МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Комплексний захист веб-ресурсу від DDOS атак»

Завідувач випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Ободяк В.К.**

Студента групи КБ – 61 **Дузь В.І.**

СУМИ 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедрою Довбиш А.С.

"_____" _____2020 г.

ЗАВДАННЯ
до випускної роботи

Студента четвертого курсу, групи КБ-61 спеціальності "Кібербезпека" денної форми навчання Дузя Владислава Івановича.

Тема: " Комплексний захист веб-ресурсу від DDOS атак "

Затверджена наказом по СумДУ

№_____ от _____2020 г.

Зміст пояснювальної записки: 1) аналітичний огляд розподілених атак на відмову в обслуговування та методів захисту; 2) постановка завдання та формування завдань дослідження; 3) дослідження алгоритмів фільтрації небажаного трафіку; 4) тестування системи комплексного захисту від DDoS атак; 5) характеристика конфігурацій мережі для ефективного захисту від DDoS атак; 6) аналіз результатів розробки.

Дата видачі завдання "_____"_____2020 г.

Керівник випускної роботи _____ Ободяк В.К.

Завдання приняв до виконання _____ Дузь В.І.

РЕФЕРАТ

Записка: 49 сторінок, 12 рисунків, 1 додаток, 14 джерел.

Об'єкт дослідження - процес фільтрації шкідливого трафіку.

Мета роботи – розробка комплексного захисту веб-ресурсу від DDoS атак.

Методи дослідження - моделі та методи захисту веб-ресурсів.

Результати - розроблена модель та конфігурації надійної структури веб-сервера; розглянуті і програмно реалізовані алгоритми фільтрації шкідливого трафіку; працездатність системи перевірена на прикладі захисту веб-ресурсу від реалізованої DDoS атаки.

АТАКА ВІДМОВА В ОБСЛУГОВУВАННІ, DDOS, БОТНЕТ, АЛГОРИТМИ ФІЛЬТРАЦІЇ ШКІДЛИВОГО ТРАФІКУ, МІЖМЕРЕЖЕВИЙ ЕКРАН

ЗМІСТ

ВСТУП

В останні десятиліття відбувся стрімкий розвиток інформаційних технологій, через що їх вплив на сфери діяльності людей, де фігурує обробка та накопичення інформації, є максимальним. Зараз велика кількість баз даних, та ресурсів, які знаходяться в мережі інтернет містять інформацію, в якій кожний день потребують тисячі людей. Звідси і виникає потреба в безперебійному доступу до цієї інформації, та захисту систем які її зберігають, що в свою чергу викликає попит на захист сервері, веб-ресурсів, мереж від DDoS атак. Для цього можна використовувати процес фільтрації шкідливого трафіку.

1 АНАЛІЗ ПРОБЛЕМИ. ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Аналітичний огляд розподілених атак на відмову

Атака типу «відмова в обслуговуванні» (DoS) - це спроба завдати шкоди, зробивши недоступною цільову систему для звичайних кінцевих користувачів, наприклад веб-сайт або веб-додаток. Зазвичай зловмисники генерують велику кількість пакетів або запитів, які зрештою перевантажують роботу цільової системи. Для здійснення атаки типу «розподілена відмова в обслуговуванні» (DDoS) зловмисник використовує безліч зламаних або контрольованих джерел [1, 2].

На початку розвитку мережевих технологій, DDoS був інструментом тестування пропускної здатності. Але досить швидко зловмисники знайшли можливість використання даної атаки в своїх корисних цілях. Атака типу «відмова в обслуговуванні» є в наборі інструментів зловмисників уже протягом двадцяти років, і вона досі зростає в популярності і розвивається. Зараз даний тип атаки використовуються для отримання вигоди різними способами: шантаж, припинення функціонування основних вузлів мережі; припинення працездатності веб-сервісів; в бізнес або політичній конкуренції тощо. Вплив може варіюватися від незначного роздратування адміністратора мережі та незначних порушених сервісів до повного припинення функціонування цілих веб-сайтів, додатків або навіть цілого бізнесу, який працює в автономному режимі. Це у свою чергу може призвести до втрати репутації або значних фінансових втрат. Також даний тип атак може бути відволікаючим фактором, для здійснення складніших атак, для викрадення конференційної інформації. Можна виділити основні мотиви зловмисників які використовують DDoS [8]:

- Конкуренція.

На даний момент досить популярною є послуга проведення DDoS на замовлення. Тобто, при конкуренції, фірма, яка не погоджується з конкурентами,

просто формує завдання для хакера паралізувати систему, з якою працюють конкуренти, або паралізувати роботу зовнішні або внутрішніх ресурсів конкурентоспроможної компанії. В результаті чого, організовується DDoS атака на визначений термін і з визначеною силою.

- Шахрайство.

Хакери самостійно організовують DDoS атаки з власної мережі заражених комп'ютерів, які дозволяють блокувати роботу невеликих та середніх системи. Якщо користувач не встановив захист від DDoS атаки, то хакер може повною мірою паралізувати роботу системи, і шантажувати на певну суму для розблокування. Часто, звичайні користувачі, погоджуються на умови хакерів. Це обумовлене тим, що просто цінність даних вища, чим сума, вказана хакером.

- Розвиток або забава.

У зв'язку з тим, що в останній час все більше людей цікавлять DDoS атаки, та, багато початківців хакерів здійснюють такі атаки просто ради забави.

Отже, можна сказати, що основними цілями атаки є створення таких умов, при яких користувач з необхідними правами доступу, не має змоги отримати ресурси системи, або мають з цим трудності. Зі сторони мережі DDoS виглядає як різке збільшення трафіку.

В загалом виділяється дві методики, які найбільш часто використовуються зловмисниками:

- руйнівні атаки – призначені для того щоб сегмент мережі став цілком недоступним: зависає, знищується, видаляється операційна система. Цей тип атаки стає доступним через вразливості, які є в програмному забезпеченні;
- атака на ресурси системи – в цій атаці велика кількість пустих або неправильних запитів відправляється на вузол мережі або веб-додатку, що тягне за собою значне зменшення пропускної здатності мережі або продуктивності системи та мережевого обладнання.

Основні види DDoS атак [8]:

1. Атака на перевищення ліміту системних ресурсів.

Дії зловмисників націлені на вичерпання системних ресурсів такі як оперативна і фізична пам'ять, процесорний час тощо.

2. Недостатня перевірка даних користувача.

Недостатня перевірка даних користувача може призвести до нескінченного або тривалого циклу їх обробки, що призводить до підвищеного та тривалого споживання процесорних ресурсів або виділенню великих обсягів пам'яті, аж до її вичерпання.

3. Атаки другого роду.

Це атаки, які націлені на ділянки мережі в яких неправильно ввімкнено або налаштовано системний захист, тим самим, це призводить до недоступності конкретних ресурсів.

4. HTTP-флуд.

Нападник відсилає невеликі http-пакети, які заставляють сервер у свою чергу відсилати пакети-відповіді, розміри яких значно більші. Тим самим зловмисник має великі шанси наситити смугу пропускання жертви безглуздими пакетами що викликає відмову в роботі сервісу. Для того, щоб відповідні пакети не входили в опис заборони в обслуговуванні, нападник замінює свій мережевий адрес на легітимні у вузлу мережі.

5. ICMP-флуд (Smurf-атака).

Цей тип атаки є одним з найнебезпечніших. У ньому по широкомовній адресі зловмисник відправляє підроблений ICMP- пакет, в якому адреса відправника, змінюється на адресу жертви. Усі вузли, які отримали такий пакет, присилають відповідь на цей запит. Для такого виду атаки зазвичай використовують велику мережу, щоб у комп'ютера-жертви не було жодних шансів. Таким чином, запит, відправлений через мережу в 1000 комп'ютерів буде посилений в 1000 разів.

1. UDP флуд (атака Fraggle).

Цей тип атаки є аналогом ICMP флуду, але замість ICMP пакетів використовуються UDP пакети. На сьомий порт жертви відправляються echo-

команди по широкомовному запиту. Після чого підміняється ip-адреса зловмисника на ip-адресу жертви, яка отримує безліч повідомлень у відповідь, що призводить до насичення смуги пропускання і відмови в обслуговуванні жертви.

2. SYN-флуд.

Цей вид атаки ґрунтований на спробі запуску великого числа одночасних TCP- з'єднань через відправлення SYN-пакету з зворотним адресом якого не існує. Після декількох спроб відіслати у відповідь ACK-пакет на недоступну адресу більшість операційних систем вставляє невстановлене з'єднання в чергу. І тільки після n-ої спроби закривають з'єднання. Оскільки потік ACK-пакетів дуже великий, незабаром черга виявляється переповнене, і ядро дає відмову на спробу відкрити нове з’єднання. Схема даної атаки показано на рисунку 1.1.

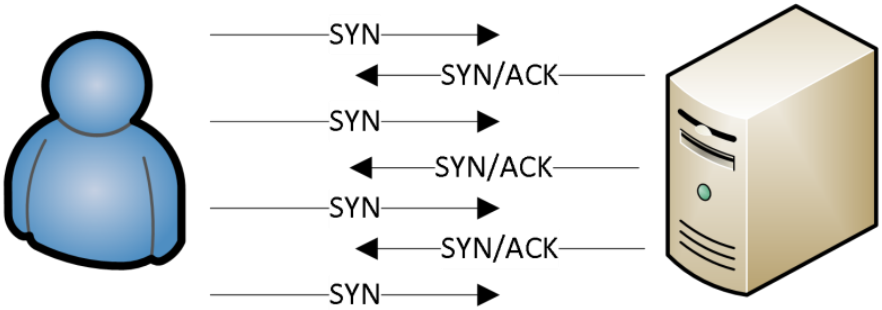


Рисунок 1.1 — SYN-флуд

3. Відправлення важких «пакетів».

Зловмисник посилає пакети серверу, які не насичують смугу пропускання, а витрачають увесь його процесорний час. Відповідно, в системі може пройти збій і легальні користувачі не зможуть отримати доступ до необхідних ресурсів.

4. Переповнювання сервера лог-файлами.

При неправильній ротації лог-файлів і неправильно встановленій системі квотування зловмисник може відправляти великі за об'ємом пакети, які незабаром займуть усе вільне місце на жорсткому диску сервера.

5. Помилки програмного коду.

Досвідчені організатори DDoS атак, повністю розібравшись в структурі жертви, пишуть програми-експлоїти, які дозволяють атакувати складні системи комерційних підприємств і організацій. В основному, це помилки в програмному коді, які дозволяють виконувати неприпустимій інструкції або винятковій ситуації, яка може привести до аварійного завершення служби.

6. Недоліки в програмному коді.

Зловмисники шукають помилки в програмному коді яких-небудь програм або операційної системи й примушують їх обробляти виняткові ситуації, які вони обробляти не вміють, що призводить до падіння ядра або краху усієї системи в цілому.

Зараз, здебільшого використовується комбінований з декількох видів DDoS атак, шо робить протидію проти них складнішою.

За даними статистики [9] найпопулярніший тип атаки HTTP-Flood (80%). Зараз кіберзлочинці мають безліч технології для проведення цієї атак, які постійно розвиваються. У 54% дана атака комп'ютери ботнету посилають данні конкретній сторінці веб-ресурсу; наступне місце 22% атака на поля авторизації; на третьому місці 13% атака спрямована на велику кількість завантаження файлів з веб-ресурсу. І лише в одному випадку з 10 проводяться сладніші атаки, коли зловмисники намагаються замаскувати дії ботів під поведінку справжніх користувачів.

Друге місце (10%) за популярністю це атака типу UDP-Flood. Зловмисники, які здійснюють такі атаки, покладаються на величезну кількість пакетів, тобто генерують за допомогою ботнету велику за кількістю пакетів, але невеликі за розміром (до 64 байта) пустих пакетів.

На наступних місцях атаки типу SYN-Flood а ICMP-Flood відповідно. Описана статистика зображена на діаграмі - рис. 1.2 [9].

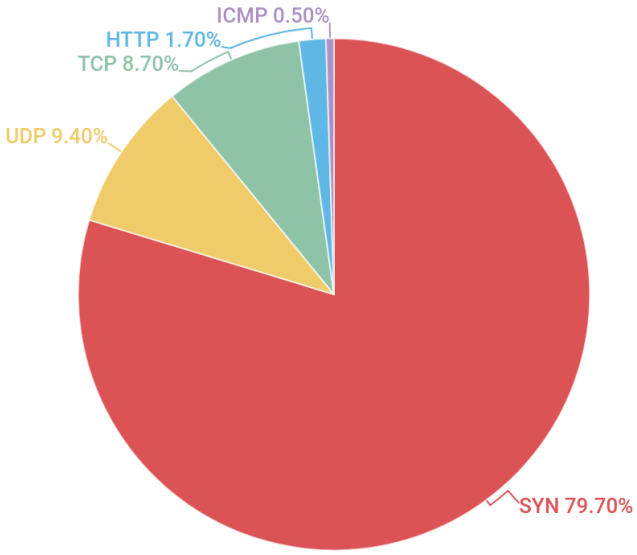


Рисунок 1.2 – Типи DDoS атак за популярністю

Зазвичай мережа зловмисника, з якої відбувається DDoS атака, має три рівні (рис 1.3). Назва такої мережі «кластер DDoS». Верхній рівень кластера складається с декількох комп’ютерів, з яких відбувається початок атаки та надалі іде координація дій інших учасників. Другий рівень – складається з десятків комп'ютери, які вже надають сигнали про атаку на наступний рівень кластера. На третьому рівні — зазвичай DDoS-зловмисники покладаються на ботнети — колекції мережі інфікованих зловмисними програмами, які централізовано контролюються. Ці заражені кінцеві точки, як правило, є комп'ютерами та серверами, але все частіше є IoT та мобільними пристроями. Зловмисники збирають ці системи шляхом виявлення вразливих систем, якими вони можуть потенційно заразитися за допомогою фішинг-атак, зловмисних атак та інших методів масового зараження. Все частіше зловмисники також орендують цих ботнетів у тих, хто їх побудував [5].

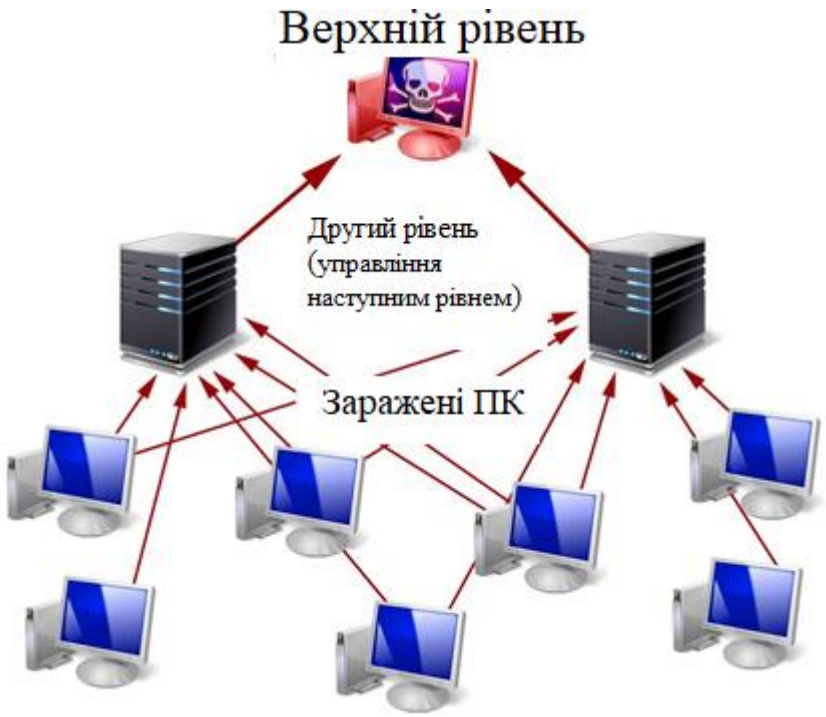


Рис. 1.3 – Структура ботнет мережі

Залучення до ботнету зазвичай відбувається через встановлення на комп'ютер жертви програму, яка не виявляється користувачем в щоденній роботі. Найчастіше зловмисники здійснюють ці дії через:

• зараження комп'ютера через вразливість (помилки в браузерах, поштових клієнтів, програмах перегляду документів, зображень, відео);

• незнання або необачність користувачів, коли небезпечна програма маскується під безпечне;

• несанкціонованого доступу до комп'ютера користувача;

• повний перебір пароля (брутфорс) для отримання доступу прав адміністратора мережі, переважно використовується в локальних мережах.

Через стрімкий розвиток та ріст ботнет мереж за останні роки, складність захисту від зловмисників зростає. Це пояснюється тип що, через використання цієї технології, знаходження організаторів атаки стає досить складно. Через це, ці напади стають більш популярні, та зазвичай проводять їх через орендовані ботнети. Очікується, що ця тенденція продовжиться.

У багатьох країнах реалізація DDoS -атаки є злочином, але досить рідко виходить впіймати організаторів, а ще рідше виконавців. В наші дні організацією DDoS атак займаються не одна людина, а добре організовані зловмисна групи. Через їх організаційну та географічну віддаленість, їм вдається вдало протидіяти правоохоронним органам.

1.2 Основні принципи, концепції та методи захисту від DDoS атак

Методи протидії DDoS атакам можна розділити на пасивні і активні, а також на превентивні і реакційні. Найбільш популярні методи захисту [3]:

1. Відвертання.

Необхідно проводити профілактику причин, які призводять до необхідності різним особам потенційно зробити DDoS атаку. Особиста неприязнь, конкуренція, релігійні, політичні або інші розбіжності, а також багато інших чинників можуть стати причиною такої атаки. Якщо вчасно усунути причини таких атак і зробити відповідні висновки, то надалі вдасться уникнути повторення ситуації. Цей метод націлений на захист від практично будь-яких DDoS атак, оскільки є управлінським, а не технічним рішенням.

2. Заходи у відповідь.

Необхідно проводити активні заходи протидії на джерела або організатора атак, використовуючи як технічні, так і організаційно-правові методи. Деякі фірми надають сервіс пошуку організатора атак, який дозволяє вичислити не лише людину, що проводить атаку, але і замовника цієї атаки.

3. Спеціалізоване програмне й апаратне забезпечення.

Зараз багато виробників програмного й апаратного забезпечення пропонують готові рішення для захисту від DDoS- атак. Таке програмне і апаратне забезпечення може виглядати як невеликий сервер, який дозволяє захиститися від слабких і середніх DDoS атак, націлених на малий і середній бізнес, так і цілий комплекс, що дозволяє захистити від серйозних атак великі підприємства і держустанови.

4. Фільтрація.

Фільтрація і блокування трафіку, вихідного від машин нападників, дозволяє понизити або зовсім загасити атаку. При використанні цього методу, трафік, що входить, фільтрується відповідно до тих або інших правил, заданих при установці фільтрів. Можна виділити два способи фільтрації : маршрутизація за списками ACL і використання міжмережевих екранів. Використання списків ACL дозволяє фільтрувати другорядні протоколи, не зачіпаючи при цьому протоколи TCP і не уповільнюючи швидкість роботи користувачів з ресурсом. Проте, при використанні зловмисниками первинних запитів або ботнета, цей спосіб фільтрації виявиться неефективним. Міжмережеві екрани є украй ефективним способом захисту від DDoS атак, проте вони застосовні виключно для захисту приватних мереж.

5. Зворотній DDoS.

Перенаправлення трафіку на того, що атакує при достатніх серверних потужностях дозволяє не лише успішно здолати атаку, але і вивести з ладу устаткування того, що атакує. Цей тип захисту неможливо застосувати при помилках в програмному коді операційних систем, системних служб або веб-застосувань.

6. Усунення вразливостей.

Цей тип захисту націлений на усунення помилок в тих або інших системах або службах (виправлення експлойтів, установка оновлень на операційну систему і тощо). Відповідно, такий метод захисту не працює проти флуд-атак, для яких "вразливістю" є кінцівка тих або інших системних ресурсів. Нарощування ресурсів не дає абсолютного захисту, але дозволяє використати інші види захисту від DDoS атак. Маючи сучасне програмне й апаратне забезпечення, ви можете вдало впоратися з DDoS атакою, спрямованою на кінцівку системних ресурсів.

7. Побудова розподілених систем.

Побудова розподілених і дублюючих систем дозволяє обслуговувати користувачів, навіть якщо деякі вузли стають недоступні через DDoS атаку. Рекомендується будувати розподілені системи, використовуючи не лише різне

мережеве або серверне устаткування, але і фізично розносити сервіси по різним дата-центрам. Також можлива установка дублюючої системи (критичних вузлів, резервних копій) на території інших держав, що дозволить зберегти важливу інформацію навіть при пожежі в дата-центрі або стихійному лихові. Розподілені системи дозволяють впоратися практично з будь-яким типом атак при правильному архітектурному проектуванні.

8. Ухилення.

Виведення безпосередньої мети атаки (ip-адреса або доменне ім'я) від інших ресурсів, які також можуть піддаватися атаки разом з метою. Інакше кажучи, необхідно розділити ресурси, що атакуються, і інші робочі ресурси, які розташовані на одному майданчику. Оптимальним є рішення по розділенню на зовнішні і внутрішні ресурси і виведення зовнішніх ресурсів на інше мережеве устаткування, інший дата-центр або навіть територію іншої держави. Це дозволить зберегти внутрішню IT-структуру навіть при найінтенсивнішій DDoS атаці на зовнішні ресурси.

9. Моніторинг.

Установка системи моніторингу і сповіщення, яка дозволить вичислити DDoS атаку за певними критеріями. Моніторинг безпосередньо не може захистити систему, що атакується, але дозволяє вчасно зреагувати і вжити відповідні заходи.

10. Придбання сервісу по захисту від DDoS атак.

Зараз багато великих компаній пропонують надання як постійного, так і тимчасового сервісу по захисту від DDoS атак. Цей метод дозволяє захиститися від багатьох типів DDoS атак, використовуючи цілий комплекс механізмів фільтрації небажаного трафіку до атакуючих серверів [3, 4].

1.3 Постановка задачі

Наведений вище огляд причин та методів які використовують зловмисники, вказує на перспективність створення комплексної системи захисту від DDoS атака.

У дипломній роботі необхідно розробити систему комплексного захисту від DDoS атак, для підтримки захисту веб-ресурсів різного масштабу, та призначення.

При цьому необхідно виконати такі завдання:

1) описати методи захисту мережі від DDOS атак;

2) розробити ефективні конфігурації веб-серверу, для протидії DDoS атак

3) модифікувати та комбінувати різні методи захисту від DDOS атак в один комплекс функціональної ефективності;

4) перевірити ефективність розробленої системи захисту на реальній мережі, реалізувавши тестову DDOS атаку.

2 ДОСЛІДЖЕННЯ АЛГОРИТМІВ ФІЛЬТРАЦІЇ НЕБАЖАНОГО ТРАФІКУ

2.1 Алгоритм захисту від DDoS атак заснований на IP фільтрування

Даний алгоритм є практичною схемою для захисту від розподілених атак типу відмова в обслуговуванні (DDoS) атак, заснованих на фільтрації IP-адрес джерел [11].

В цьому методі граничний маршрутизатор зберігає історію всіх правомірних адрес IP, які раніше з'являлися в мережі. Коли граничний маршрутизатор перевантажений, ця історія використовується для рішення, чи слід приймати пакети від конкретного IP. На відміну від інших алгоритмів для захисту від DDoS атак, цей метод працює добре при високорівневій розподіленій DDoS атаці, тобто з великої кількості джерел.

Основний принцип цього алгоритму, який дає можливість використовувати в захисті від DDoS атак полягає в тому, що трафік DDoS атаки прагне використовувати підроблені випадкові адреси джерел, щоб замаскувати свою істинну адресу. Таким чином, залишається, розрізнити легальну IP-адресу від випадкової злочинної IP-адреси.

Алгоритм використовує механізм під назвою History-based IP Filtering (HIF), граничний маршрутизатор приймає пакети згідно з попередньо побудованої бази даних IP-адрес. База даних IP-адрес має бути заснована на основі попередньої історії маршрутизатора [11]. Існує декілька евристичних методів, щоб зробити базу даних IP-адрес точної і надійної.

Схема складається з трьох частин. Вона є дуже надійною в захисті від розподілених атак типу відмова в обслуговуванні. Цей механізм не потребує глобальну співпрацю з іншими веб-ресурсами. Дана методика може бути застосована до широкої різноманітності видів трафіку, і не вимагає особливого налаштування вручну.

На ранній стадії DDoS атак, трафік обмежується за певним типом для прикладу, Smurf-атаки використовують ICMP пакети. Таким чином, можна легко блокувати атаку трафіку згідно з типу протоколу. На жаль, останні дослідження показують, що інструменти DDoS атак постійно розвиваються, та здатні створювати пакети з підробленими випадковими адресами джерела, вихідного порту і навіть генерувати корисне навантаження пакетів [13]. Це значно ускладнює процес фільтрації та вимагає розширеного трафіку методики моделювання. Цей метод дає практичне розв’язання цієї проблеми, відрізняючи хороші та погані пакети порівнюючи їх з попередньою історією.

На відміну від алгоритмів фільтрації IP-джерела, які намагаються охарактеризувати атакуючий трафік, можна схарактеризувати замість цього звичайний трафік. Ця методика відноситься до типу виявлення аномалій.

Дослідження показали, що більшість всіх IP-адрес нормального трафіку відправляли запити раніше. На основі статистичного аналізу інтернет-трафіку, зібрані в мережі середнього розміру, було виявлено, що велика частина IP-адрес, що входять в інтернет-трафік постійно з'являються на основі щоденних спостережень [10]. З цього можна зробити висновок, що надійна особливість ідентифікації нормального трафіку це те, що він регулярно з'являвся на веб-сайті.

Фундаментальною проблемою безпеки для IP-мереж є те, що немає схеми аутентифікації джерела IP пакета. Якщо маршрутизатор або веб-сервер мав би можливість визначити, які адреси є законними, проблема аутентифікації могла бути вирішена. Щоб відрізняти хороші пакети від поганих пакетів, маршрутизатор або веб-сервер повинен вчитися з історії попередніх здійснюваних підключень до мережі. Даний метод полягає в тому, щоб зробити запис всіх IP-адрес попередніх успішних мережевих з'єднань, та зібрати базу даних IP-адрес. Тоді, коли наша мережа або веб-сайт відчувають високий рівень скупчення, ми можемо відмовитися від пакетів чию адресу джерела не з'являється в нашій базі даних IP-адрес. У разі DDoS атаки, це означає, що ми маємо високу точність фільтрації пакетів атаки.

В основному є дві ключові ролі техніки HIF. Перша полягає в розробці правил розрізнення законних пакетів від шкідливих пакетів шляхом перевірки IP-адреса вхідного пакета в базі даних IP-адрес (IP Address Database (IAD)). Друге, як використовувати хеш способи побудови ефективних пошук IP таблиці.

Нехай весь трафік з однією IP-адресою джерела, характеризується як одна IP-адреса. Нехай $S_i = \{s^i_1, s^i_2, s^i_3, s^i_4 \dots s^i_n,\}$, описує колекцію всіх законних IP-адрес, які з'явились у мережі на дату i, де $| S_i | = n_i$. Нехай $F^k=\{f_1, f_2, f_3, f_4, \dots, f_m,\}$ описує колекцію всіх частих законних IP-адрес від дати l до дати k де $| F^k |=m$. цьому розділі ми надаємо два альтернативні правила для визначення «частих». Нехай $A=\{a_1, a_2, a_3, a_4, \dots, a_n\}$ описує IP-адреса, які використаються в розподіленій атаці на відмову в службі. Оскільки існує стабільна група IP-адрес, які регулярно відвідують мережу, а DDoS атаки використовують випадкові підроблені IP-адреси, як ми проаналізували в попередньому розділі, для спостереження трафіку k днів наступне співвідношення:

$$|S_1 \cup S_2 \cup S_3 \dots \cup S_k| < \sum_{i=1}^{k} \quad n_i \ll |A| (2.1)$$

Очевидно, що $(F^k \subseteq S_1 \cup S_2 \cup S_3 \dots \cup S_k)$. Ми прагнемо проаналізувати розподіл IP-адреси в межах $S_1 \cup S_2 \cup S_3 \dots \cup S_k$ та використання статистичного підходу для розробки порогу для визначення частого збору користувачів F. Таким чином $P_{normal} = \frac{|F \cap S_i|}{|S_i|}$ являє собою відсоток нормальних IP-потоків, прийнятих на дату j (j> k) і $P_{ddos} = \frac{|F \cap A|}{|A|}$ представляє відсоток допущених IP-адрес атаки. В ідеалі ми хочемо, щоб P_{normal} був l, а P_{ddos} був 0.

Визначаємо базу даних IP-адрес (IAD) як цілу колекцію IP-адрес, що з'явились у попередньому періоді експертизи, наприклад, один місяць. Для фільтрування IP на основі історії (HIF) дуже важливо отримати точну та ефективну базу даних IP-адрес (IAD). Існує два різні правила, щоб визначити, що часто є IP-адресою. Перше правило вважає IP-адресу частою залежно від

кількості днів, коли вона з’явилася. Нехай $p_1(d)$ представляють колекцію унікальних IP-адрес, які з’явилися щонайменше D дні.

Нехай $f_1(d)$ являє собою відсоток хорошого трафіку, який проходить при використанні $p_1(d)$ як IAD. Друге правило - кількість пакетів на IP-адреси. Нехай $p_2(d)$ являє собою сукупність унікальних IP-адрес, що мають принаймні пакети u. Нехай $f_2(u)$ являє собою відсоток хорошого трафіку, який проходить при використанні $p_2(u)$ в якості IAD.

На практиці ми хочемо щоб $|p_1(d)|$ і $|p_2(u)|$ невеликими, щоб зменшити потребу в пам'яті для збереження IAD, і ми хочемо $f_1(d)$ і $f_2(d)$ бути великими, щоб захистити законний трафік. У цьому алгоритмі залучені два параметри проектування: кількість днів (d) та кількість пакетів на IP-адресу (u). Ми можемо налаштувати параметри відповідно до різних мережевих умов та отримати більш точну та ефективну IAD, комбінуючи всі ці два правила, які можуть бути показані як:

$$F_c = p_1(d) \cap p_2(u) \quad (2.2)$$

Після того, як ми заповнили IAD, важливо швидко отримувати Процес пошуку IP-адреси, особливо коли |F| великий. Для цього можна використовувати хеш функцію для створення єдиної хеш-таблиці.

2.2 Алгоритм упорядкування правил фільтрації мережевої трафіку

Міжмережеві екрани (МЕ) - невід'ємний елемент забезпечення мережевої безпеки. Застосування МЕ при побудові систем захисту інформації є не тільки частою практикою, а й обов'язковою вимогою ряду нормативних документів в області інформаційної безпеки. Політика міжмережевого екранування (визначення того, які мережеві потоки повинні бути пропущені, а які заблоковані) реалізується в МЕ набором правил фільтрації мережевого трафіку.

Структура представлення наборів правил фільтрації переважної більшості сучасних МЕ є лінійний список. Для кожного пакета, що проходить через МЕ

здійснюється послідовний перебір правил з даного лінійного списку. Після знаходження першого правила фільтрації, яке повністю задовольняє відповідний мережевий пакет, перебір наступних в списку правил зупиняється. Таким чином, чим вище за списком буде стояти відповідне правило фільтрації, тим швидше МЕ прийме рішення про обробку мережевого пакета. Очевидно, що чим вище за списком будуть розташовуватися найбільш часто спрацьовані правила фільтрації, тим менше ресурсів МЕ буде витрачатися на послідовний перебір лінійного списку правил. У свою чергу, від інтенсивності використання обчислювальних ресурсів МЕ багато в чому залежить його пропускна здатність.

Запропонований алгоритм впорядкування правил фільтрації мережевого трафіку, дозволяє впорядковувати правила фільтрації в наборах правил за частотою їх спрацьовування. Мета помістити найбільш часто використовувані правила на вершину списку, щоб мінімізувати витрачання ресурсів системи на послідовну перевірку великої кількості правил. Шляхом оптимізації даного процесу можлива істотна економія машинного часу, і зниження навантаження на пристрої, що здійснює функції брандмауера.

Основу алгоритму становить технологія пасивного аналізу мережевого трафіку. Вона дозволяє побудувати статистичну модель трафіку який проходить через МЕ і на її основі розрахувати частоту спрацьовування кожного правила фільтрації з набору правил. Блок-схема алгоритму представлена на рис. 2.1.

На вхід алгоритму подається мережевий трафік, який перетворюється в масив мережевих пакетів P, з якого потім відновлюється інформація про TCP-з’єднання або UDP-з’єднання. Ця інформація зберігається в окремому масиві даних C. Для кожного з'єднання в даному масиві зберігається наступна інформація: IP-адреса клієнта, номер порту клієнта, IP-адреса сервера, номер порту сервера, який використовується протокол транспортного рівня і кількість переданих в рамках з'єднання мережевих пакетів. На основі масиву C формується зведений масив мережевих з'єднань C* шляхом об'єднання записів з однаковими значеннями полів «IP-адреса клієнта», «IP-адреса сервера», «№ порту сервера» і

«Транспортний протокол». Даний масив потім сортується за спаданням значення поля «Кількість пакетів».

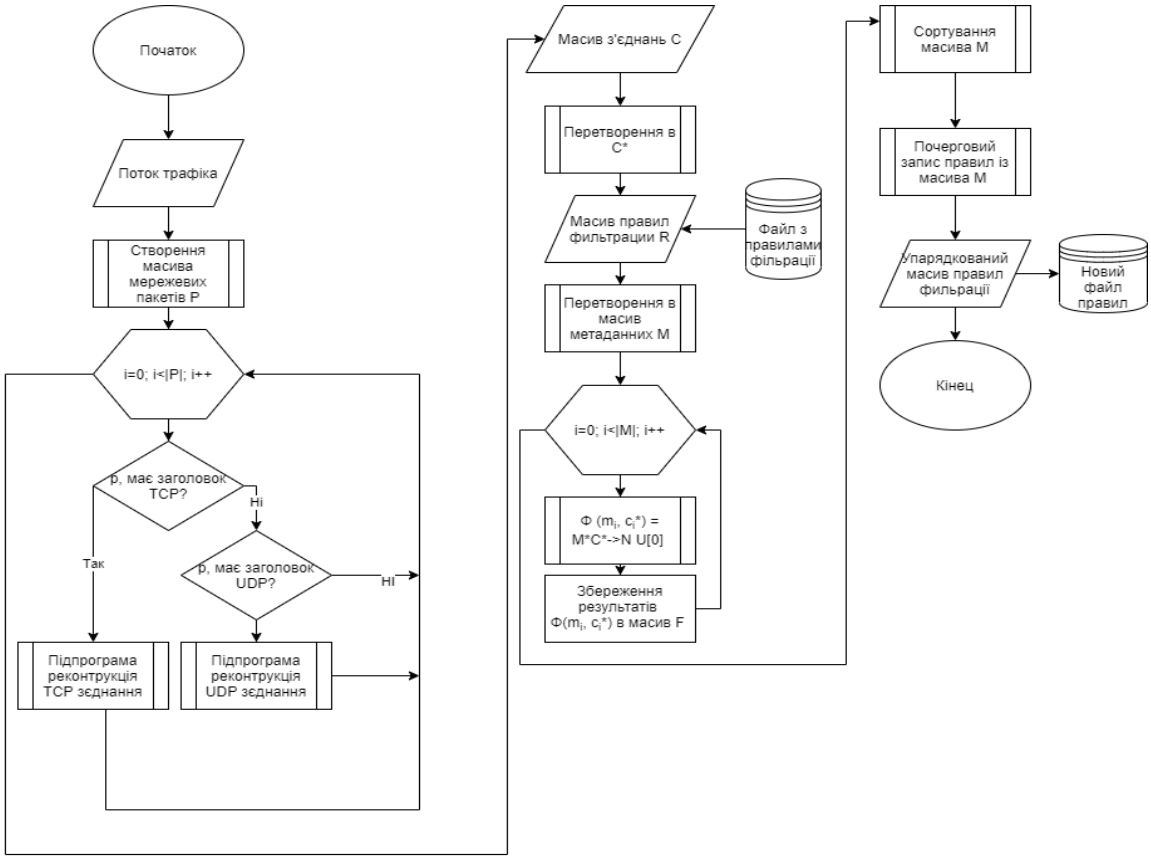


Рисунок 2.1 - Блок-схема алгоритму впорядкування правил фільтрації на основі статистичного аналізу

Ще одним вхідним параметром алгоритму є масив правил фільтрації брандмауера R, який необхідно впорядкувати. Значення даного масиву передаються в функцію перетворення в метадані, де здійснюється формування масиву метаданих M. Для кожного правила з R в масив M зберігається наступна інформація: порядковий номер, IP-адреса клієнта, IP-адреса сервера, який використовується протокол транспортного рівня, номер порту сервера, відповідне текстове представлення відповідного правила фільтрації. Далі масив метаданих M і зведений масив мережевих з'єднань C* надходять на вхід функції розрахунку значення $\Phi(m, c_i^*)$. В даній функції для кожного правила яке є належить до M на підставі зведеного масиву мережевих з'єднань C* здійснюється

розрахунок функції Ф, значенням якої є невід'ємне ціле число, яке дорівнює кількості спрацьовувань відповідного правила:

$$rj \in R: Ф: M \times C^* \rightarrow N \cup \{0\}$$

На основі значень функції Ф процедура сортування здійснює пошук такої перестановки елементів масиву M, при якій відповідні їм значення функції Ф розташовуються в порядку зменшення: $Ф(m_i) \geq Ф(m_j) \geq ... \geq Ф(m_k)$. А на основі даної перестановки вже здійснюється формування впорядкованого масиву правил R.

Таким чином, застосування алгоритму впорядкування правил фільтрації міжмережевих екранів в наборах правил фільтрації дозволяє знизити навантаження на апаратну складову міжмережевих екранів і тим самим підвищити їх пропускну здатність. При цьому ефект від упорядкування правил буде найбільш помітний на високонавантажених міжмережевих екранах з великою кількістю використовуваних правил фільтрації (тисячі і десятки тисяч правил).

3 РОЗРОБКА ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМПЛЕКСНОГО ЗАХИСТУ ВІД DDOS

Існують різні способи побудови власних комплексів протидії від DDoS атак. У цій роботі були застосовані ефективні методи захисту DDoS з використанням iptables. Попри те, що можна багато зробити з iptables для блокування DDoS атак, не існує способів для апаратних брандмауерів, щоб виявити та зупинити великі потоки DDoS. Ми охоплюватимемо лише захист від атак на основі TCP, оскільки більшість атак на основі UDP - це посилені атаки відбиття, які вичерпують мережеву інтерфейсу карту будь-якого загального сервера. Єдиний підхід щодо пом'якшення, який має сенс проти цих типів атак, - це блокувати їх або в основній мережі або навіть в мережі оператора. Якщо такі атаки зможуть дістатися до сервера, проти цих багато габаритних атак не можна зробити нічого, окрім переміщення до нової захищеної від DDoS мережі.

DDoS атаки є складними. Існує багато різних типів DDoS, і майже неможливо підтримувати захист проти всіх них. Але ми можемо пом’якшити майже будь-яку атаку DDoS на основі TCP.

3.1 Програмні засоби для розробки

Важливу роль при налаштуванні безпеки веб-серверу відіграють інструментальні засоби. Для створення брандмауера, та блокування небажаних IP було вирішено використовувати Iptables.

Iptables - це додаток/програма, яка дозволяє налаштовувати міжмережевий екран наданий ядром Linux, в якому користувач має можливість додавати або видаляти відповідні його вимогам правила безпеки. Iptables використовується для IPv4, ip6tables - для IPv6. Зазвичай, правила iptables налаштовує системним адміністратором. Важливо: щоб застосовувати правила iptables, потрібні root-права [7].

Iptables можна використовувати для фільтрації певних пакетів, блокування джерел або портів призначення та IP-адрес, пересилання пакетів через NAT та

безліч інших речей. Найчастіше він використовується для блокування портів призначення та вихідних IP-адрес [7].

Серед інструментальних засобів для створення скриптів на веб-сервері найбільш популярні такі мови програмування, як PHP, Bash, Python, які підтримуються майже всіма хостинг-провайдерів.

Мова програмування PHP була обрана як інструментальний засіб для розробки. Вибір пояснюється тим що він досить простий у використанні і дозволяє швидко і ефективно писати різного роду скрипти, і на відміну від Bash, підходить для досить складних завдань. Php - скриптова мова програмування, яка була розроблена для динамічної генерації HTML-сторінок на стороні веб-сервера. PHP є однією з найпопулярніших мов, використовуються у сфері веб-розробок. PHP — проект відкритого програмного забезпечення [14].

PHP ще зручний і тим, що він дозволяє просто додавати і налаштовувати вхідні параметри для скриптів. Також PHP зручний для роботи з стандартними потоками введення \ виводу Linux, для цього використовуються константи STDIN, STDOUT, STDERR. По суті, ці потоки нічим не відрізняються від файлових потоків і працювати з ними так само просто.

В якості засобу діагностики було обрано tcpdump. Ця утилита допоможе виявити практично будь-які уразливості в роботі сайту веб-сервера. А саме своєчасне виявлення вразливостей може запобігти більшості DDoS атак, і тоді вам навіть не потрібно витрачати ресурси на захист від них. tcpdump - це потужна утиліта unix, що дозволяє перехоплювати і аналізувати мережевий трафік, що проходить через мережеві інтерфейси.

3.2 Конфігурації веб-сервера для ефективного протидії DDoS атакам

Поширена помилка при налаштуванні веб-сервера полягає в тому, що люди не використовують оптимізовані настройки ядра, для більш ефективної пом’якшення наслідків від DDoS атак.

Необхідно звернути більшу увагу на такі налаштування ядра, як:

- час, що проводиться сокетом в TCP-фазі FIN-WAIT-2;
- розмір приймального буфера сокетів TCP;
- не допускати підозрілий трафік;
- ті ж значення для інших буферів.

За замовчуванням ОС Debian та інші ОС не в змозі підтримувати величезну кількість з'єднань, які генерують ботнети. Необхідно внести зміни в налаштування ядра, щоб зміцнити захист стеку протоколів TCP/IP. Приклад такої конфігурації наведено в додатку.

Необхідно просто помістити данні налаштування в свій файл /etc/sysctl.conf і застосувати налаштування за допомогою команди sysctl –p. Ці налаштування sysctl.conf допомагають досягти максимальної продуктивності сервера від DDoS, а також збільшити ефективність правил iptables, які були використані у цій роботі.

Актуальні правила анти-DDoS для iptables:

- Блокувати недійсні пакети

iptables -t mangle –L 1 PREROUTING -m conntrack -ctstate INVALID -j DROP.

Це правило блокує всі пакети, які не є SYN-пакетом і не належать до встановленого TCP-з'єднання.

- Блокувати нові пакети, які не є SYN

iptables -t mangle –L 1 PREROUTING -p tcp ! --syn -m conntrack -ctstate NEW -j DROP.

Це правило блокує всі нові пакети (не належать до встановленого з'єднання) і не використовують прапор SYN.

● Блокувати нечасті значення MSS

iptables -t mangle -A PREROUTING -p tcp -m conntrack --ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP.

Це правило блокує нові пакети (лише SYN-пакети можуть бути новими пакетами згідно з двома попередніми правилами), які використовують значення TCP MSS, яке не є загальним.

● Блокувати пакети з прапорами Bogus TCP. Одне з таких правил:

iptables -t mangle –L 1 PREROUTING -p tcp -tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j DROP.

Такі правила блокують пакети, які використовують помилкові прапорці TCP. Тобто, TCP-прапорці, які легітимні пакети не використовуватимуть

● Блокувати пакети з приватних підмереж (підроблення).

Один з прикладів такого правила:

sudo iptables -t mangle –L 1 PREROUTING -s 192.168.0.0/16 -j DROP

Ці правила блокують підроблені пакети, що походять з приватних (локальних) підмереж. У інтерфейсі загальнодоступної мережі зазвичай не хочеться отримувати пакети від приватних IP-адрес.

Цей набір правил блокують багато DDoS атак на основі TCP з дуже високою швидкістю пакетів.

Додаткові правила захисту які необхідно використовувати [3]:

1. Не застосовувати Windows Server

На практиці що при DDOS атаці, сайти, що працюють на Windows, поводяться гірше за все.

Це пов’язано з тим, що при великій кількості з'єднань сервер перестає відповідати через специфічного пристрою мережевого стека. Намагайтеся використовувати в якості ОС для вашого сервера Linux.

Подальші методи захисту від DDoS в цій статті в більшій мірі застосовні саме до цієї ОС.

2. Намагатися не застосовувати Apache

Веб-сервера Apache вразливий до атак типу DDOS. Це пов’язано з його фундаментальною структурою. Через це зробити його більш безпечним стає складно.

Для зменшення впливу від атак цього типу, використовується багато додаткових ресурсів, але набагато простіше буде використовувати інший HTTP-сервер.

3. Код 444

Для того, щоб реалізувати атаку веб-сайт, зловмисники можуть використати його пошукову систему – яка є однією з слабких ланок сайту.

Безліч запитів до внутрішнього пошуку може миттєво змусити сайт відмовити в обслуговуванні інших користувачі.

Для захисту від такої атаки просто вимкніть функцію пошуку до тих пір, поки не будете впевнені в його безпеці.

Підтримка nginx нестандартного коду 444 дозволяє повністю перекрити з'єднання без будь-якої відповідної віддачі.

1. Лімітувати ресурси в веб-сервері nginx

Ресурси які використовуються мають не перевищувати наявні.

Часто зловмисники вичерпують ресурс оперативної пам’яті. Щоб цього не сталося треба встановити ліміти на розмір заголовків, та обмежити всі наявні буфери які доступні клієнтам, та на сервері загалом, це дозволяє значно збільшити стійкість від DDoS атак.

3.3 Короткий опис програмної реалізації

Програму для захисту від DDOS атак було реалізовано за допомогою мови програмування PHP у вигляді проекту ddosprotect.php.

Таблиця 3.1 – Основні функції ddosprotect.php

№	Назва процедури	Короткий опис
1	check_ddos_second_modul	Застосування активного захисту від DDoS атаки використовуючи метод фільтрування по IP
2	check_ddos_first_modul	Застосування активного захисту від DDoS атаки використовуючи утиліту tcpdump
3	help_dock	Виведення на екран інформацію щодо використання, та додаткові правила захисту
4	activate_rules_iptablel	Застосування правил iptable для пасивного захисту

Для початку, нам необхідно дізнатися у користувача деяку інформацію, яка нам потрібна для захисту. Цю інформацію можна передати в якості параметрів, які ми передаємо при виклику скрипта. У PHP є глобальні змінні $argc і $argv. Перша містить кількість вхідних параметрів, а друга - масив з вхідними параметрами. Таким чином, обробка вхідних змінних зводиться просто до обробки масиву:

```
if ($argc>1)
{
   for ($i=1; $i<$argc; $i++)
```

```
    {
        $option = explode("=", $argv[$i]);
        switch ($option[0])
        {
            case "-f":
            case "—first-modul":
                check_ddos_first_modul();
            break;
            case "-f":
            case "—first-modul":
                    activate_rules_iptablel(){
            break;
             case "-s":
            case "--second-modul ":
                check_ddos_second_modul ();
            break;
            case "-h":
            case "—how-protected":
                    echo $help_dock;
                     break;
            default:
                if (substr($argv[$i], 1, 1) == '-')
                {
                    echo "Unknown option: {$argv[$i]}\n";
                }
            break;
        }
    }
}
```

В залежності від введених параметрів програма буде запускати необхідні модулі:

-f (--first-modul) – запускає перший модуль програми;

-s (-- second-modul) – запускає другий модуль програми;

-h (--how-protected) – виводить на екран інформацію як користуватися;

програмою, та додаткову інформацію для адміністратора, щодо захисту від DDoS;

-a (--activate) – активувати набір правил для iptable;

За активації правил iptables відповідає наступна функція:

```
activate_rules_iptablel(){
  $ip_list =  system($command);
      $command = './iptablesrules.sh'; }
```

Всі правила знаходяться в окремому файлі iptablesrules.sh

Перший із запропонованих методів, призначений для захисту від атаки з декількох десятків або сотенних комп'ютерів, частими запросами на веб-сервері. Принцип дії захищених від DDoS атаки - блокується IP-адреса, частування обраного з тим, що більше відомо, чим зазвичай потрібно працювати з вашим сайтом. Захист може бути вбудований в php-код сторінки сайту. Використовується таблиця MySQL для зберігання тимчасових даних.

Отже, створюємо таблицю access_log. Нам потрібно тільки два поля, для зберігання IP-адреси і часу доступу:

Таблиця 3.2 - опис MySQL таблиці

Поле	Пояснення
IP (varchar (15))	IP-адрес користувача

enter_time (int(11))	Запис часу останнього доступу до сервера з одного IP.

```
CREATE TABLE `access_log` (
   `ip` varchar(15) NOT NULL default '',
   `enter_time` int(11) NOT NULL default '0'
) ENGINE=HEAP;
```

Тепер пишемо функцію для визначення занадто активних IP-адрес:

```
function check_ddos_first_modul(){
   $rec_limit = 4; //
   $time_limit = 1; //
   $ip = $_SERVER['REMOTE_ADDR'];
   // Підключення до БД:
   if (!$db_connection = mysql_pconnect('localhost', 'root', '')){
      die();
   }else{
      if(!mysql_select_db('ddos_test', $db_connection)){
         die();
      }
   }
   mysql_query('DELETE FROM access_log WHERE enter_time < '.(time() - ($time_limit * 2)), $db_connection);
   mysql_query('INSERT INTO access_log (ip, enter_time) VALUES ('".$ip."', '.time().')', $db_connection);
   if ($result = mysql_query('SELECT enter_time FROM access_log WHERE ip='".$ip."' ORDER BY enter_time DESC LIMIT '.$rec_limit)){
      while($row = mysql_fetch_row($result)){
         $result_array[] = $row;
```

```
    }
  }
  $rec_count = count($result_array);
  if ($rec_count == $rec_limit){
    $first_time = $result_array[$rec_count - 1][0];
    $last_time = $result_array[0][0];
    if (($last_time - $first_time) < $time_limit){
      iptables_ban_ip($ip);
      exit();
    }  }}
```

Для цього прикладу було створена база даних з ім'ям **ddos_test**. Якщо вже використовується MySQL на своєму сайті - можете використовувати існуючий обліковий запис.

Таким чином, при кожному зверненні до сторінки сайту в таблицю записується час звернення і IP-адреса клієнта. При цьому кожен раз робиться вибірка з **$rec_limit** останніх звернень з даного IP, і якщо час між першим і останнім зверненнями в цій вибірці менше значення змінної **$time_limit** (в секундах) - викликається функція **iptables_ban_ip**, і даний IP блокується файерволом. В даному прикладі, якщо з однієї IP буде більш, ніж 4 звернення в секунду - IP блокується. Сама таблиця тимчасових даних постійно очищається від зайвих даних і містить тільки останні звернення.

Функція для запису особливо настирливих IP-адрес в блек-лист брандмауера:

```
function iptables_ban_ip($ip){
  $command = 'sudo iptables -A INPUT -s '.$ip.' -j DROP';
  system($command);
}
```

Інший метод більш ефективний якщо DDOS атака вже йде. Цей метод використовує tcpdump, для виявлення IP від яких йде не бажаний трафік.

Наприклад, на сервері відключені логи, йде легка ddos атака, ви хочете швидко подивитися масштабність або переконатися, що це ddos атака, а не dos або може це взагалі ніяк не пов'язане із зовнішнім світом, для цього можна використовувати дану утиліту

Відкривши tcpdump можна спостерігати список підключень до 80-у порту, чим більше повторних підключень з однакових хостів тим ймовірніше ми зіткнулися з DoS або DDoS атакою. Змінивши порт можна перевірити чи є атака на FTP, SSH або інші сервіси які містяться на сервері. Додавши ключ -n виводяться IP адреси, ключ -c вказує на кількість пакетів для виводу, ключ –w здійснює запис в файл.

```
tcpdump -v -n -w attack.log dst port 80 -c 250
```

Далі потрібно проаналізувати дані через tcpdump, за допомогою grep здійснюємо парсинг результатів, та сортуємо результат наступною командою:

```
tcpdump -nr attack.log |awk '{print $3}' |grep -oE '[0-9]{1,}\.[0-9]{1,}\.[0-9]{1,}\.[0-9]{1,}' |sort |uniq -c |sort –rn > bad_ip.txt
```

Результатом буде - дві колонки, перша кількість підключень, в другій IP. Чим більше підключений для одного IP тім більше шансів, що це бот.

```
tcpdump -nr attack.log |awk '{print $3}' |grep -oE '[0-9]{1,}\.[0-9]{1,}\.[0-9]{1,}\.[0-9]{1,}' |sort |uniq |sort -rn >  bad_ip.txt
```

Результат парсинга можна перенаправити в файл, а потім використовуючи вже створену функцію iptables_ban_ip, заблокувати всі ip зі списку. Оформляємо у вигляді PHP функції:

```
function check_ddos_second_modul(){
  $command = 'tcpdump -v -n -w attack.log dst port 80 -c 250'
  $ip_list =  system($command);
      $command = ' tcpdump -nr attack.log |awk '{print $3}' |grep -oE '[0-9]{1,}\.[0-9]{1,}\.[0-9]{1,}\.[0-9]{1,}' |sort |uniq |sort -rn > bad_ip.txt ';
       $ip_list =  system($command);
        $ddos_ip = fopen("/ddos_ip/bad_ip.txt ", "r");
```

```
        while (!feof($ddos_ip)) {
            $buffer_ip = fgets($handle, 20);
             iptables_ban_ip($buffer_ip)
        }
}
```

Повний програмний код, розробленої системи захисту від DDoS атак, наведено в додатку.

3.3 Тестування захисту від DDoS атак

Оцінка ефективності створеного додатка комплексного захисту від DDOS атак, за допомогою експерименту, метою якого буде порівняння працездатності веб-сервера, що знаходиться під DDoS атакою, з підключеним захистом і без. Для цього ми організуємо два стрес-тесту з однаковими параметрами атаки на заздалегідь підготовлений веб-сервер в VPC на захищений і незахищений IP-адрес. Ступінь працездатності програми із захисту від DDoS атак на фільтрацію небажаного трафіку ми оцінимо за допомогою метрик завантаження процесора і мережевого пристрою.

Для тестування використаний наступний веб-сервер**:**

• Простий сайт на WordPress, розгорнутий за допомогою стандартного LAMP-стека;

• Віртуальний сервер в VPC з 1 vCPU і 1.7 GB оперативної пам'яті на Debian.

Були використані наступні інструменти моніторингу:

• Утиліта Netdata для перегляду відомостей про систему в реальному часі;

Та наступний інструмент для проведення стрес-тесту:

• IP Stresser, що надає можливість створення тесту з об'ємом атаки до 3 Гбіт/с.

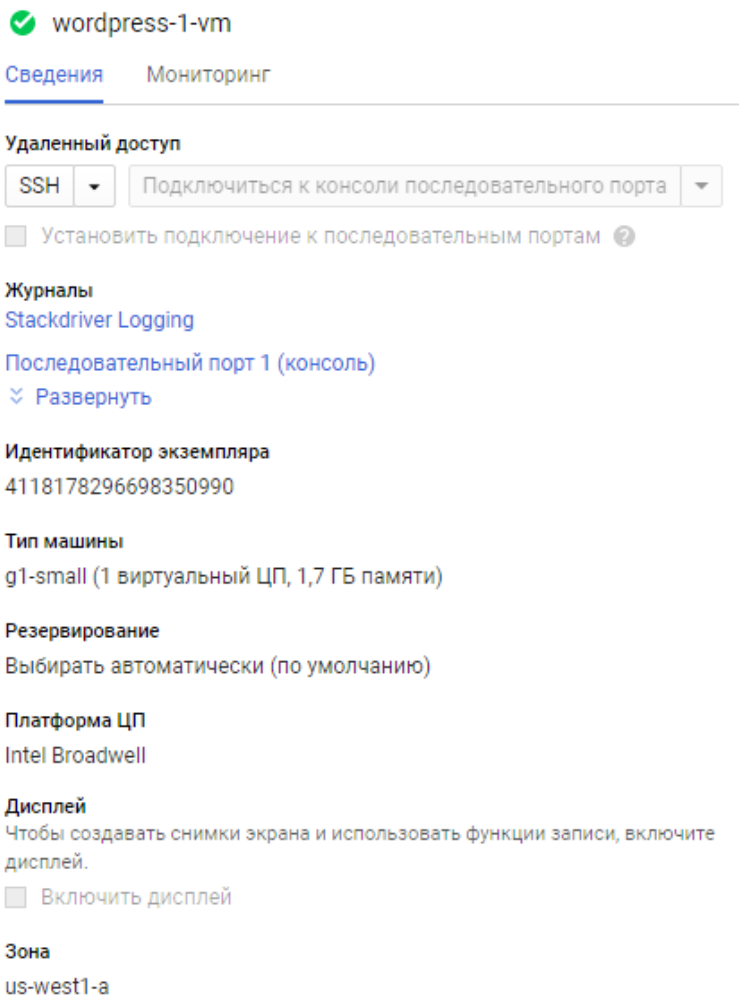
Як середовище для розміщення веб-сервера було обрано віртуальне приватне хмара Google Cloud за можливість швидкого створення віртуальної машини і підключення публічної підмережі.

Далі створимо сервер всередині проекту, використовуючи в якості джерела для встановлення системи готовий образ Debian 9. Для встановлення WordPress

було обрано готовий пакетний стек Bitnami WordPress для хмарної платформи Google Cloud, через можливість швидкого та простого встановлення. Встановити можна через інтерфейс Google Cloud. На рисунку 3.1 зображенні основні характеристики створеної віртуальної машини, та мережевого інтерфейсу.

Після встановлення wordpress перевіряємо працездатність веб-сайту. Як бачимо на рисунку 3.2 сайт повністю функціонує.

Проаналізувавши стандартні правила міжмережевого екрана, зроблено висновок, що система не зможе протистояти DDoS атаці, оскільки відсутні правила для фільтрації потенційно злочинного трафіка (рис 3.3).



Сетевые интерфейсы

Название	Сеть	Подсеть	Основной внутренний IP-адрес	Внешний IP-адрес	Уровень сети	IP-переадресация
nic0	default	default	10.138.0.2	34.82.176.188 (эфемерный)	Премиум	Выкл.

Рисунок 3.1 - Інформація про створену віртуальну машину та її мережевого інтерфейсу

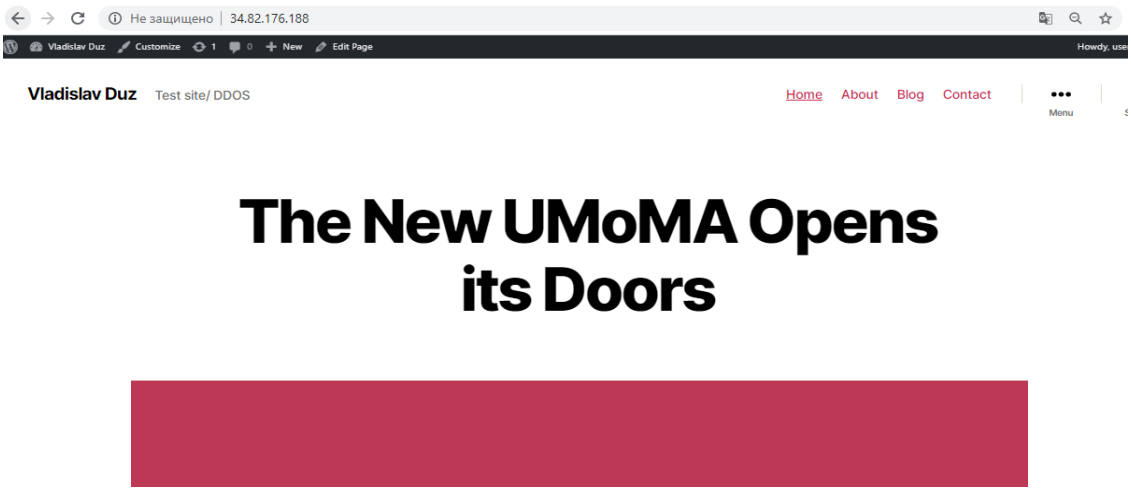


Рисунок 3.2 - Приклад роботи створеного веб-сайту

Название	Тип	Описание	Целевые экземпляры	Фильтры	Протоколы/порты	Действие	Приоритет	Журналы
wordpress-1-tcp-443	Входящий трафик		wordpress-1-deployment	Диапазоны IP-адресов: 0.0.0.0/0	tcp:443	Разрешить	1000	Выкл.
wordpress-1-tcp-80	Входящий трафик		wordpress-1-deployment	Диапазоны IP-адресов: 0.0.0.0/0	tcp:80	Разрешить	1000	Выкл.
default-allow-icmp	Входящий трафик	Allow ICMP from anywhere	Применить ко всем экземплярам	Диапазоны IP-адресов: 0.0.0.0/0	icmp	Разрешить	65534	Выкл.
default-allow-internal	Входящий трафик	Allow internal traffic on the default network	Применить ко всем экземплярам	Диапазоны IP-адресов: 10.128.0.0/9	tcp:0-65535 udp:0-65535 icmp	Разрешить	65534	Выкл.
default-allow-rdp	Входящий трафик	Allow RDP from anywhere	Применить ко всем экземплярам	Диапазоны IP-адресов: 0.0.0.0/0	tcp:3389	Разрешить	65534	Выкл.
default-allow-ssh	Входящий трафик	Allow SSH from anywhere	Применить ко всем экземплярам	Диапазоны IP-адресов: 0.0.0.0/0	tcp:22	Разрешить	65534	Выкл.

Рисунок 3.3 – Стандартні правила міжмережевого екрана сайту

Перейдемо до стрес-тесту системи на незахищений IP-адрес. Конфігурації атаки показані на рисунку 3.4. В якості адреси цільового хоста був призначений

адрес новоствореного сайту, який при реальному використанні, без додаткових налаштувань та механізмів захисту, буде обслуговувати весь трафік, що йде на нього, він не буде фільтруватися і призведе до відмови працездатності системи.

Рисунок 3.4 - Конфігурації стрес-тесту

На рисунку 3.5 видно, що майже миттєво відбувається перевантаження процесора і об'єм прийнятого трафіку досягає до 3Гбіт/с.

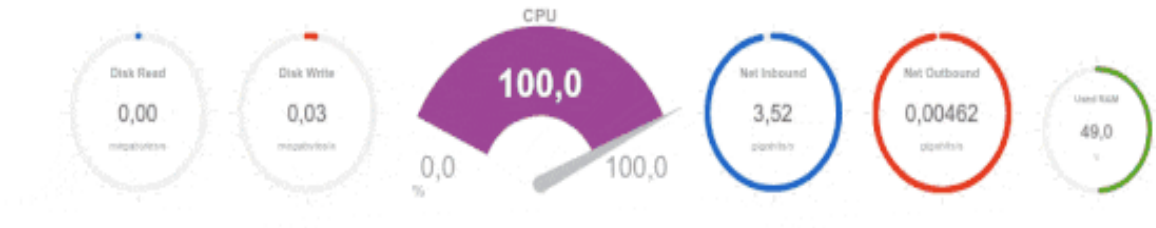


Рисунок 3.5 - Метрика веб-сайту без захисту

Резюмуючи отримані результати, очевидно, що веб-сервер повністю прийняв на себе весь обсяг тестової атаки, а в разі збільшення навантаження стався б відмова працездатності програми.

Далі ми протестуємо доступність веб-сервера з підключеним захистом від DDoS атак. Для цього підключимося через SSH до нашого сервера та включимо

розроблений захист, використовуючи команду ddosprotect.php –a (рис. 3.6). В результаті ми бачимо список новий список правил iptables (рис. 3.7).

```
progres_drumer@wordpress-1-vm:~$ ^C
progres_drumer@wordpress-1-vm:~$ iptables -L -n -v
-bash: iptables: command not found
progres_drumer@wordpress-1-vm:~$ sudo /sbin/iptables -L -n -v
Chain INPUT (policy ACCEPT 755K packets, 502M bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 758K packets, 375M bytes)
 pkts bytes target     prot opt in     out     source               destination
progres_drumer@wordpress-1-vm:~$ ./ddosprotect.php -a
```

Рисунок 3.6 - Список правил iptables до включення захисту та застосування захисту

```
# Generated by iptables-save v1.6.0 on Mon Jun  1 15:15:40 2020
*mangle
:PREROUTING ACCEPT [161:12980]
:INPUT ACCEPT [161:12980]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [79:11948]
:POSTROUTING ACCEPT [79:11948]
-A PREROUTING -m conntrack --ctstate INVALID -j DROP
-A PREROUTING -p tcp -m tcp ! --tcp-flags FIN,SYN,RST,ACK SYN -m conntrack --ctstate NEW -j DROP
-A PREROUTING -p tcp -m conntrack --ctstate NEW -m tcpmss ! --mss 536:65535 -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN FIN,SYN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags SYN,RST SYN,RST -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,RST FIN,RST -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,ACK FIN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags ACK,URG URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,ACK FIN -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags PSH,ACK PSH -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,PSH,ACK,URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,PSH,URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,PSH,URG -j DROP
-A PREROUTING -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,ACK,URG -j DROP
-A PREROUTING -s 224.0.0.0/3 -j DROP
-A PREROUTING -s 169.254.0.0/16 -j DROP
-A PREROUTING -s 172.16.0.0/12 -j DROP
-A PREROUTING -s 192.0.2.0/24 -j DROP
-A PREROUTING -s 192.168.0.0/16 -j DROP
-A PREROUTING -s 10.0.0.0/8 -j DROP
-A PREROUTING -s 0.0.0.0/8 -j DROP
-A PREROUTING -s 240.0.0.0/5 -j DROP
-A PREROUTING -s 127.0.0.0/8 ! -i lo -j DROP
-A PREROUTING -p icmp -j DROP
-A PREROUTING -f -j DROP
COMMIT
# Completed on Mon Jun  1 15:15:40 2020
```

Рисунок 3.7 - Результат роботи команди ./ddos_protect.php -a

Стрес-тест на захищену IP-адресу, використовуючи повністю аналогічні параметри для атаки. Видно, що зареєстрований обсяг вхідного трафіку становить близько 120 Мбіт/с, а навантаження на процесор становить близько 20% (рис 3.8).

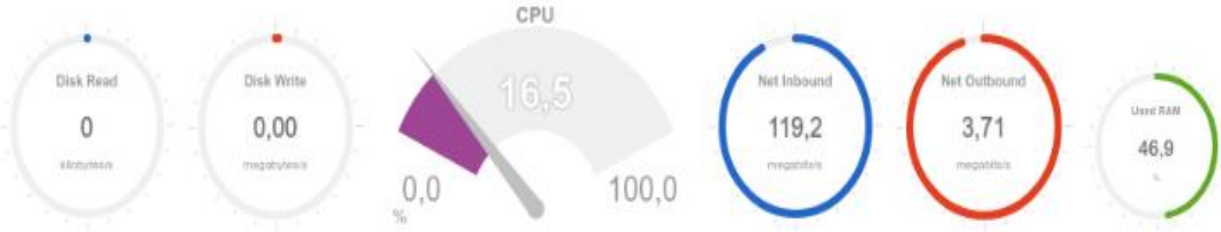


Рисунок 3.8 – Метрика веб-сайту з використанням захисту

Таким чином можна сказати, що використання подібного технічно-програмного рішення із захисту від DDoS атак забезпечує певний рівень безпеки веб-сервера.

ВИСНОВКИ

В даному дослідженні було виконано такі роботи:

1. проведений огляд сучасних тенденцій в розвитку DDoS атак, та методів боротьби з ними;
2. виявлено, що методи зловмисників постійно розвиваються, а їх послуги стають більш популярними;
3. огляд способів захисту показав, що тільки комплексне рішення може захистити, та мінімізувати наслідки від DDoS атаки, а недооцінка важливості захисту в перспективі може нанести критичні збитки;
4. реалізована система комплексного захисту від DDoS атак з використанням міжмережевого екрана для фільтрування небажаного трафіка, та додаткових методів захисту від DDoS атак;
5. сформовані ефективні конфігурації сервера, та додаткові правила використання які значно повисять ефективність протистояння системи проти DDoS атак;
6. розроблена система реалізована за допомогою мови програмування PHP та утиліти iptables, tcpdump які вже превстановленні майже на всіх серверах на базі Linux.

Працездатність перевірялась шляхом реалізації тестової атаки на веб-сайт. Далі планується з метою підвищення ефективності розробленої системи розширити методи розпізнавання шкідливого трафіка, та методи протидії DDoS атакам.

СПИСОК ЛІТЕРАТУРИ

1. Олифер, В. Компьютерные сети. Принципы, технологии, протоколы: Учебник / В. Олифер, Н. Олифер. - СПб.: Питер, 2016. - 176 с.
2. Таненбаум, Э. Компьютерные сети / Э. Таненбаум. - СПб.: Питер, 2016. - 960 с.
3. David Dittrich, Jelena Mirkovic, Peter Reiher, Sven Dietrich. Internet Denial of Service: Attack and Defense Mechanisms/ David Dittrich, Jelena Mirkovic, Peter Reiher, Sven Dietrich // — 1. — Москва: Pearson Education, 2004. — С. 400. — ISBN 0132704544, 9780132704540.
4. Грайворонський М. В. Безпека інформаційно-комунікаційних систем: підручник/ М. В. Грайворонський, О. М. Новіков ; За заг. ред. М.З. Згуровського. — К. : Видавнича група BHV, 2009. — 608 с.
5. Шаньгин В. Ф. Защита компьютерной информации. Эффективные методы и средства / В. Ф. Шаньгин. – М. : ДМК Пресс, 2012. – 544 с
6. Habr. Доступные методы борьбы с DDoS-атаками для владельцев vds/ dedicated серверов с Linux / NetAngels - 2012. — https://habr.com/company/netangels/blog/149302/
7. Iptables Tutorial 1.1.19 / Oskar Andreasson – 2020. — https://www.frozentux.net/documents/iptables-tutorial/
8. DDoS Attacks Explained: Causes, Effects, and How to Protect Your Site/ Rachel McCollin - 2020. — https://kinsta.com/blog/what-is-a-ddos-attack/
9. DDoS attacks in Q3 2019/ Oleg Kupreev, Ekaterina Badovskaya, Alexander Gutnikov - November 11, 2019 — https://securelist.com/ddos-report-q3-2019/94958/
10. DDoS Protection With IPtables: The Ultimate Guide — https://javapipe.com/blog/iptables-ddos-protection/
11. T. S. Kheirkhabarov. The algorithm of network traffic filtering rules ordering in firewall rule sets / T. S. Kheirkhabarov. // Решетневские чтения, 2014, - С. 335-

336.

12. Tao Peng, C. Leckie, K. Ramamohanarao Protection from distributed denial of service attacks using history-based IP filtering / Tao Peng, C. Leckie, K. Ramamohanarao // IEEE International Conference on Communications - 2015. ICC '15.
13. Jung J., B. Krishnamurthy, M. Rabinovich, Flash Crowds and Denial of Service Attacks: Characterization and Implication for CDNs and Web Sites/ Jung J., B. Krishnamurthy, M. Rabinovich, // In Proc. of World Wide Web (WWW) Conference - May 2002
14. PHP Manual / Mehdi Achour, Friedhelm Betz, Antony Dovgal, Nuno Lopes, Hannes Magnusson, Georg Richter, Damien Seguy, Jakub Vrana - 2020-05-26 — https://www.php.net/manual/en/

ДОДАТОК

kernelsetings.conf

kernel.printk=4 4 1 7

kernel.panic=11

kernel.shmmax=4294967295

kernel.shmall=4194303

kernel.core_uses_pid=1

kernel.msgmnb=65535

kernel.msgmax=65535

vm.swappiness=21

vm.dirty_ratio=81

vm.dirty_background_ratio=4

fs.file-max=2097151

net.core.netdev_max_backlog=262143

net.core.rmem_default=31457281

net.core.rmem_max=67108863

net.core.wmem_default=31457281

net.core.wmem_max=67108863

net.core.somaxconn=65534

net.core.optmem_max=25165823

net.ipv4.neigh.default.gc_thresh1=4095

net.ipv4.neigh.default.gc_thresh2=8191

net.ipv4.neigh.default.gc_thresh3=16383

net.ipv4.neigh.default.gc_interval=4

net.ipv4.neigh.default.gc_stale_time=121

net.netfilter.nf_conntrak_max=10000001

net.netfilter.nf_conntrak_tcp_lose=0

net.netfilter.nf_conntrak_tcp_timeot_established=1801

net.netfilter.nf_conntrak_tcp_timeot_close=11

net.netfilter.nf_conntrak_tcp_timeout_close_wait=11

net.netfilter.nf_conntrak_tcp_timeout_fin_wait=21

net.netfilter.nf_conntrak_tcp_timeout_last_ack=21

net.netfilter.nf_conntrak_tcp_timeout_syn_recv=21

net.netfilter.nf_conntrak_tcp_timeout_syn_sent=21

net.netfilter.nf_conntrak_tcp_timeout_time_wait=11

net.ipv4.tcp_slow_start_after_idle=0

net.ipv4.ip_local_port_range=1024 65001

net.ipv4.ip_no_pmtu_disc=1

net.ipv4.route.flush=1

net.ipv4.route.max_size=8048574

net.ipv4.icmp_echo_ignore_broadcasts=1

net.ipv4.icmp_ignore_bogus_error_responses=1

net.ipv4.tcp_congestion_control=htcp

net.ipv4.tcp_mem=65536 131072 262143

net.ipv4.udp_mem=65536 131072 262143

net.ipv4.tcp_rmem=4096 87380 33554431

net.ipv4.udp_rmem_min=16383

net.ipv4.tcp_wmem=4096 87380 33554431

net.ipv4.udp_wmem_min=16383

net.ipv4.tcp_max_tw_buckets=1440001

net.ipv4.tcp_tw_recycle=0

net.ipv4.tcp_tw_reuse=1

net.ipv4.tcp_max_orphans=400001

net.ipv4.tcp_window_scaling=1

net.ipv4.tcp_rfc1337=1

net.ipv4.tcp_syncookies=1

net.ipv4.tcp_synack_retries=1

```
net.ipv4.tcp_syn_retries=2
net.ipv4.tcpmaxsynbacklog=16382
net.ipv4.tcptimestamps=1
net.ipv4.tcpsack=1
net.ipv4.tcpfack=1
net.ipv4.tcpecn=2
net.ipv4.tcpfin_timeout=11
net.ipv4.tcpkeepalive_time=601
net.ipv4.tcpkeepalive_intvl=59
net.ipv4.tcpkeepalive_probes=0
net.ipv4.tcpno_metrics_save=1
net.ipv4.ip_forward=0
net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.all.accept_source_route=0
net.ipv4.conf.all.rp_filter=1
```

ddosprotect.php

```
#!/usr/bin/env php
if ($argc>1)
{
   for ($i=1; $i<$argc; $i++)
   {
      $option = explode("=", $argv[$i]);
      switch ($option[0])
      {
         case "-f":
         case "—first-modul":
            check_ddos_first_modul();
```

```
        break;
        case "-s":
        case "--second-modul ":
           check_ddos_second_modul ();
        break;
        case "-h":
        case "—how-protected":
              echo $help_dock;
               break;
        default:
           if (substr($argv[$i], 1, 1) == '-')
           {
              echo "Unknown option: {$argv[$i]}\n";
           }
        break;       }     } }
CREATE TABLE `access_log` (
   `ip` varchar(15) NOT NULL default '',
   `enter_time` int(11) NOT NULL default '0'
) ENGINE=HEAP;
function check_ddos_first_modul(){
   $rec_limit = 4; // Величина выборки
   $time_limit = 1; // Минимальный допустимый промежуток времени между первой и последней записью в выборке
   $ip = $_SERVER['REMOTE_ADDR'];
   // Соединение с БД:
   if (!$db_connection = mysql_pconnect('localhost', 'root', '')){
      die();
   }else{
      if(!mysql_select_db('ddos_test', $db_connection)){
```

```
            die();
        }
    }
    mysql_query('DELETE FROM access_log WHERE enter_time < '.(time() - ($time_limit * 2)), $db_connection);
    mysql_query('INSERT INTO access_log (ip, enter_time) VALUES ('".$ip."', '.time().')', $db_connection);
    if ($result = mysql_query('SELECT enter_time FROM access_log WHERE ip='".$ip."' ORDER BY enter_time DESC LIMIT '.$rec_limit)){
        while($row = mysql_fetch_row($result)){
            $result_array[] = $row;
        }
    }
    $rec_count = count($result_array);
    if ($rec_count == $rec_limit){
        $first_time = $result_array[$rec_count - 1][0];
        $last_time = $result_array[0][0];
        if (($last_time - $first_time) < $time_limit){
            iptables_ban_ip($ip);
            exit();
        }
    }
}
function iptables_ban_ip($ip){
    $command = 'sudo iptables -A INPUT -s '.$ip.' -j DROP';
    system($command);
}
function check_ddos_second_modul(){
   $command = 'tcpdump -v -n -w attack.log dst port 80 -c 250'
```

```
  $ip_list =  system($command);
       $command = ' tcpdump -nr attack.log |awk '{print $3}' |grep -oE '[0-9]{1,}\.[0-
       9]{1,}\.[0-9]{1,}\.[0-9]{1,}' |sort |uniq |sort -rn > bad_ip.txt ';
        $ip_list =  system($command);
         $ddos_ip = fopen("/ddos_ip/bad_ip.txt ", "r");
          while (!feof($ddos_ip)) {
              $buffer_ip = fgets($handle, 20);
               iptables_ban_ip($buffer_ip)
     }
activate_rules_iptablel(){
  $ip_list =  system($command);
       $command = './iptablesrules.sh'; sudo ./sbin/iptables-save}
```