**Захист від DDoS-атак**

**Вступ**

DDoS-атаки (Distributed Denial of Service) — це один із найпоширеніших видів кібератак, спрямованих на перевантаження сервера або мережі, щоб зробити їх недоступними для користувачів. У цьому матеріалі ми розглянемо, як можна реалізувати базовий захист від DDoS-атак за допомогою мови програмування Python. Хоча повноцінний захист потребує комплексного підходу, цей приклад допоможе зрозуміти основи виявлення та блокування підозрілої активності.

DDoS-атака полягає в тому, що велика кількість запитів надсилається до сервера з різних джерел одночасно. Це може призвести до перевантаження ресурсів сервера (процесора, пам’яті чи пропускної здатності мережі), через що легітимні користувачі не можуть отримати доступ до послуги. Наша мета — виявити аномальну поведінку та обмежити її вплив.

Ping Flood — це тип DDoS-атаки, коли зловмисник надсилає велику кількість ICMP-запитів (ping) до цільової мережі, перевантажуючи її ресурси. У цьому матеріалі ми створимо скрипт на Python для виявлення підозрілої активності в мережі та інтегруємо його з Fail2Ban для автоматичного блокування атакуючих IP-адрес.

* Використаємо Scapy для точного аналізу мережевих пакетів.
* Логуватимемо підозрілу активність у файл для Fail2Ban.
* Додамо гнучкість для моніторингу не лише ICMP, а й інших типів трафіку (наприклад, TCP).

import time

from scapy.all import sniff, IP, ICMP, TCP

from collections import defaultdict

import os

*# Налаштування параметрів*

REQUEST\_LIMIT = 10 *# Максимальна кількість запитів від одного IP за CHECK\_INTERVAL секунд*

CHECK\_INTERVAL = 10 *# Інтервал перевірки у секундах*

LOG\_FILE = "/var/log/fail2ban\_network\_flood.log" *# Шлях до лог-файлу для Fail2Ban*

*# Лічильники запитів від IP*

request\_counter = defaultdict(list)

def log\_suspicious\_ip(ip):

"""Логування підозрілої IP-адреси у файл для Fail2Ban."""

timestamp = time.strftime('%Y-%m-%d %H:%M:%S')

with open(LOG\_FILE, "a") as log:

log.write(f"{timestamp} - Suspicious activity from {ip}\n")

print(f"Підозріла активність від {ip} записана в лог.")

def packet\_handler(packet):

"""Обробка вхідних мережевих пакетів."""

current\_time = time.time()

*# Перевіряємо, чи є IP-шар у пакеті*

if IP in packet:

ip\_src = packet[IP].src

*# Фільтруємо ICMP або TCP-пакети (можна додати інші протоколи)*

if packet.haslayer(ICMP) or packet.haslayer(TCP):

*# Очищаємо старі записи (старші за CHECK\_INTERVAL секунд)*

request\_counter[ip\_src] = [t for t in request\_counter[ip\_src] if current\_time - t < CHECK\_INTERVAL]

*# Додаємо новий запит*

request\_counter[ip\_src].append(current\_time)

*# Перевіряємо ліміт запитів*

if len(request\_counter[ip\_src]) > REQUEST\_LIMIT:

log\_suspicious\_ip(ip\_src)

del request\_counter[ip\_src] *# Очищаємо записи після логування*

def monitor\_traffic():

"""Запуск моніторингу мережевого трафіку."""

print("Запуск моніторингу трафіку...")

*# Фільтруємо ICMP і TCP-пакети, store=0 для економії пам’яті*

sniff(filter="icmp or tcp", prn=packet\_handler, store=0)

if \_\_name\_\_ == "\_\_main\_\_":

*# Перевіряємо наявність лог-файлу та права доступу*

if not os.path.exists(LOG\_FILE):

print(f"Лог-файл {LOG\_FILE} не існує. Створіть його та перевірте права.")

elif not os.access(LOG\_FILE, os.W\_OK):

print(f"Немає прав на запис у {LOG\_FILE}. Надайте права (наприклад, chmod 666).")

else:

monitor\_traffic()

**Як працює скрипт?**

1. **Аналіз пакетів**: Scapy відстежує ICMP (ping) і TCP-пакети, отримуючи IP-адресу відправника.
2. **Підрахунок запитів**: Для кожної IP зберігається список часових міток запитів за останні CHECK\_INTERVAL секунд.
3. **Виявлення атаки**: Якщо кількість запитів перевищує REQUEST\_LIMIT, IP вважається підозрілою.
4. **Логування**: Підозрілі IP записуються в лог-файл із часовою міткою.
5. **Fail2Ban**: Fail2Ban читає лог-файл і блокує IP через iptables.

**Налаштування системи**

Для роботи скрипта потрібно підготувати систему:

1. **Створення лог-файлу**:

bash

CollapseWrapCopy

sudo touch /var/log/fail2ban\_network\_flood.log

sudo chmod 666 /var/log/fail2ban\_network\_flood.log

1. **Налаштування фільтра Fail2Ban**: Створіть файл /etc/fail2ban/filter.d/network-flood.conf:

ini

CollapseWrapCopy

[Definition]

failregex = .\*Suspicious activity from <HOST>

ignoreregex =

1. **Додавання правила в Fail2Ban**: Відредагуйте /etc/fail2ban/jail.local:

ini

CollapseWrapCopy

[network-flood]

enabled = true

filter = network-flood

action = iptables-multiport[name=NetworkFlood, port="all", protocol="all"]

logpath = /var/log/fail2ban\_network\_flood.log

maxretry = 1

bantime = 3600

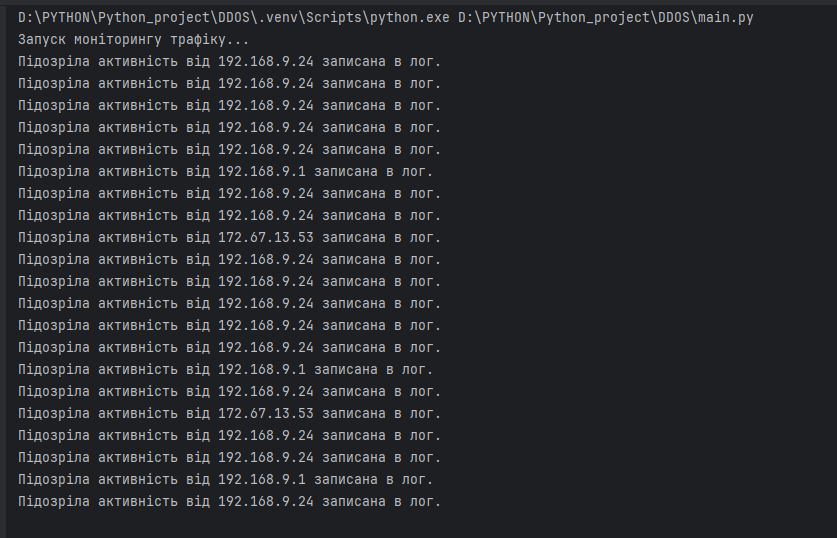
findtime = 10

1. **Перезапуск Fail2Ban**:

sudo systemctl restart fail2ban

1. **Запуск скрипта**: Скрипт потрібно запускати з правами root через необхідність доступу до мережевих пакетів:

sudo python3 network\_protection.py



**Переваги підходу**

* **Автоматизація**: Fail2Ban самостійно блокує IP, не потребуючи ручного втручання через iptables.
* **Гнучкість**: Легко налаштувати пороги (REQUEST\_LIMIT, CHECK\_INTERVAL) та додати нові протоколи (UDP, наприклад).
* **Журналювання**: Усі підозрілі дії фіксуються в лог-файлі для подальшого аналізу.
* **Ефективність проти Ping Flood**: Скрипт добре справляється з атаками типу ICMP-флуду.

**Недоліки**

* **Обмеження по масштабах**: Не захистить від розподілених атак із великою кількістю унікальних IP.
* **Помилкові спрацьовування**: Легітимні користувачі з високою активністю можуть бути заблоковані.
* **Ресурси**: Scapy може споживати багато пам’яті при інтенсивному трафіку.

**Покращення**

1. **Додавання білого списку**: Дозвольте ігнорувати певні IP (наприклад, локальні адреси).
2. **Аналіз патернів**: Перевіряйте не лише кількість запитів, а й їхній вміст чи частоту.
3. **Інтеграція з мережевими інструментами**: Використовуйте апаратні рішення для масштабного захисту.

**Висновок**

Цей скрипт у поєднанні з Fail2Ban є ефективним рішенням для захисту локальної мережі від атак типу Ping Flood та схожих DDoS-атак. Він підходить для невеликих мереж або як основа для більш складних систем захисту. Для реального використання в критичних системах рекомендується додати професійні інструменти, такі як NGFW (Next-Generation Firewall) або хмарні сервіси.