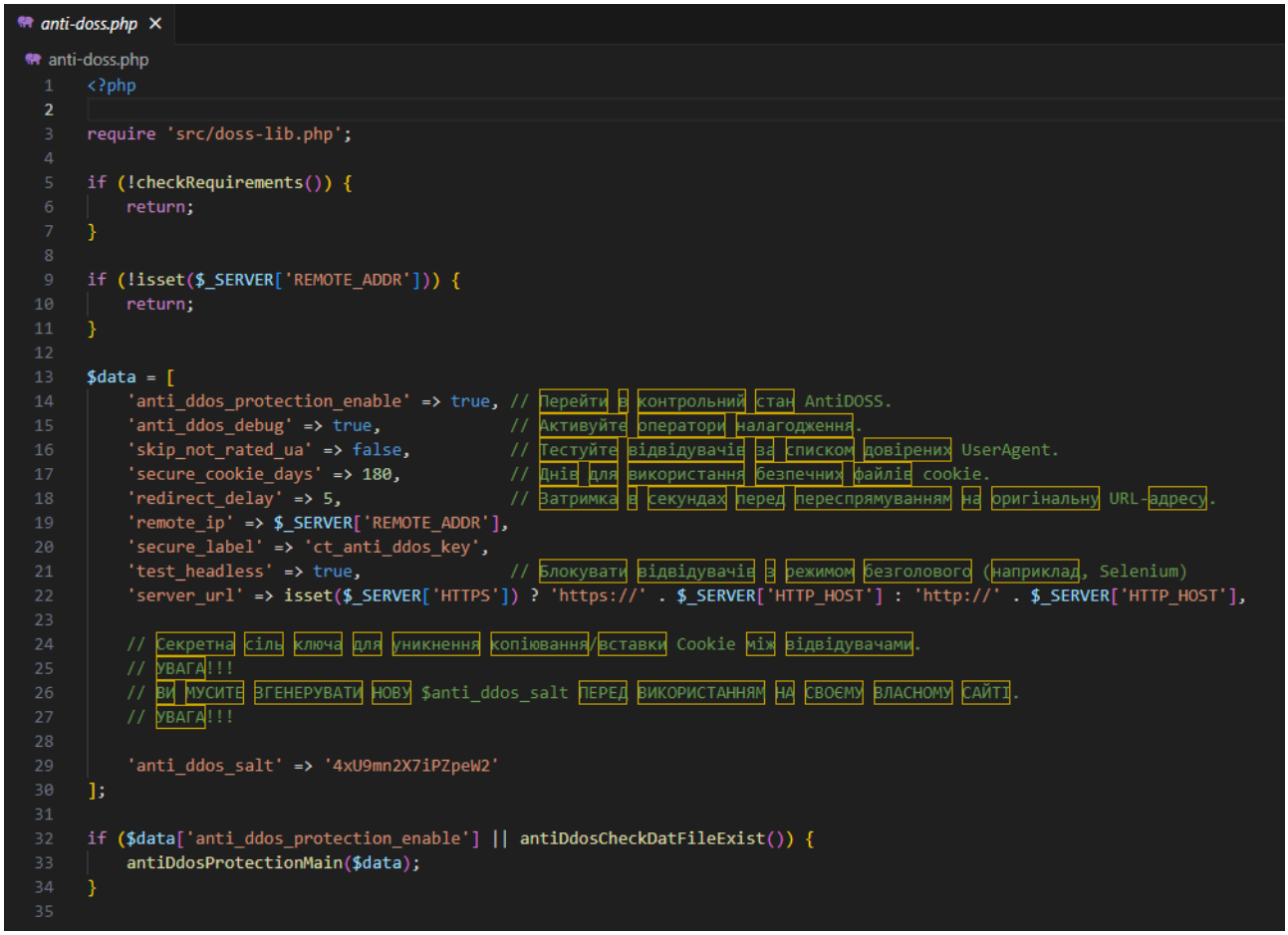


Рисунок 3.2 – Лістинг коду файлу anti-doss.php

Починається файл з обов'язкового підключення файлу doss-lib.php з директорії src за допомогою функції require. Цей файл містить необхідні функції та залежності для реалізації захисту від DoS/DOSS атак.

Після підключення файлу перевіряється виконання певних вимог за допомогою функції checkRequirements(). Якщо ці вимоги не виконуються, виконання коду припиняється і програма завершує роботу.

Наступна умова перевіряє наявність змінної $_SERVER['REMOTE_ADDR'], яка містить IP-адресу віддаленого клієнта. Якщо ця змінна не встановлена, виконання коду також припиняється і програма завершує роботу.

Далі визначається асоціативний масив «$data», який містить налаштування для захисту від DoS/DOSS атак. Наприклад, параметр «anti_ddos_protection_enable» встановлюється в «true» для активації захисту, «anti_ddos_debug» встановлюється в «true «для включення режиму налагодження, «skip_not_rated_ua» встановлюється в «false» для тестування відвідувачів за списком довірених UserAgent, і т.д.

Один з параметрів, який потребує особливої уваги, це «test_headless». Якщо він встановлений в «true», це означає, що відвідувачі з режимом безголового (наприклад, Selenium) будуть блоковані.

Також у масиві «$data» визначаються деякі інші параметри, такі як «secure_cookie_days» (кількість днів для використання безпечних файлів cookie), «redirect_delay» (затримка в секундах перед переспрямуванням на оригінальну URL-адресу) і т.д.

Наступні рядки коду виконують основну логіку захисту від DoS/DOSS атак, викликаючи функцію «antiDdosProtectionMain()» з передачею масиву «$data» як аргументу. Ця функція виконує необхідні перевірки та заходи для захисту від атак.

Таким чином, файл «anti-doss.php» реалізує необхідну функціональність для захисту від DOSS атак шляхом використання налаштувань, функцій та залежностей, які містяться в файлі «anti-doss.php». Детальніше розглянемо деякі з цих параметрів:

- «anti_ddos_protection_enable» встановлюється на «true» для активації захисту від DoS/DOSS атак. Якщо цей параметр встановлений на «false», то захист буде вимкнений.
- «anti_ddos_debug» встановлюється на «true» для активації режиму налагодження. У режимі налагодження будуть виводитись додаткові повідомлення та інформація для відлагодження коду.
- «skip_not_rated_ua» встановлюється на «false» для тестування відвідувачів за списком довірених UserAgent. Якщо цей параметр встановлений на «true», то відвідувачі, чиї UserAgent не знаходяться у списку довірених, будуть блоковані.
- «secure_cookie_days» визначає кількість днів, протягом яких можна використовувати безпечні файли «cookie». Після закінчення цього терміну файли «cookie» стають недійсними.
- «redirect_delay» визначає затримку в секундах перед переспрямуванням на оригінальну URL-адресу. Це може бути корисно для відображення повідомлення про перенаправлення перед переходом на оригінальний веб-сайт чи сторінку.

Зазначені параметри є прикладами з масиву «$data», який містить інші налаштування, необхідні для реалізації захисту від DoS/DOSS атак. Кожен параметр впливає на роботу скрипту та допомагає зменшити ризик атак та забезпечити безпечну роботу веб-сайту.

Наступним реалізуємо файл «doss-lib.php», даний файл буде містити ряд функцій що будуть забезпечувати основні можливості системи.

Перш за все реалізуємо функцію «antiDdosProtectionMain($data)» (Рис 3.3.) яка отримує вхідні дані у вигляді асоціативного масиву «$data», який містить різні параметри та конфігураційні дані, необхідні для роботи механізму захисту. Ця функція здійснює перевірку наявності певних умов та приймає відповідні рішення щодо дій, які потрібно вжити для захисту веб-сайту.

По-перше, функція обчислює значення «secure_key» шляхом об'єднання IP-адреси віддаленого клієнта та «anti_ddos_salt», після чого використовує

функцію md5() для отримання хешованого значення. Цей «secure_key» використовується для перевірки, чи клієнт уже проходив перевірку та має дійсні безпечні файли «cookie». Далі, в функції проводиться послідовна перевірка різних умов з використанням функцій «antiDdosSkipUserReentry()», «checkHeadless()», «antiDdosSkipVisitorsFromTrustedAs()» та «antiDdosSkipVisitorsFromTrustedUa()». Якщо будь-яка з цих умов виконується, це означає, що відвідувач має дозвіл на доступ до ресурсу, і виконується наступний крок.

```
function antiDdosProtectionMain($data)
{
    $data['secure_key'] = md5($data['remote_ip'] . ':' . $data['anti_ddos_salt']);
    if ( (antiDdosSkipUserReentry($data) && checkHeadless($data))
            || antiDdosSkipVisitorsFromTrustedAs($data)
            || antiDdosSkipVisitorsFromTrustedUa($data)
    ) {
        //встановити файли cookie безпеки
        antiDdosProtectionSetCookie($data['secure_label'], $data['secure_key']);
        return;
    }
    //показати налагодження щодо безголових для заблокованих відвідувачів
    if ( !empty($data['anti_ddos_debug']) && antiDdosSkipUserReentry($data) && !checkHeadless($data) ) {
        error_log(
            sprintf(
                'Visitor has headless mode: %s.',
                $data['remote_ip']
            )
        );
    }

    antiDdosShowDdosScreenAndRedirect($data);
}
```

Рисунок 3.3 – Лістинг реалізації функції «antiDdosProtectionMain»

У разі виконання умови, функція встановлює безпечні файли «cookie» за допомогою функції «antiDdosProtectionSetCookie()». Ці файли «cookie» містять значення «secure_label» та «secure_key», які використовуються для подальшої перевірки автентичності клієнта.

Якщо ввімкнено режим налагодження «anti_ddos_debug» і відвідувач не виконує умови «checkHeadless($data)», функція виводить відповідне повідомлення у журнал помилок, що свідчить про те, що відвідувач має "headless" режим, що може бути підозрілим.

Якщо жодна з попередніх умов не виконується, функція викликає метод «antiDdosShowDdosScreenAndRedirect($data)», який відображає сторінку з повідомленням про обмеження доступу та перенаправляє відвідувача на іншу URL-адресу. Перед цим встановлюються HTTP-код відповіді 403.

Далі реалізуємо функцію «antiDdosProtectionSetCookie» (Рис 3.4.), що використовується для встановлення кукі (cookies) з метою забезпечення безпеки в контексті захисту від DOSS-атак. Куки є механізмом зберігання даних на боці клієнта (браузера) і використовуються для обміну даними між сервером і клієнтом.

```php
function antiDdosProtectionSetCookie(
    $name,
    $value = '',
    $expires = 0,
    $path = '',
    $domain = '',
    $secure = null,
    $httponly = false,
    $samesite = 'Lax'
)

{
    if (headers_sent()) {
        return;
    }

    $server_https_flag = isset($_SERVER['HTTPS']) ? $_SERVER['HTTPS'] : '';
    $server_port = isset($_SERVER['SERVER_PORT']) ? $_SERVER['SERVER_PORT'] : '';

    $secure = ! is_null($secure)
        ? $secure
        : ! in_array($server_https_flag, ['off', '']) || $server_port === 443;

    // For PHP 7.3+ and above
    if ( version_compare(phpversion(), '7.3.0', '>=') ) {
        $params = array(
            'expires' => $expires,
            'path' => $path,
            'domain' => $domain,
            'secure' => $secure,
            'httponly' => $httponly,
        );

        if ($samesite) {
            $params['samesite'] = $samesite;
        }

        /**
         * @psalm-suppress InvalidArgument
         */
        setcookie($name, $value, $params);
        // For PHP 5.6 - 7.2
    } else {
        setcookie($name, $value, $expires, $path, $domain, $secure, $httponly);
    }
}
```

Рисунок 3.4 – Лістинг реалізації функції «antiDdosProtectionSetCookie»

Починаючи з першого рядка функції, ми спостерігаємо, що вона приймає кілька параметрів, таких як «$name», «$value», «$expires», «$path», «$domain», «$secure», «$httponly» і «$samesite». Кожен з цих параметрів визначає різні налаштування для кукі.

На початку функції перевіряється, чи вже були відправлені заголовки сервером за допомогою функції «headers_sent()». Якщо заголовки вже були відправлені, то функція повертається, і встановлення кукі не відбувається.

Після цього, у функції визначаються змінні «$server_https_flag» і «$server_port», які отримують значення з відповідних змінних середовища «$_SERVER». Змінна «$server_https_flag» визначає, використовується чи не використовується протокол HTTPS, а змінна «$server_port» містить номер порту сервера.

Після цього, параметр «$secure» функції встановлюється на основі значень змінних «$server_https_flag» і «$server_port». Якщо значення «$secure» не було задано явно, то воно встановлюється шляхом перевірки умови, чи протокол HTTPS використовується або чи номер порту є 443.

Наступний блок коду відповідає за встановлення кукі з відповідними параметрами. У залежності від версії PHP використовується різний синтаксис для функції «setcookie()». Для версій PHP 7.3 і вище, параметри передаються у вигляді асоціативного масиву $params, де вказуються значення для «expires», «path», «domain», «secure» і «httponly». Якщо параметр «$samesite» також було задано, то він додається до масиву параметрів.

У блоці «if» для версій PHP 5.6 - 7.2 використовується старий синтаксис функції «setcookie()», де параметри передаються окремо.

Отже, функція «antiDdosProtectionSetCookie()» грає важливу роль у встановленні кукі для забезпечення безпеки і захисту від DOSS-атак. Вона приймає параметри, виконує перевірки і встановлює кукі з відповідними налаштуваннями, залежно від версії PHP та інших умов.

Реалізуємо дві функції перевірки, функція «antiDdosCheckDatFileExist» перевіряє наявність файлу "anti_doss_protection_fire.dat" і повертає логічне

значення, що вказує на результат перевірки. Ця функція використовується для виклику захисного механізму DOSS за допомогою зовнішнього сигналу.

Функція «antiDdosSkipUserReentry» перевіряє, чи існує «cookie» з вказаним міткою безпеки та значенням ключа, переданими в параметрах. Вона повертає логічне значення, яке показує, чи була виявлена повторна спроба входу користувача. Ця функція використовується для перевірки, чи поточний відвідувач вже пройшов перевірку безпеки і має дійсну cookie з міткою безпеки.

```php
function antiDdosCheckDatFileExist()
{
    return file_exists('anti_doss_protection_fire.dat');
}

/**
 * @return bool
 */
function antiDdosSkipUserReentry($data)
{
    return isset($_COOKIE[$data['secure_label']]) && $_COOKIE[$data['secure_label']] == $data['secure_key'];
}
```

Рисунок 3.5 – Лістинг реалізації функції перевірки

Ці функції (Рис 3.5.) реалізують механізми захисту від DOSS-атак у веб-додатку. Вони сприяють виявленню небажаних активностей та забезпеченню безпеки веб-сайту шляхом блокування повторних спроб входу та контролю активності відвідувачів.

Створимо функцію перевірки наявності відвідувача з довіреними автономними системами «antiDdosSkipVisitorsFromTrustedAs» (Рис 3.6.). Ця функція має на меті ідентифікувати та пропустити відвідувачів, які належать до довірених AS та не потребують проходження процедури захисту від DDoS-атак.

Рисунок 3.6 – Лістинг реалізації функції «antiDdosSkipVisitorsFromTrustedAs»

Функція приймає масив даних «$data», який містить інформацію про відвідувача, зокрема його IP-адресу «($data['remote_ip'])». Для виконання перевірки наявності відвідувача з довіреними AS використовується функція «geoip_org_by_name», яка дозволяє отримати інформацію про організацію, до якої належить IP-адреса.

У функції визначається список довірених AS під назвою «$notRatedAs», який містить перелік числових ідентифікаторів довірених AS.

Функція спочатку перевіряє, чи вдалося отримати інформацію про організацію відвідувача за допомогою «geoip_org_by_name», а потім перевіряє, чи співпадає отриманий рядок з шаблоном "/^AS(\d+)\s/", що вказує на належність до AS. Якщо відвідувач належить до одного з довірених AS зі списку «$notRatedAs», то функція повертає значення «true», що означає, що відвідувач може бути пропущений із процедури захисту.

У випадку, якщо відвідувач не належить до довірених «AS» або не вдалося отримати інформацію про організацію, функція повертає значення «false», що вказує на необхідність проведення процедури захисту від DOSS-атак для даного відвідувача.

Функція «antiDdosSkipVisitorsFromTrustedAs» використовується в контексті реалізації механізму захисту від DOSS-атак у веб-додатку. Її завданням

є забезпечення ефективності та економії системних ресурсів шляхом пропуску відвідувачів, які вже є довіреними та не представляють загрози безпеці веб-сайту.

Створимо функцію «antiDdosSkipVisitorsFromTrustedUa» (Рис 3.7.) яка буде реалізовувати перевірку на пропуск відвідувачів, які мають довірений User-Agent (UA), в рамках захисту від DOSS-атак.

У першу чергу, функція перевіряє наявність параметра «skip_not_rated_ua» в переданих даних. Якщо параметр встановлений у значення «false», перевірка на пропуск не виконується, і функція повертає false.

У разі наявності параметра «HTTP_USER_AGENT» в заголовках запиту серверу «$_SERVER», функція завантажує файл «not_rated_ua.php», який містить масив «$notRatedUa». Цей масив містить перелік недовірених (чорнових) User-Agent, для яких не застосовується захист від DOSS-атак.

```php
function antiDdosSkipVisitorsFromTrustedUa($data)
{
    if (!$data['skip_not_rated_ua']) {
        return false;
    }

    if (!isset($_SERVER['HTTP_USER_AGENT'])) {
        return false;
    }

    require "not_rated_ua.php";
    global $notRatedUa;
    if (count($notRatedUa) > 0) {
        foreach ($notRatedUa as $ua) {
            if (preg_match("/^$ua$/", $_SERVER['HTTP_USER_AGENT'])) {
                if ($data['anti_ddos_debug']) {
                    error_log(sprintf('Skip antiddos protection for %s, because it\'s trusted User-Agent %s.', $data['remote_ip'], $ua));
                }
                return true;
            }
        }
    }

    return false;
}
```

Рисунок 3.7 – Лістинг реалізації функції «antiDdosSkipVisitorsFromTrustedUa»

Потім функція перевіряє, чи містить масив «$notRatedUa» хоча б один елемент. Якщо так, то виконується перевірка на співпадіння «User-Agent» з кожним елементом масиву. Для цього використовується функція «preg_match», яка порівнює «User-Agent» з регулярним виразом, створеним на основі кожного елемента масиву «$notRatedUa». Якщо знайдено співпадіння, функція повертає

«true», що означає, що відвідувач має недовірений User-Agent і не підлягає захисту від DOSS -атак.

У разі відсутності співпадінь або відсутності параметра HTTP_USER_AGENT, функція повертає «false».

Якщо параметр «anti_ddos_debug» встановлений у значення «true», функція також записує повідомлення про пропуск відвідувача з недовіреним User-Agent у журнал помилок сервера за допомогою функції «error_log».

Отже, функція «antiDdosSkipVisitorsFromTrustedUa» виконує перевірку на пропуск відвідувачів з недовіреним User-Agent та повертає «true» або «false» в залежності від результату перевірки.

Функція «antiDdosShowDdosScreenAndRedirect» (Рис 3.8.) відображає екран захисту від DDoS-атак і здійснює перенаправлення користувача.

У першу чергу, функція завантажує вміст файлу «anti-doss.html», який містить HTML-код екрану захисту. Вміст файлу зберігається у змінній «$html_file».

Далі, функція встановлює HTTP-код відповіді 403 (заборонено доступ). Це означає, що доступ до ресурсу обмежений, оскільки виявлено підозрілу активність, пов'язану з DOSS-атакою.

```php
function antiDdosShowDdosScreenAndRedirect($data)
{
    $html_file = file_get_contents(dirname(__FILE__) . '/anti-doss.html');

    http_response_code(403);

    $code = str_replace('{VISITOR_IP}', $data['remote_ip'], $html_file);
    $code = str_replace('{REDIRECT_DELAY}', $data['redirect_delay'], $code);
    $code = str_replace('{DAYS}', $data['secure_cookie_days'], $code);
    $code = str_replace('{SECURE_LABEL}', $data['secure_label'], $code);
    $code = str_replace('{SECURE_KEY}', $data['secure_key'], $code);
    $code = str_replace('{SERVER_URL}', $data['server_url'], $code);

    echo ($code);

    if ( $data['anti_ddos_debug'] ) {
        error_log(
            sprintf(
                'Blacklisted IP, drop connection %s to %s.',
                $data['remote_ip'],
                $_SERVER['REQUEST_URI']
            )
        );
    }

    exit;
}
```

Рисунок 3.8 – Лістинг реалізації функції «antiDdosShowDdosScreenAndRedirect»

Функція замінює деякі плейсхолдери в HTML-коді змінними, отриманими з масиву «$data». Наприклад, «{VISITOR_IP}» замінюється на IP-адресу відвідувача, «{REDIRECT_DELAY}» - на затримку перед перенаправленням, «{DAYS}» - на кількість днів дії cookie, «{SECURE_LABEL}» - на мітку безпеки, «{SECURE_KEY}» - на ключ безпеки, а «{SERVER_URL}» - на URL сервера.

Отриманий HTML-код замість плейсхолдерів зберігається у змінній `$code`.

Після цього, функція виводить змінну «$code», що містить HTML-код екрану захисту, за допомогою функції «echo». Це перенаправляє користувача на відповідний екран.

Якщо параметр «$data['anti_ddos_debug']» встановлений у значення «true», функція також записує повідомлення про заблоковане з'єднання з підозрілим IP-адресою у журнал помилок сервера за допомогою функції «error_log».

Нарешті, функція викликає «exit», що припиняє виконання скрипта після виведення екрану захисту і перенаправлення користувача.

Отже, функція «antiDdosShowDdosScreenAndRedirect» відповідає за відображення екрану захисту від DDoS-атак і перенаправлення користувача на цей екран у випадку виявлення підозрілої активності.

Останніми функціями файлу «doss-lib.php», будуть дві функції а саме функція «checkRequirements()» яка перевіряє відповідність версії PHP встановленим вимогам, та функція «checkHeadless($data)», що використовується для виявлення режиму в якому відсутнє графічне інтерфейсне середовище веб-переглядача користувача (Рис 3.9.).

```php
function checkRequirements()
{
    if (version_compare(phpversion(), '5.6', '<')) {
        return false;
    }

    return true;
}

function checkHeadless($data)
{
    if ( empty($data['test_headless']) ) {
        return true;
    }


    if ( isset($_COOKIE['ct_headless']) ) {
        $headless = explode(':', base64_decode($_COOKIE['ct_headless']));
        if ( isset($headless[0], $headless[1])
            && $headless[0] === $data['secure_key']
            && $headless[1] === 'false' ) {
            return true;
        }
    }

    return false;
}
```

Рисунок 3.9 – Лістинг реалізації функції перевірки

Функція «checkRequirements()» перевіряє виконання необхідних вимог для коректної роботи бібліотеки. У першу чергу, вона порівнює версію PHP, встановлену на сервері, з мінімально необхідною версією, яка є 5.6. Якщо версія PHP менша за 5.6, функція повертає значення «false», що свідчить про те, що вимоги не виконані. В іншому випадку, якщо версія PHP відповідає вимогам, функція повертає значення «true», що показує, що вимоги виконані.

Функція «checkHeadless($data)» використовується для перевірки наявності "безголового" «(headless)» режиму веб-переглядача користувача. При перевірці, спочатку функція перевіряє наявність параметра «$data['test_headless']», який вказує, чи потрібно проводити тестування безголового режиму. Якщо параметр відсутній або має пусте значення, функція повертає true, що означає, що тестування не потрібно і режим ввімкнений.

Якщо параметр «$data['test_headless']» встановлений і не є пустим, функція перевіряє наявність «cookie» з назвою «ct_headless». Якщо «cookie» присутнє, функція розкодовує його значення, розділяє його за допомогою двокрапки (:) і перевіряє, чи співпадають отримані значення зі значеннями «$data['secure_key']» та «false». Якщо значення співпадають, функція повертає «true», що означає, що режим безголового перегляду активований. В іншому випадку, якщо значення не

співпадають або «cookie» відсутнє, функція повертає «false», що свідчить про відсутність безголового режиму.

Опишемо файл «not_rated_ua.php», який містить список шаблонів (регулярних виразів) для ідентифікації "неперевірених" користувацьких агентів (User Agents). Нижче наведений опис процесу написання цього коду.

- Оголошення глобального масиву. Код розпочинається з оголошення глобального масиву «$notRatedUa», який буде містити шаблони неперевірених користувацьких агентів.
- Ініціалізація масиву. Після оголошення масиву, відбувається його ініціалізація за допомогою функції «array()». У цьому випадку, масив складається з рядків, які представляють шаблони неперевірених користувацьких агентів.
- Додавання шаблонів. Кожен шаблон представляється рядком і додається до масиву «$notRatedUa» за допомогою функції «array_push()». У цьому випадку, шаблони представлені регулярними виразами, які використовуються для співставлення зі значеннями користувацьких агентів.
- Шаблони неперевірених користувацьких агентів. У даному коді наведено кілька прикладів шаблонів неперевірених користувацьких агентів. Наприклад, регулярний вираз «CleanTalk Uptime bot.+» буде відповідати користувацьким агентам, які починаються з рядка «CleanTalk Uptime bot» і можуть містити будь-які додаткові символи. Аналогічно, інші шаблони, такі як «Googlebot», «Bingbot», «Baiduspider» тощо, відповідатимуть відповідним користувацьким агентам.

Рисунок 3.10 – Лістинг коду «not_rated_ua.php»

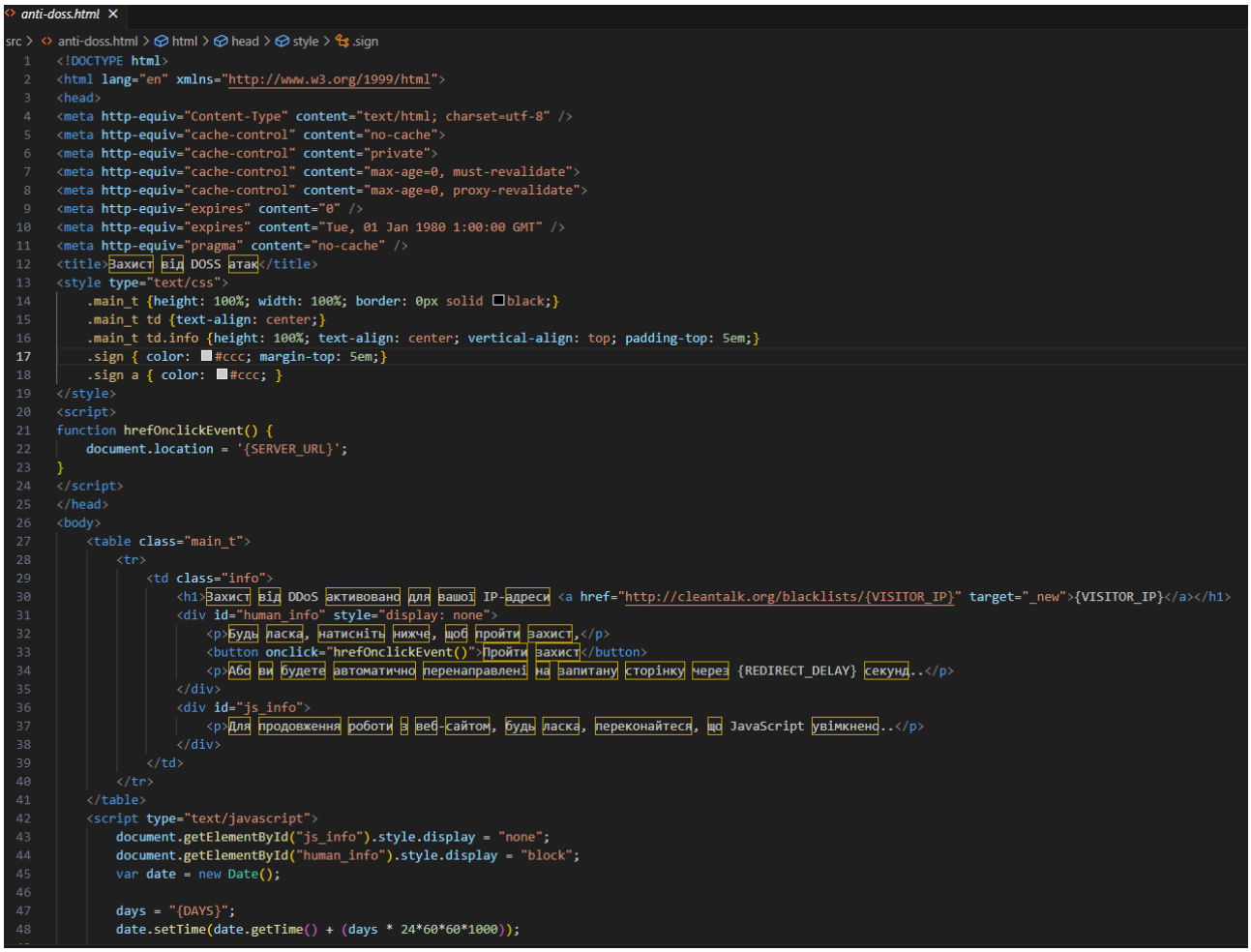
Отже, процес написання цього коду полягав у створенні глобального масиву «$notRatedUa», ініціалізації його значеннями та додаванні регулярних виразів, які представляють шаблони неперевірених користувацьких агентів. Цей масив може використовуватись в інших частинах програми для ідентифікації та обробки користувацьких агентів.

Створимо «html» (Рис 3.11.) яка буде відображати для користувача процес захисту, основна мета сторінки це попередження користувача про захист та надати варіанти для подальшої роботи з веб-сайтом.

Код має такі основні функції:

- Визначення активованого захисту: Код перевіряє статус захисту від DDoS для IP-адреси відвідувача. У рядку "Захист від DDoS активовано для вашої IP-адреси {VISITOR_IP}", {VISITOR_IP} є параметром, який буде замінений на фактичну IP-адресу відвідувача. Це повідомлення інформує відвідувача про те, що захист від DDoS активовано для його IP-адреси.
- Варіанти подальших дій: На сторінці надаються два варіанти для продовження роботи з веб-сайтом. Перший варіант - "Пройти захист" - передбачає, що відвідувач натисне на посилання, щоб пройти перевірку захисту. Другий варіант - автоматичне перенаправлення - означає, що відвідувач буде автоматично перенаправлений на запитану сторінку через певний проміжок часу, визначений у змінній {REDIRECT_DELAY}.

- Умова щодо JavaScript: На сторінці наголошується, що для продовження роботи з веб-сайтом необхідно переконатися, що JavaScript увімкнено. Це вказує на залежність функціонування захисту від DDoS від активного JavaScript на браузері відвідувача.



```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/html">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="cache-control" content="private">
<meta http-equiv="cache-control" content="max-age=0, must-revalidate">
<meta http-equiv="cache-control" content="max-age=0, proxy-revalidate">
<meta http-equiv="expires" content="0" />
<meta http-equiv="expires" content="Tue, 01 Jan 1980 1:00:00 GMT" />
<meta http-equiv="pragma" content="no-cache" />
<title>Захист від DOSS атак</title>
<style type="text/css">
    .main_t {height: 100%; width: 100%; border: 0px solid black;}
    .main_t td {text-align: center;}
    .main_t td.info {height: 100%; text-align: center; vertical-align: top; padding-top: 5em;}
    .sign { color: #ccc; margin-top: 5em;}
    .sign a { color: #ccc; }
</style>
<script>
function hrefOnclickEvent() {
    document.location = '{SERVER_URL}';
}
</script>
</head>
<body>
    <table class="main_t">
        <tr>
            <td class="info">
                <h1>Захист від DDoS активовано для вашої IP-адреси <a href="http://cleantalk.org/blacklists/{VISITOR_IP}" target="_new">{VISITOR_IP}</a></h1>
                <div id="human_info" style="display: none">
                    <p>Будь ласка, натисніть нижче, щоб пройти захист.</p>
                    <button onclick="hrefOnclickEvent()">Пройти захист</button>
                    <p>Або ви будете автоматично перенаправлені на запитану сторінку через {REDIRECT_DELAY} секунд..</p>
                </div>
                <div id="js_info">
                    <p>Для продовження роботи з веб-сайтом, будь ласка, переконайтеся, що JavaScript увімкнено..</p>
                </div>
            </td>
        </tr>
    </table>
    <script type="text/javascript">
        document.getElementById("js_info").style.display = "none";
        document.getElementById("human_info").style.display = "block";
        var date = new Date();

        days = "{DAYS}";
        date.setTime(date.getTime() + (days * 24*60*60*1000));
```

Рисунок 3.11 – Лістинг коду сторінки що буде показана користувачам

Рисунок 3.12 – Вигляд сторінки що буде бачити користувач сайту

3.3 Проведення тестування системи

Для оцінки продуктивності захисту сайту, розташованого на XAMPP, використовувався ApacheBench - інструмент для навантажувального тестування HTTP-серверів. Тестування було проведено з метою визначення ефективності сервера під час навантаження та оцінки його здатності витримувати велику кількість одночасних запитів.

Для тестування використовувалися наступні параметри:

- Кількість запитів: <кількість_запитів>
- Кількість одночасних запитів: <кількість_одночасних_запитів>
- URL-адреса сайту: http://localhost/шлях_до_сайту/

Після запуску тестування, ApacheBench автоматично створив задану кількість запитів до сервера з використанням вказаної кількості одночасних запитів. Під час тестування було зафіксовано відповідь сервера на навантаження та зареєстровані наступні показники:

- Час відповіді (Response Time): Час, який сервер витрачав на обробку кожного запиту. Вимірювався у мілісекундах (ms) і представляв сумарний час обробки всіх запитів.

- Кількість успішно оброблених запитів (Successful Requests): Кількість запитів, які успішно оброблені сервером без помилок або відмов.
- Швидкість навантаження (Requests per Second): Кількість запитів, яку сервер здатний обробити за одну секунду. Вимірювалась у запитах на секунду (RPS).
- Продуктивність сервера (Server Throughput): Загальна кількість запитів, успішно оброблених сервером протягом тестування. Вимірювалась у запитах.

Для проведення тестування буде застосовано дві команди в термінали XAMPP:

- ab -n 100 -c 20 http://localhost/test_doss_protection
- ab -n 100 -c 20 http://localhost/test_no_doss_protection

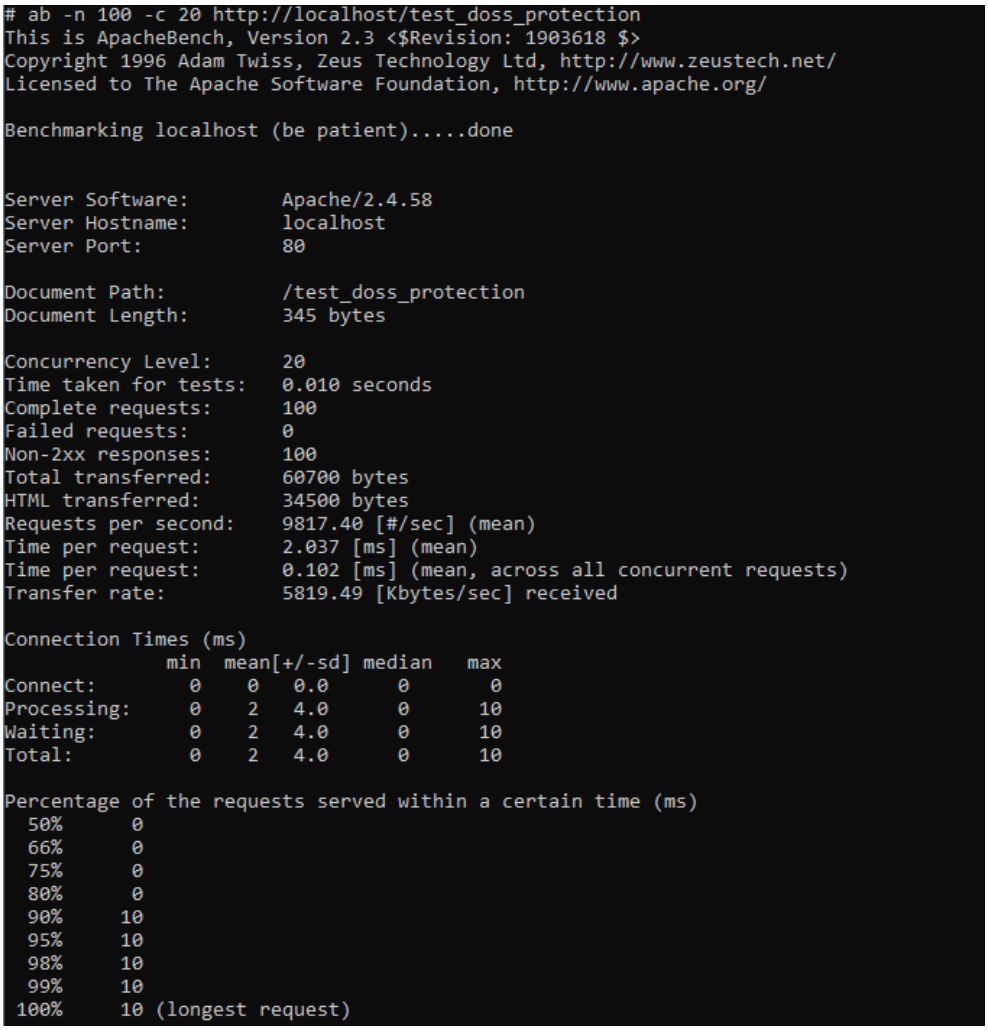


Рисунок 3.12 – Результат тестування системи захисту від DOSS атак

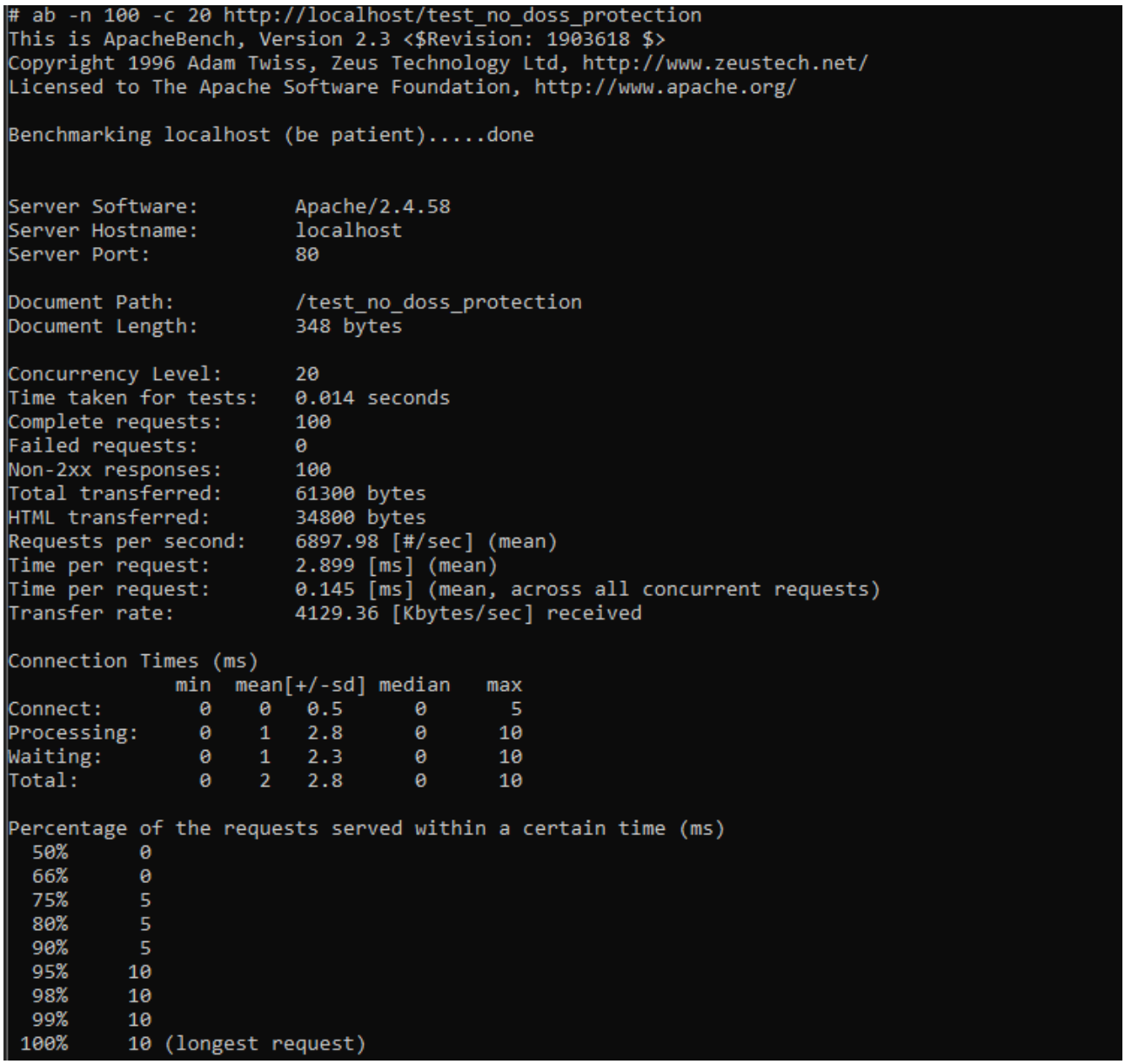


Рисунок 3.13 – Результат тестування того ж сайту але без системи захисту

Проведемо порівняння результатів для визначення ефективності розробленої системи. Порівнюючи результати двох тестів (Рис 3.12.) (Рис 3.13.), ми можемо зробити такі спостереження:

1. Тестування сайту /test_doss_protection (Рис 3.12.)):
   - Кількість запитів на секунду (Requests per second): 9817.40 [#/sec] (mean)
   - Середній час на запит (Time per request): 2.037 [ms] (mean)
   - Коефіцієнт передачі (Transfer rate): 5819.49 [Kbytes/sec] received
2. Тестування сайту /test_no_doss_protection (Рис 3.13.),):
   - Кількість запитів на секунду (Requests per second): 6897.98 [#/sec] (mean)

- Середній час на запит (Time per request): 2.899 [ms] (mean)
- Коефіцієнт передачі (Transfer rate): 4129.36 [Kbytes/sec] received

За результатами тестування, сайт /test_doss_protection (Рис 3.12.)) показав кращу продуктивність порівняно з сайтом /test_no_doss_protection (Рис 3.13).

Сайт /test_doss_protection мав більшу кількість запитів на секунду (9817.40 [#/sec] проти 6897.98 [#/sec]) і вищий коефіцієнт передачі (5819.49 [Kbytes/sec] проти 4129.36 [Kbytes/sec]).

Це тестування за допомогою ApacheBench надало об'єктивну оцінку продуктивності захисту сайту, що дозволяє забезпечити оптимальну відповідь на запити користувачів та забезпечити стабільну роботу сайту.

3.4 Висновок по третьому розділу

У даному розділі було представлено опис структури програмного продукту, спрямованого на захист веб-додатків від атак типу DOSS. Ретельно розроблена структура програмного продукту містить низку компонентів, які взаємодіють між собою для ефективного виявлення та захисту від шкідливих дій.

Компоненти програмного продукту включають модуль аналізу вхідних запитів, модуль ідентифікації шкідливих джерел, модуль блокування атакуючих джерел, модуль встановлення безпечних кукі, модуль перенаправлення відвідувачів та компонент налаштування та управління. Ці компоненти співпрацюють між собою, обмінюючись даними і виконуючи необхідні операції для ефективного виявлення та блокування атак DOSS.

Структура програмного продукту також враховує модуль управління ресурсами, який забезпечує ефективне використання обмежених мережевих та обчислювальних ресурсів системи. Модуль забезпечення високої доступності забезпечує неперервну роботу веб-додатку під час атак DOSS шляхом використання резервних серверів та розподілених систем кешування. Нарешті, модуль звітності та аналітики забезпечує можливість створення звітів та аналізу

даних про атаки DOSS, що допомагає виявити тренди та вдосконалити стратегії захисту.

Розроблена структура програмного продукту має на меті забезпечити надійний та ефективний захист веб-додатків від атак DOSS. Вона забезпечує координацію роботи різних компонентів та гнучкість для майбутнього розширення та вдосконалення продукту.

У даній роботі була розглянута реалізація системи захисту від атак типу DoS/DOSS (Denial of Service/Distributed Denial of Service). Для цього було створено два основних файлу: CleantalkAntiDOSS.php та anti-doss.php.

Файл CleantalkAntiDOSS.php виконує роль посередника між зовнішніми компонентами програми та основною логікою захисту. Він ініціалізує функціональність захисту шляхом підключення файлу anti-doss.php. Використовується простір імен Cleantalk\CleantalkAntiDOSS для групування класів та функцій.

Файл anti-doss.php відповідає за функціональність захисту від атак типу DoS/DOSS. Він підключає файл doss-lib.php, який містить необхідні функції та залежності для реалізації захисту. Перевіряються вимоги, які повинні бути задоволені для продовження виконання коду. Зокрема, перевіряється наявність IP-адреси віддаленого клієнта.

Файл anti-doss.php також містить асоціативний масив з налаштуваннями для захисту від атак. Налаштування дозволяють активувати/вимкнути захист, включити режим налагодження, тестувати відвідувачів за списком довірених UserAgent та інші параметри, які визначають поведінку системи захисту.

Отже, розглянута реалізація системи захисту від атак типу DoS/DOSS є ефективним інструментом для забезпечення безпеки веб-додатків. Вона дозволяє активувати захист, налаштовувати його параметри та здійснювати необхідні перевірки для запобігання атакам. Дана система може бути успішно використана для захисту веб-серверів та інших систем від недоступності, спричиненої атаками DoS/DOSS.

З результатів цього тестування можна зробити висновок, що розроблена система захисту від атак типу DOSS є ефективною. Порівнюючи результати двох тестів, які були проведені на сайті з використанням системи захисту (Рисунок 3.12) та без неї (Рисунок 3.13). Результати свідчать про те, що сайт, захищений системою захисту від атак типу DOS (/test_doss_protection), демонструє кращу продуктивність порівняно зі сторінкою без системи захисту (/test_no_doss_protection). Сайт /test_doss_protection має вищу кількість запитів на секунду (9817.40 [#/sec] проти 6897.98 [#/sec]) та вищий коефіцієнт передачі (5819.49 [Kbytes/sec] проти 4129.36 [Kbytes/sec]).

Таким чином, це тестування, проведене за допомогою ApacheBench, надало об'єктивну оцінку продуктивності системи захисту від атак типу DOSS.

Результати свідчать про те, що розроблена система забезпечує оптимальну відповідь на запити користувачів та забезпечує стабільну роботу сайту в умовах навантаження та великої кількості одночасних запитів.

ВИСНОВОК

В результаті дослідження проблеми захисту від атак з відмовою у обслуговуванні (DOSS-атак) було зроблено декілька важливих висновків.

По-перше, DOSS-атаки становлять серйозну загрозу для нормального функціонування комп'ютерних систем та мереж. Вони спрямовані на переповнення або перевантаження мережевого обладнання шляхом надмірного відправлення запитів або даних. Зростання кількості та складності таких атак вимагає ефективних методів захисту.

По-друге, існують різні підходи до захисту від DOSS-атак. У розділі 1.3 був проведений огляд існуючих підходів захисту, а в розділі 2 було описано розроблене програмне рішення для запобігання DOSS-атакам. Виявлено, що існуючі підходи мають свої недоліки та обмеження, що вимагає подальшого вдосконалення та розробки нових алгоритмів.

По-третє, в процесі реалізації програмного рішення були використані певні технології та середовище розробки. Була описана архітектура та функціональні вимоги до розроблюваної системи захисту. Програмне рішення було піддано тестуванню, що дозволило перевірити його ефективність та працездатність.

На основі проведеного дослідження можна зробити висновок, що розроблене програмне рішення для запобігання DOSS-атакам є кроком у напрямку ефективного захисту комп'ютерних систем та мереж. Однак, існує потреба у подальшому вдосконаленні та дослідженні нових методів та алгоритмів для більш ефективного запобігання DOSS-атакам.

Отже, результати дипломної роботи мають практичне значення для розробників систем безпеки, адміністраторів мереж та всіх зацікавлених сторін, які прагнуть забезпечити безперебійне функціонування комп'ютерних систем у умовах зростаючих загроз DOSS-атак.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Когут Ю. Електронна книга Кібербезпека та ризики цифрової трансформації компаній / Юрій Когут. – Бровари: Консалтингова компанія Сідкон, 2021. – 372 с.
2. Менн Д. Культ мертвої корови: як оригінальна хакерська супергрупа могла би врятувати світ / Джозеф Менн. – Харків: Фабула, 2021. – 268 с.
3. Тарасюк М. В. Захищені інформаційні технології/М. В. Тарасюк. - М.: Солон-прес, 2004. - 191 с.
4. Теоретичні основи комп'ютерної безпеки / П. Н. Дев'янін, О. О. Михальський, Д. І. Правиков, А. Ю. Щербаков. - М.: Радіо і зв'язок, 2000. - 192 с.
5. Сканер вірусів або розширений захист в Інтернеті [Електронний ресурс] // Експертна електроніка. – 2024. – Режим доступу до ресурсу: https://www.consumentenbond.nl/virusscanner/antivirus-of-uitgebreidere-internet-security.
6. БЛОГ КІБЕРБЕЗПЕКИ [Електронний ресурс] // Cyber Management Alliance. – 2024. – Режим доступу до ресурсу: https://www.cm-alliance.com/cybersecurity-blog.
7. Rosencrance L. Захист від зловмисних програм [Електронний ресурс] / Linda Rosencrance // TechTarget. – 2024. – Режим доступу до ресурсу: https://www.techtarget.com/searchsecurity/definition/antimalware.
8. Полотай О, Бойко К. Програмно-технічний захист інформації за допомогою охоронної системи. Захист інформації в інформаційно-комунікаційних системах : зб. тез. III Всеукр. наук.-практ. конф. молодих учених, студентів і курсантів. Львів, ЛДУ БЖД. – 2019. С. 76-78.
9. Полотай О, Рожко Д. Організаційно-технічні методи захисту інформації від несанкціонованого доступу. "Інформаційна безпека в сучасному суспільстві": збірник тез доповідей III Міжнародної науково-технічної конференції. – Львів: ЛДУ БЖД, 2018. – С. 52-53.

10. Ящук В. І. Онтологія наукових досліджень та методологія наукового пізнання / В.І. Ящук // Економіка в контексті глобальних змін суспільства:матеріали Міжнародної науково-практичної конференції (м. Дніпро, 18 липня 2020 р.). – Дніпро: НО «Перспектива», 2020. – 140 с. (С.100-104).
11. "DDoS, Machine Learning, Measures". // "Understanding Denial-of- Service Attacks". / , 2016. – (Taylor & Francis Group)
12. “Sdn architecture,” june 2014, accessed: 2014-09-12. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdnresources/technical reports/TR SDN ARCH 1.0 06062014.pdf
13. «Хакінг і Фрікінг» - книга Максима Левіна. Описання DDoS-атаки за допомогою катастрофи «Збірка» - А.К. Гуц, Д.Н. Лавров. Режим доступу до ресурсу: https://cyberleninka.ru/article/n/opisanie-ddos-ataki-s-pomoschyu- katastrofy-sborka/viewer
14. D-FACE: An anomaly based distributed approach for early detection of DDoS attacks and flash events. // Journal of Network and Computer Applications. – 2018.
15. Distributed Denial of Service Attack and Defense, Shui Yu, 2018.
16. Famous DDoS Attacks | The Largest DDoS Attacks Of All Time [Електронний ресурс] – Режим доступу до ресурсу: https://www.cloudflare.com/learning/ddos/famous-ddos-attacks/.
17. Hacking: Denial of Service Attacks Paperback – December 17, 2019 by Alex Wagner. https://infocom.ua/- що таке DDoS атаки, та яку мету вони переслідують
18. Ilker Ozchelik, R.R.Brooks, «Distributed Denial of Service Attacks: Real-world Detection», 2020.
19. Internet Printing Protocol/1.0: Encoding and Transport [Електронний ресурс]. – 1999. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc2565.
20. Jugal Kalita, Dhruba Kumar Bhattacharya «DDoS Attacks: Evolution,

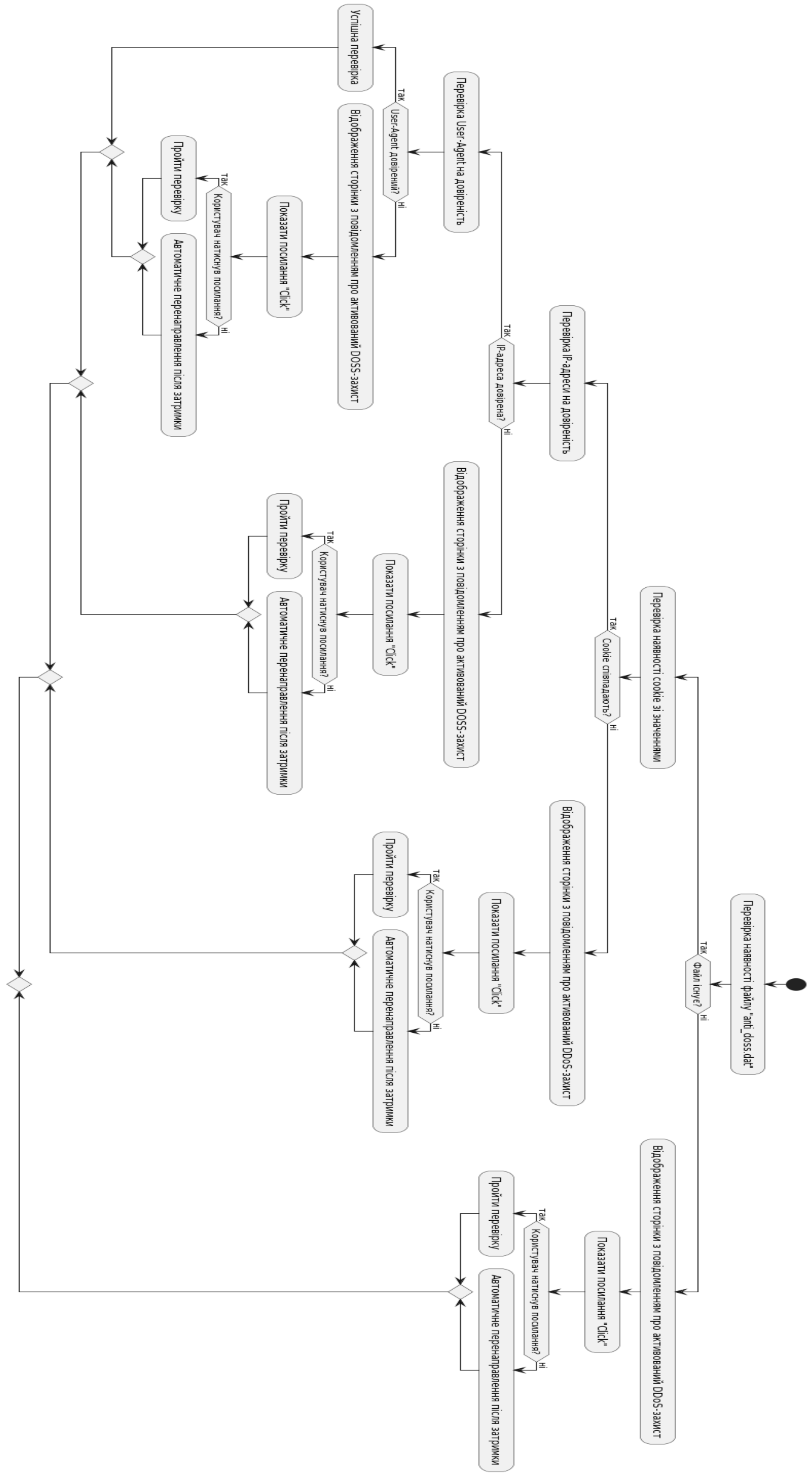
Detection, Prevention, Reaction», 2018.
21. M. Sachdeva, G. Singh, K. Kumar, and K. Singh, “Measuring impact of ddos attacks on web services,”2010.
22. M. Sachdeva, G. Singh, K. Kumar, and K. Singh, “Measuring impact of ddos attacks on web services,”2010.
23. O.Polotai, O. Belej, N. Nestor. Developing a local positioning algorithm based on the identification of objects in a wireless Wi-Fi network of the mall. IEEE
24. 16th International Conference on the Perspective Technologies and Methods in MEMS Design, MEMSTECH 2020 - Proceedings, 2020, pp. 53-58.
25. O.Polotai, O. Belej., N. Nestor, S. Panchak Developing a Model of Cloud Computing Protection System for the Internet of Things. 2020 IEEE 16th International Conference on the Perspective Technologies and Methods in MEMS Design, MEMSTECH 2020 - Proceedings, 2020, pp. 53-58.
26. O.Polotai, O.Belej, K.Kolesnyk Application of neural networks in intrusion monitoring system for wireless sensor networks. Conference on computer science and information technologies. CSIT 2020: advances in intelligent systems and computing, vol 1293, Springer, Cham. – pp.1101-1115.
27. R. Braga, E. Mota, and A. Passito, “Lightweight DDoS flooding attack detection using nox/openflow,” in Proc. 35th IEEE Conf. Local Computer Networks (LCN), 2010.
28. S. Farahmandian, M. Zamani, A. Akbarabadi, J. M. Zadeh, S. M. Mirhosseini, and S. Farahmandian, “A survey on methods to defend against DDoS attack in cloud computing,” in Proc. Recent Advances in Knowledge Engineering andSystem Science, Feb. 2013.
29. S. Shin, V. Yegneswaran, P. Porras, and G. Gu, “Avant- Guard: Scalable and Vigilant Switch Flow Management in Software-Defined Networks,” Proc. ACM SIGSAC Conf. Computer & Commun. Security, 2013, pp. 413–24.
30. The Coremelt Attack [Електронний ресурс] – Режим доступу до ресурсу: https://netsec.ethz.ch/publications/papers/studer_esorics09.pdf.
31. Wang, H., Xu, L., & Gu, G. (2015, June). FloodGuard: A DoS Attack

Prevention Extension in Software-Defined Networks. In Dependable Systems and Networks (DSN), 2015 45th Annual IEEE/IFIP International Conference on(pp. 239- 250). IEEE

32. PHP: PHP Manual - Manual [Електронний ресурс] // The PHP Group. – 2024. – Режим доступу до ресурсу: https://www.php.net/manual/en/.
33. Самойленко А. А., Єрмолаєв О. І. Комп'ютерна безпека: підручник для студентів вищих навчальних закладів. – К.: НТУУ «КПІ», 2021.
34. "Machine Learning for DDoS Detection and Mitigation: Techniques and Applications" - Автор: Yang Xiang, Xianghui Cao, Wanlei Zhou (2021).
35. "Cybersecurity: Protecting Critical Infrastructures from Cyber Attack and Cyber Warfare" - Автор: Thomas A. Johnson (2021).
36. Балабанов О. Г., Лук'янова Н. В., Пархоменко А. А. Інформаційна безпека: теорія та практика: монографія. – К.: ВПЦ «Київський університет», 2020.
37. Львівський С. В., Білоус О. О., Виноградов О. В. Захист комп'ютерних мереж: підручник. – К.: КНЕУ, 2020.
38. Косенко В. О., Курганський В. С. Кібербезпека: основні аспекти захисту від кіберзлочинів та кібертероризму. – К.: ВПЦ «Київський університет», 2020.90
39. "Securing the Cloud: Cloud Computer Security Techniques and Tactics" - Автори: Vic (J.R.) Winkler, Prashant Haldankar (2022).

ДОДАТКИ

Додаток А «Схема алгоритму програми»



Додаток Б «Лістинг коду index.html»

```
<!DOCTYPE html>
<html lang="en" xmlns="http://www.w3.org/1999/html">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="cache-control" content="private">
<meta http-equiv="cache-control" content="max-age=0, must-
revalidate">
<meta http-equiv="cache-control" content="max-age=0, proxy-
revalidate">
<meta http-equiv="expires" content="0" />
<meta http-equiv="expires" content="Tue, 01 Jan 1980 1:00:00 GMT"
/>
<meta http-equiv="pragma" content="no-cache" />
<title>Захист від DOSS атак</title>
<style type="text/css">
    .main_t {height: 100%; width: 100%; border: 0px solid black;}
    .main_t td {text-align: center;}
    .main_t td.info {height: 100%; text-align: center; vertical-
align: top; padding-top: 5em;}
    .sign { color: #ccc; margin-top: 5em;}
    .sign a { color: #ccc; }
</style>
<script>
function hrefOnclickEvent() {
    document.location = '{SERVER_URL}';
}
</script>
</head>
<body>
    <table class="main_t">
        <tr>
            <td class="info">
                <h1>Захист від DDoS активовано для вашої IP-адреси
<a href="http://cleantalk.org/blacklists/{VISITOR_IP}"
target="_new">{VISITOR_IP}</a></h1>
                <div id="human_info" style="display: none">
                    <p>Будь ласка, натисніть нижче, щоб пройти
захист,</p>
                    <button onclick="hrefOnclickEvent()">Пройти
захист</button>
```

```
                    <p>Або ви будете автоматично перенаправлені на
запитану сторінку через {REDIRECT_DELAY} секунд..</p>
                </div>
                <div id="js_info">
                    <p>Для продовження роботи з веб-сайтом, будь
ласка, переконайтеся, що JavaScript увімкнено..</p>
                </div>
            </td>
        </tr>
    </table>
    <script type="text/javascript">
        document.getElementById("js_info").style.display = "none";
        document.getElementById("human_info").style.display =
"block";
        var date = new Date();

        days = "{DAYS}";
        date.setTime(date.getTime() + (days * 24*60*60*1000));

        const ctSecure = location.protocol === "https:" ? ";
secure" : "";
        document.cookie = "{SECURE_LABEL}" + "=" +
escape("{SECURE_KEY}") + "; expires = " + date.toGMTString() + ";
path=/; samesite=lax" + ctSecure;
        document.cookie = 'ct_headless' + "=" +
encodeURIComponent(btoa('{SECURE_KEY}' +':'+ navigator.webdriver))
+ "; " + 0 + "path=/; samesite=lax";

        setTimeout(function(){
           window.location.reload(1);
        }, {REDIRECT_DELAY} * 1000);
    </script>
</body>
</html>
```

Додаток В «Лістинг коду doss-lib.php»

```
<?php
function antiDdosProtectionMain($data)
{
    $data['secure_key'] = md5($data['remote_ip'] . ':' .
$data['anti_ddos_salt']);
    if ( (antiDdosSkipUserReentry($data) && checkHeadless($data))
            || antiDdosSkipVisitorsFromTrustedAs($data)
            || antiDdosSkipVisitorsFromTrustedUa($data)
    ) {
        //встановити файли cookie безпеки
        antiDdosProtectionSetCookie($data['secure_label'],
$data['secure_key']);
        return;
    }
    //показати налагодження щодо безголових для заблокованих
відвідувачів
    if ( !empty($data['anti_ddos_debug']) &&
antiDdosSkipUserReentry($data) && !checkHeadless($data) ) {
        error_log(
            sprintf(
                'Visitor has headless mode: %s.',
                $data['remote_ip']
            )
        );
    }
    antiDdosShowDdosScreenAndRedirect($data);
}

function antiDdosProtectionSetCookie(
    $name,
    $value = '',
    $expires = 0,
    $path = '',
    $domain = '',
    $secure = null,
    $httponly = false,
    $samesite = 'Lax'
)

{
    if (headers_sent()) {
        return;
    }
```

```
    $server_https_flag = isset($_SERVER['HTTPS']) ?
$_SERVER['HTTPS'] : '';
    $server_port = isset($_SERVER['SERVER_PORT']) ?
$_SERVER['SERVER_PORT'] : '';

    $secure = ! is_null($secure)
        ? $secure
        : ! in_array($server_https_flag, ['off', '']) ||
$server_port === 443;

    // For PHP 7.3+ and above
    if ( version_compare(phpversion(), '7.3.0', '>=') ) {
        $params = array(
            'expires' => $expires,
            'path' => $path,
            'domain' => $domain,
            'secure' => $secure,
            'httponly' => $httponly,
        );

        if ($samesite) {
            $params['samesite'] = $samesite;
        }

        setcookie($name, $value, $params);
        // For PHP 5.6 - 7.2
    } else {
        setcookie($name, $value, $expires, $path, $domain,
$secure, $httponly);
    }
}

function antiDdosCheckDatFileExist()
{
    return file_exists('anti_doss_protection_fire.dat');
}

function antiDdosSkipUserReentry($data)
{
    return isset($_COOKIE[$data['secure_label']]) &&
$_COOKIE[$data['secure_label']] == $data['secure_key'];
}
```

```
function antiDdosSkipVisitorsFromTrustedAs($data)
{
    if (!function_exists('geoip_org_by_name')) {
        return false;
    }

    // Список надійних автономних систем.
    $notRatedAs =
[13238,15169,8075,10310,36647,13335,2635,32934,38365,55967,

16509,2559,19500,47764,17012,1449,43247,32734,15768,33512,18730,30
148];

    $visitorOrg = geoip_org_by_name($data['remote_ip']);
    if ($visitorOrg !== false && preg_match("/^AS(\d+)\s/",
$visitorOrg, $matches)) {
        foreach ($notRatedAs as $asn) {
            if ($asn == $matches[1]) {
                if ($data['anti_ddos_debug']) {
                    error_log(sprintf('Skip antiddos protection
for %s, because it\'s trusted AS%d.', $data['remote_ip'], $asn));
                }

                return true;
            }
        }
    }

    return false;
}

function antiDdosSkipVisitorsFromTrustedUa($data)
{
    if (!$data['skip_not_rated_ua']) {
        return false;
    }

    if (!isset($_SERVER['HTTP_USER_AGENT'])) {
        return false;
    }

    require "not_rated_ua.php";
    global $notRatedUa;
    if (count($notRatedUa) > 0) {
```

```
        foreach ($notRatedUa as $ua) {
            if (preg_match("/^$ua$/",
$_SERVER['HTTP_USER_AGENT'])) {
                if ($data['anti_ddos_debug']) {
                    error_log(sprintf('Skip antiddos protection
for %s, because it\'s trusted User-Agent %s.', $data['remote_ip'],
$ua));
                }
                return true;
            }
        }
    }

    return false;
}

function antiDdosShowDdosScreenAndRedirect($data)
{
    $html_file = file_get_contents(dirname(__FILE__) . '/anti-
doss.html');

    http_response_code(403);

    $code = str_replace('{VISITOR_IP}', $data['remote_ip'],
$html_file);
    $code = str_replace('{REDIRECT_DELAY}',
$data['redirect_delay'], $code);
    $code = str_replace('{DAYS}', $data['secure_cookie_days'],
$code);
    $code = str_replace('{SECURE_LABEL}', $data['secure_label'],
$code);
    $code = str_replace('{SECURE_KEY}', $data['secure_key'],
$code);
    $code = str_replace('{SERVER_URL}', $data['server_url'],
$code);

    echo ($code);

    if ( $data['anti_ddos_debug'] ) {
        error_log(
            sprintf(
                'Blacklisted IP, drop connection %s to %s.',
                $data['remote_ip'],
                $_SERVER['REQUEST_URI']
```

```
            )
        );
    }

    exit;
}

function checkRequirements()
{
    if (version_compare(phpversion(), '5.6', '<')) {
        return false;
    }

    return true;
}

function checkHeadless($data)
{
    if ( empty($data['test_headless']) ) {
        return true;
    }


    if ( isset($_COOKIE['ct_headless']) ) {
        $headless = explode(':',
base64_decode($_COOKIE['ct_headless']));
        if ( isset($headless[0], $headless[1])
            && $headless[0] === $data['secure_key']
            && $headless[1] === 'false' ) {
            return true;
        }
    }

    return false;
}
```

Додаток Г «Лістинг коду anti-doss.php»

```
<?php
require 'src/doss-lib.php';

if (!checkRequirements()) {
    return;
}

if (!isset($_SERVER['REMOTE_ADDR'])) {
    return;
}

$data = [
    'anti_ddos_protection_enable' => true, // Перейти в
контрольний стан AntiDOSS.
    'anti_ddos_debug' => true,             // Активуйте оператори
налагодження.
    'skip_not_rated_ua' => false,          // Тестуйте
відвідувачів за списком довірених UserAgent.
    'secure_cookie_days' => 180,           // Днів для
використання безпечних файлів cookie.
    'redirect_delay' => 5,                 // Затримка в секундах
перед переспрямуванням на оригінальну URL-адресу.
    'remote_ip' => $_SERVER['REMOTE_ADDR'],
    'secure_label' => 'ct_anti_ddos_key',
    'test_headless' => true,               // Блокувати
відвідувачів з режимом безголового (наприклад, Selenium)
    'server_url' => isset($_SERVER['HTTPS']) ? 'https://' .
$_SERVER['HTTP_HOST'] : 'http://' . $_SERVER['HTTP_HOST'],

    // Секретна сіль ключа для уникнення копіювання/вставки Cookie
між відвідувачами.
    // УВАГА!!!
    // ВИ МУСИТЕ ЗГЕНЕРУВАТИ НОВУ $anti_ddos_salt ПЕРЕД
ВИКОРИСТАННЯМ НА СВОЄМУ ВЛАСНОМУ САЙТІ.
    // УВАГА!!!

    'anti_ddos_salt' => '4xU9mn2X7iPZpeW2'
];

if ($data['anti_ddos_protection_enable'] ||
antiDdosCheckDatFileExist()) {
    antiDdosProtectionMain($data);
}
```