DOI 10.20535/2411-1031.2023.11.2.293760
УДК 004.056

ДАНИЇЛ ЖУРАВЧАК,
ЕДУАРД КІЙКО,
ВАЛЕРІЙ ДУДИКЕВИЧ

ВИКОРИСТАННЯ EBPF ДЛЯ ІДЕНТИФІКАЦІЇ ВІРУСІВ-ВИМАГАЧІВ, ЩО ВИКОРИСТОВУЮТЬ DNS-ЗАПИТИ DGA

У сучасному світі, де інтернет став невід'ємною частиною функціонування державних та корпоративних установ, цілісність та доступність інформації стає ключовим питанням для багатьох організацій та індивідуальних користувачів. Особливо актуальним є питання захисту від вірусів-крипторів та атак, зокрема, за допомогою DGA (Domain Generation Algorithms) – методу, який використовують зловмисники для автоматичної генерації доменних імен для комунікації клієнт-сервер (Command & Control) у екосистемі вірусів на базі протоколу DNS, що ускладнює їх виявлення та блокування через особливості використання DNS у сучасних ком’ютерних мережах. З огляду на зростання кількості атак, що використовують DGA, з'являється необхідність в розробці нових методів, що будуть швидші та зможуть аналізувати великі потоки трафіку у режимі реального часу, та забезпечать функціонал для їх виявлення та блокування. eBPF (extended Berkeley Packet Filter) — це сучасний інструмент, що дозволяє створювати невеликі програми для моніторингу та аналізу різних аспектів роботи системи в реальному часі, включаючи мережевий трафік. Дані програми виконуються безпосередньо у ядрі операційної системи та / або на рівні мережевої карти. У цьому дослідженні ми розглядаємо можливість застосування eBPF для виявлення DGA-активності в DNS-трафіку. Мета – визначити ефективність виявлення вірусів-вимагачів у режимі реального часу. Було розроблено лабораторне середовище для аналізу вірусів-вимагачів, у якому велася розробка модулів на базі eBPF, їхнє тестування та симуляція атаки.. Крім цього було налаштовано хмарне середовище аналізу даних на базі Splunk і на базі даного аналізу розроблені правила виявлення атаки типу DGA. Стаття представляє результати розробки програми на базі eBPF для аналізу DNS-трафіку, проведення атак DGA, та методи їх виявлення. Ці результати можуть стати важливим внеском у розробку стратегій захисту від маліційних атак в мережі.

Ключові слова: eBPF, DGA, DNS, шкідливе програмне забезпечення, система виявлення вторгнень, моніторинг, ядро, кібербезпека.

Постановка проблеми. З розширенням глобальної мережі Інтернет, зростанням числа підключених пристроїв і взаємопов'язаності сервісів, збільшується і потенційний ризик кібератак. Особливо актуальними стають атаки, які важко виявити та заблокувати звичайними засобами. Однією з таких атак є атаки, засновані на використанні DGA (Domain Generation Algorithms) [1].

DGA – це алгоритми, що генерують велику кількість доменних імен за короткий проміжок часу. Зловмисники використовують цей метод для обходу блокувань і виявлення маліційних доменів, створюючи тимчасові підключення до своїх командних серверів [2]. Звичайні методи блокування та виявлення часто виявляються безсилі перед таким типом атаки, оскільки кожен раз генерується новий домен, що уникає виявлення.

Тому виникає проблема: як ефективно виявляти та блокувати домени, створені за допомогою DGA, у реальному часі? Традиційні методи, такі як чорні списки доменів [3] або евристичний аналіз, можуть не бути достатньо швидкими або точними для виявлення динамічно генерованих доменів.

Також, для виявлення DGA активності неодноразово звертались до машинного навчання. Як показало дослідження Міщенко 2021 року [4], цей метод теж може бути ефективним. Однак, ефективне навчання моделей може вимагати багато часу і ресурсу.

У цьому контексті, технологія eBPF [5], яка дозволяє моніторити та аналізувати мережевий трафік в реальному часі, може стати потенційним рішенням проблеми. Але яка ефективність такого підходу? І чи може він забезпечити надійний захист від DGA-атак? Відповіді на ці питання є основною метою нашого дослідження.

Аналіз останніх досліджень і публікацій. Розробка DNS протоколу припадає на середину 1980-х років. В результаті, з розвитком мережевих і комп'ютерних технологій, розвивався і ринок шкідливого ПЗ. Це призвело до того, що в протоколі з часом почали знаходити все більше вразливостей [6]. В той же час, дослідження Бойко "Застосунок моніторингу трафіку та визначення DDoS-атак за допомогою eBPF" (2023), вказує перелік утиліт, які з'являлись з розвитком мережевих протоколів і дозволяли аналізувати трафік. Наприклад, PRTG Network Monitor є одним із перших застосунків, задачею якого є аналіз мережевого трафіку [7]. Застосунок є уніфікованим інструментом моніторингу, який може контролювати практично будь-який об'єкт, що має IP-адресу. Він складається з двох елементів: PRTG сервера і зондів. Основний PRTG сервер відповідає за налаштування та контроль даних, а зонди в свою чергу збирають дані та контролюють процеси на пристроях за допомогою датчиків.

Шлях BPF розпочався у 1992 році, коли його запровадили як зручний спосіб перехоплення пакетів для спостереження на рівні користувача. Він мав список переваг і було виявлено, що він працює в 20 разів швидше, ніж інші способи обробки пакетів на той час. З роками ентузіасти Linux розширили функціональність BPF до розширеної версії (також відома як eBPF). Тепер eBPF працює з 64-розрядними регістрами замість 32-розрядних. Програми eBPF можна приєднати до інших подій ядра, наприклад, початок виконання програми або підключення нового пристрою через USB. Функціональні можливості eBPF більше залежать від простору користувача [8].

Метою статті є дослідження інструментів технології eBPF на предмет моніторингу DNS трафіку та виявлення загроз, таких, як DNS тунель, ексфільтрація даних, DGA.

Виклад основного матеріалу дослідження.

Створення лабораторного середовища.

Для розробки нашого рішення для мережевого моніторингу в вигляді скрипта мовами Python та C, а також для проведення демонстрації в вигляді виявлення потенційно шкідливого DNS трафіку, було створене лабораторне середовище (рис. 1):

1. Хостова машина на ОС Windows.
2. Віртуальна машина на ОС Ubuntu.
3. Хмарна версія рішення для обробки і візуалізації даних Splunk Cloud.

Рисунок 1 – Архітектурна діаграма лабораторного середовища

Основні задачі хоста на базі Windows:

1. Середовище розробки.
2. Розміщення віртуальної машини на ОС Ubuntu.

Основні задачі віртуальної машини на базі Ubuntu:

1. Середовище виконання основного скрипта програми.
2. Середовище виконання тестового сценарію генерації доменів DGA.
3. Середовище виконання тестового сценарію генерації доменів DGA.

Основні задачі рішення Splunk Cloud:

1. Обробка і зберігання лог-файлів, отриманих з віртуальної машини, на якій виконувався тестовий сценарій.
2. Графічна візуалізація даних отриманих з віртуальної машини Ubuntu.
3. Аналіз отриманих результатів.

В результаті, запропоноване лабораторне середовище, дозволяє покрити всі потреби дослідження, а саме: розробка скрипта для моніторингу DNS трафіку мовами Python та C, тестування сценарію, наближеного до реальних умов (генерування трафіку DGA), а також візуалізація отриманих даних та проведення аналізу результатів дослідження.

Практична реалізація моніторингового рішення.

На базі утиліти з відкритим вихідним кодом [9], було розроблене рішення на мовах програмування C та Python, яке здатне перехоплювати всі запити та відповіді DNS, які виконуються хостом, що моніториться [10]. Крім того, в рішенні реалізований функціонал логування, що дозволяє його легку інтеграцію з рішеннями обробки, зберігання, та візуалізації лог-файлів та метрик, таких, як Splunk, Elastic, NewRelic та ін.

Для отримання інформації про мережеві пакети використовуємо функції udp_sendmsg [11] та tcp_sendmsg [12], які здійснюють їхню відправку. Існує 2 способи дістатись до аргументів функції в eBPF: позначити їх як параметри функції, або використати макрос PT_REGS_PARMx, де x – це номер аргумента (ми будемо використовувати цей метод) [13].

Відрізок програми, де використовується udp_sendmsg*:*

```
int trace_udp_sendmsg(struct pt_regs *ctx, struct sock *sk) {
  u16 sport = sk->sk_num;
  u16 dport = sk->sk_dport;

  // Processing packets only on port 53.
  // 13568 = ntohs(53);
  if (sport == 13568 || dport == 13568) {
    // Preparing the data:
    u32 saddr = sk->sk_rcv_saddr;
    u32 daddr = sk->sk_daddr;
    u64 pid_tgid = bpf_get_current_pid_tgid();
    u64 uid_gid = bpf_get_current_uid_gid();
    // Forming the key structure.
    // These strange transformations will be explained below.
    struct port_key key = {.proto = 17};
    key.saddr = htonl(saddr);
    key.daddr = htonl(daddr);
    key.sport = sport;
    key.dport = htons(dport);
```

Функціонал tcp_sendmsg буде мати ідентичний вигляд, за виключенням того, що в структурі port_key, значення буде дорівнювати 6. 17 та 6 – це коди для UDP та TCP протоколів відповідно.Ці значення знаходяться в файлі /etc/protocols. Також вказуємо порт 53, оскільки нас цікавлять тільки пакети, які будуть відправлятись на цей порт.

Наша програма буде логувати пакети та інформацію про процес, який їх надіслав, а також передавати цю інформацію в простір користувача:

```
// By the key, look for a value in the eBPF table:
        struct port_val *p_val;
        p_val = proc_ports.lookup(&key);
        // If no value is found, then we have no information about the
        // process and there is no point in continuing:
        if (!p_val) {
            return 0;
        }
        // Network device index:
        p_val->ifindex = skb->ifindex;
        // Transmit the structure with the process information along with
        // skb->len bytes sent to the socket:
        dns_events.perf_submit_skb(skb, skb->len, p_val,
                                   sizeof(struct port_val));
        return 0;
    } //dport == 53 || sport == 53
  } //ethernet->type == ETH_P_IP
return 0;
```

Крім того, для TCP пакетів, ми перевіряємо флаги, для того, щоб відфільтрувати пакети, які не несуть даних(пакети типу SYN, ACK, і т. д.).

На стороні Python ми виконуємо 3 кроки: завантажуємо C-код в Linux ядро, отримуємо з нього дані, та опрацьовуємо їх.

Найважчий процес тут - це опрацьовування даних. Для цього розбираємо пакет на поля, які можуть бути опрацьвані Python. Для розбору пакету, використовуємо написану функцію print_dns, для того, щоб зменшити залежність від сторонніх Python-модулів.

```
def print_dns(cpu, data, size):
import ctypes as ct
class SkbEvent(ct.Structure):
_fields_ = [
("ifindex", ct.c_uint32),
("pid", ct.c_uint32),
("tgid", ct.c_uint32),
("uid", ct.c_uint32),
("gid", ct.c_uint32),
("comm", ct.c_char * 64),
("raw", ct.c_ubyte * (size - ct.sizeof(ct.c_uint32 * 5) - ct.sizeof(ct.c_char * 64)))
]
```

Також, використовуємо Python бібліотеку logging для виводу даних в текстовий файл.

Крім того, до існуючої утиліти додаємо файл ebpf-dns-monitor.service, який дозволить скрипту запускатись автоматично, від імені свого сервісу. Вміст файлу:

```
[Unit]
Description=Lightweight tool for DNS traffic monitor
After=multi-user.target

[Service]
Type=simple
Restart=always
ExecStart=/usr/bin/python3 /etc/ebpf-dns-monitor/ebpf-dns-main.py

[Install]
WantedBy=multi-user.target
```

Таким чином, надаємо скрипту запускатись автоматично при старті ОС, що прибирає необхідність запускати його вручну і дозволяє йому покривати весь час активності сервера. Таким чином, утиліта підлаштована під запити бізнесу.

Запускаємо утиліту за допомогою systemctl:

```
sudo systemctl enable ebpf-dns-monitor.service
sudo systemctl start ebpf-dns-monitor.service
```

Приклад вихідних даних:

```
COMM=systemd-resolve PID=645 TGID=645 DEV=ens33 PROTO=UDP SRC=192.168.40.130 DST=192.168.40.2 SPT=55146 DPT=53 UID=101 GID=103 DNS_QR=0 DNS_NAME=www.google-analytics.com. DNS_TYPE=A
COMM=systemd-resolve PID=645 TGID=645 DEV=ens33 PROTO=UDP SRC=192.168.40.130 DST=192.168.40.2 SPT=35062 DPT=53 UID=101 GID=103 DNS_QR=0 DNS_NAME=www.google-analytics.com. DNS_TYPE=AAAA
```

Симуляція генерування DNS запитів на згенеровані DGA домени та їхнє виявлення. Тестування рішення.

Для того, щоб провести демонстрацію розробленого рішення, запустимо скрипт, який симулює DGA мережевий трафік, з бібліотеки з відкритим вихідним кодом Domain Generation Algorithms/Bazarbackdoor [14]. Спочатку запустимо розроблений скрипт eBPF_dns_main.py, для отримання картини мережевого трафіку в межах норми, а потім запустимо скрипт dga.py з бібліотеки Domain Generation Algorithms для симуляції потенційно шкідливої DNS активності.

Для візуалізації отриманих даних, використаємо платформу для зберігання і обробки великих даних, Splunk.

Відобразимо демонстраційний часовий проміжок роботи утиліти (рис. 2).

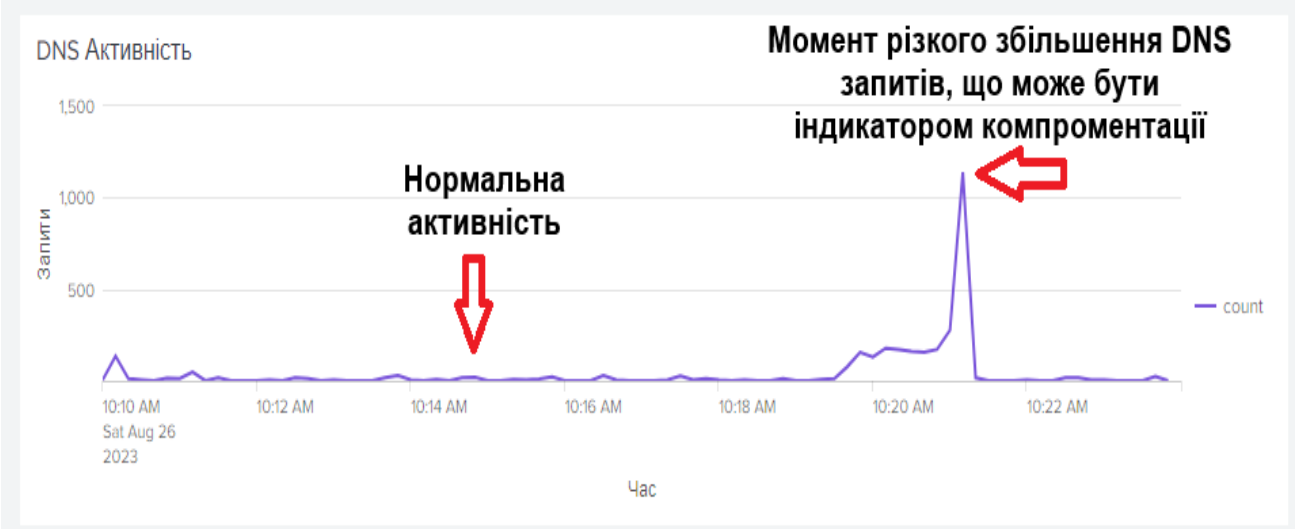

Рисунок 2 – Активність мережевого трафіку DNS: Виявлення аномалії атаки

Розслідуємо потенційний інцидент, за допомогою панелей, які покажуть домени, на які були здійснені запити та процеси, якими вони були запущені в даний проміжок часу (рис. 3).

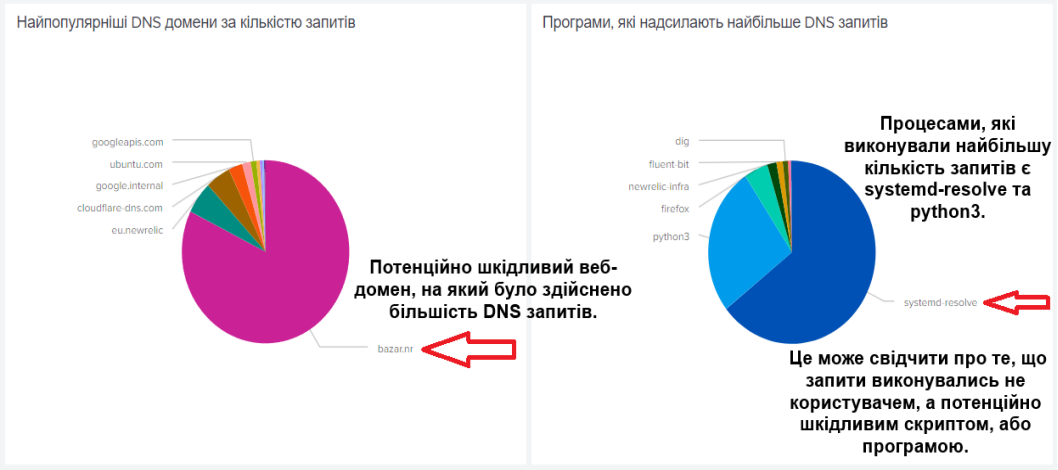

Рисунок 3 – Активність мережевого трафіку DNS: Статистика запитів

В системі Splunk спробуємо знайти приклади запитів, які були здійснені на підозрілий домен. Для цього здійснюємо пошук за ключовим словом “bazar.nr” (рис. 4).

Час	Процес	ID Процесу	Протокол	Порт	Домен
2023-08-26 10:21:16.000	python3	7031	UDP	53	qepionex.bazar.nr.
2023-08-26 10:21:15.000	python3	7031	UDP	53	qepioncu.bazar.nr.
2023-08-26 10:21:15.000	python3	7031	UDP	53	qefosoli.bazar.nr.
2023-08-26 10:21:14.000	python3	7031	UDP	53	qefosoty.bazar.nr.
2023-08-26 10:21:13.000	python3	7031	UDP	53	qefosouf.bazar.nr.
2023-08-26 10:21:12.000	python3	7031	UDP	53	qefosovu.bazar.nr.
2023-08-26 10:21:12.000	python3	7031	UDP	53	qefosoex.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefosocu.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoudli.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoudty.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefouduf.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoudvu.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoudex.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoudcu.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoelli.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoelty.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoeluf.bazar.nr.
2023-08-26 10:21:11.000	python3	7031	UDP	53	qefoelvu.bazar.nr.

Рисунок 4 – Активність мережевого трафіку DNS:
Запити до потенційно шкідливого серверу

В результаті пересвідчуємось, що дійсно велика кількість схожих запитів була відправлена на невідомий нам домен bazar.nr, за допомогою Python скрипта в короткий проміжок часу. Крім того, візуально можна визначити, що ці домени, на які робляться запити, скоріш за все, є доменами згенерованими за технологією DGA. Таким чином, виявляємо потенційно шкідливе ПЗ на хост-системі Linux за допомогою створеного скрипта мовами C та Python, який використовує інструменти eBPF для логування мережевого трафіку DNS.

В реальній ситуації, наступними кроками спеціаліста з кібербезпеки, який виявив цей інцидент має бути подальше його розслідування, виявлення початкового процесу або програми, яка спричинила ці запити локально, на стороні Linux сервера або комп’ютера. Крім того, гарним кроком буде перевірка домену на легітимність, або приналежність до відомих вірусів, або груп кіберзловмисників, і потенційно, блокування всіх запитів до цього домену на рівні роутера, або свіча(якщо така можливість присутня).

В нашому випадку ми просто пересвідчились, що написаний скрипт коректно виконує свою функцію і за допомогою рішення для візуалізації даних, спроможний виявляти потенційно шкідливу активність, наприклад, автоматично згенеровані запити на DGA домени.

Висновки. Дослідження показало, що використання інструментів eBPF для моніторингу мережевого трафіку є економним в використанні ресурсів, ефективним та зручним методом для виявлення цілого ряду вразливостей і векторів кібератак. Нам вдалось використати функціонал технології eBPF для створення утиліти, засобами мов програмування C, для приєднання програми до мережевого сокета, і отримання інформації про мережевий пакет, мови програмування Python, для подальшої обробки отриманого мережевого пакета, переведення його в зручний для читання формат, а також запис отриманих даних в лог-файл. Також було використано скрипт з відкритим кодом для генерації DNS запитів, створених за технологією DGA, для демонстраційного сценарію. Аналіз отриманих результатів в середовищі обробки, зберігання та аналізу даних, Splunk Cloud, показав, що подібний скрипт може використовуватись для виявлення загроз, які експлуатують протокол DNS.

Перспективи подальших досліджень. Крім того, eBPF є відносно новою розробкою, яка ще не знайшла свого місця в ніші кібербезпекових рішень, орієнтованих на виявлення загроз локально, на стороні сервера або комп'ютера. Виходячи з цього, можна зробити висновок, що запит на дослідження рішень для моніторингу на базі eBPF з часом буде тільки зростати, як в науковому, так і в державному і корпоративному сегментах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] А. Ю. Божко, та О. М. Барановський, "Методи виявлення DGA в DNS-запитах", на *XIX Всеукр. наук.-практ. конф. студ., асп. та мол. вч. Теоретичні і прикладні проблеми фізики, математики та інформатики*, Київ, 2021, с. 309-311. [Електронний ресурс]. Доступно: https://ela.kpi.ua/bitstream/123456789/52980/1/%28309-311%29_Bozhko.pdf. Дата звернення: Серп. 11, 2023.

[2] А. Ю. Божко, "Виявлення центрів керування шкідливого ПЗ за допомогою аналізу DNS трафіку", дипл. роб., НТУУ КПІ, Київ, 2021. [Електронний ресурс]. Доступно: https://ela.kpi.ua/bitstream/123456789/57102/1/Bozhko_Bakalavr.pdf. Дата звернення: Серп. 11, 2023.

[3] К. Ю. Бобровнікова, "Інформаційна технологія виявлення бот-мереж у корпоративних мережах на основі аналізу DNS-трафіка", дис. канд. наук., ТНТУ, Тернопіль, 2016. [Електронний ресурс]. Доступно: https://elartu.tntu.edu.ua/bitstream/123456789/18599/5/Avtoreferat_Bobrovnikova_K_JU.pdf. Дата звернення: Серп. 14, 2023.

[4] Н. В. Міщенко, "Інформаційна технологія розпізнавання трафіку алгоритмів генерації доменів на основі глибинного машинного навчання", квал. роб., СДУ, Суми, 2021. [Електронний ресурс]. Доступно: https://essuir.sumdu.edu.ua/bitstream-download/123456789/84128/1/Mishchenko_mag_rob.doc.pdf. Дата звернення: Серп. 15, 2023.

[5] Д. О. Зіньковський, "Розробка програмного забезпечення для фільтрації та трасування трафіку з використанням технології BPF в Linux", квал. роб., НТУ ДП, Дніпро, 2021. [Електронний ресурс]. Доступно: https://ir.nmu.org.ua/handle/123456789/158957. Дата звернення: Серп. 14, 2023.

[6] Д. Д. Бойко, та Є. О. Давиденко, "Засоби обробки та аналізу мережевих пакетів в ОС Linux", на *XVII Міжнар. наук. конф. Ольвійський форум – 2023: стратегії країн Причорноморського регіону в геополітичному просторі*, Миколаїв, 2023.

[7] Д. Д. Бойко, "Застосунок моніторингу трафіку та визначення DDoS-атак за допомогою eBPF", квал. роб., ЧНУ, 2023. [Електронний ресурс]. Доступно: https://krs.chmnu.edu.ua/jspui/handle/123456789/2893. Дата звернення: Серп. 14, 2023.

[8] S. Tesliuk, "Monitoring network traffic and detecting attacks using eBPF", bach. thesis, UCU, Lviv, 2021. [Online]. Available: http://www.er.ucu.edu.ua:8080/handle/1/2877. Accessed on: Sep. 16, 2023.

[9] oghie/final_code_eBPF_dns.py. [Online]. Available: https://gist.github.com/oghie/b4e3accf1f87afcb939f884723e2b462. Accessed on: Sep. 19, 2023.

[10] eduard-daily/eBPF-dns-monitor. [Online]. Available: https://github.com/eduard-daily/eBPF-dns-monitor. Accessed on: Sep. 28, 2023.

[11] linux/net/ipv4/udp.c. [Online]. Available: https://github.com/torvalds/linux/blob/master/net/ipv4/udp.c#L1057. Accessed on: Sep. 28, 2023.
[12] linux/net/ipv4/tcp.c. [Online]. Available: https://github.com/torvalds/linux/blob/master/net/ipv4/tcp.c#L1328. Accessed on: Sep. 28, 2023.
[13] H. Nurkholish, "A Deep Dive into eBPF: Writing an Efficient DNS Monitoring". [Online]. Available: https://medium.com/@nurkholish.halim/a-deep-dive-into-ebpf-writing-an-efficient-dns-monitoring-2c9dea92abdf. Accessed on: Sep. 12, 2023.
[14] baderj/domain_generation_algorithms/bazarbackdoor/. [Online]. Available: https://github.com/baderj/domain_generation_algorithms/tree/master/bazarbackdoor. Accessed on: Sep. 28, 2023.

Стаття надійшла до редакції 30.09.2023.

REFERENCE

[1] A. Bozhko, and O. Baranovsky, "Methods of detecting DGA in DNS requests", in *Proc. XIX All-Ukr. scien. and pract. conf. stud., grad. stud. and jun. scient. Theoretical and applied problems of physics, mathematics and computer science*, Kyiv, 2021, pp. 309-311. [Online]. Available: https://ela.kpi.ua/bitstream/123456789/52980/1/%28309-311%29_Bozhko.pdf. Accessed: Aug. 11, 2023.
[2] A. Bozhko, "Detection of malware command centers using DNS traffic analysis", bach. thesis, NTUU, Kyiv, 2021. [Online]. Available: https://ela.kpi.ua/bitstream/123456789/57102/1/Bozhko_Bakalavr.pdf. Accessed: Aug. 11, 2023.
[3] K. Bobrovnikova, "Information technology for botnet detection in corporate networks based on DNS traffic analysis", PhD. thesis, TNTU, Ternopil, 2017. [Online]. Available: https://elartu.tntu.edu.ua/bitstream/123456789/18599/5/Avtoreferat_Bobrovnikova_K_JU.pdf Accessed: Aug. 11, 2023.
[4] N. Mischenko, "Information Technology for recognizing Domain Generation Algorithm traffic based on Deep Machine Learning", M.S. thesis, SSU, Sumy, 2021. [Online]. Available: https://essuir.sumdu.edu.ua/bitstream-download/123456789/84128/1/Mishchenko_mag_rob.doc.pdf. Accessed: Aug. 15, 2023.
[5] D. Zinkovsky, "Development of Software for traffic filtering and tracing using BPF technology in Linux", bach. thesis, DNTU, Dnipro, 2021. Accessed: Aug. 14, 2023. [Online]. Available: https://ir.nmu.org.ua/handle/123456789/158957. Accessed: Aug. 14, 2023.
[6] D. Boiko, and Y. Davydenko, "Tools for processing and analyzing network packets in Linux OS", in *Proc. XVII Intl. scient. conf. Olevs'kyi Forum – 2023: Strategies of Black Sea Region Countries in Geopolitical Space*, Mykolaiv, 2023.
[7] D. Boiko, "Traffic monitoring application and DDoS attack detection using eBPF", bach. thesis, BSNU, Mykolaiv, 2023. [Online]. Available: https://krs.chmnu.edu.ua/jspui/handle/123456789/2893. Accessed: Aug. 16, 2023.
[8] S. Tesliuk, "Monitoring network traffic and detecting attacks using eBPF", bach. thesis, UCU, Lviv, 2021. [Online]. Available: http://www.er.ucu.edu.ua:8080/handle/1/2877. Accessed on: Sep. 16, 2023.
[9] oghie/final_code_eBPF_dns.py. [Online]. Available: https://gist.github.com/oghie/b4e3accf1f87afcb939f884723e2b462. Accessed on: Sep. 19, 2023.
[10] eduard-daily/eBPF-dns-monitor. [Online]. Available: https://github.com/eduard-daily/eBPF-dns-monitor. Accessed on: Sep. 28, 2023.
[11] linux/net/ipv4/udp.c. [Online]. Available: https://github.com/torvalds/linux/blob/master/net/ipv4/udp.c#L1057. Accessed on: Sep. 28, 2023.
[12] linux/net/ipv4/tcp.c. [Online]. Available: https://github.com/torvalds/linux/blob/master/net/ipv4/tcp.c#L1328. Accessed on: Sep. 28, 2023.
[13] H. Nurkholish, "A Deep Dive into eBPF: Writing an Efficient DNS Monitoring". [Online]. Available: https://medium.com/@nurkholish.halim/a-deep-dive-into-ebpf-writing-an-efficient-dns-monitoring-2c9dea92abdf. Accessed on: Sep. 12, 2023.

[14] baderj/domain_generation_algorithms/bazarbackdoor/. Online]. Available: https://github.com/baderj/domain_generation_algorithms/tree/master/bazarbackdoor. Accessed on: Sep. 28, 2023.

DANYIL ZHURAVCHAK,
EDUARD KIIKO,
VALERY DUDYKEVYCH

USING EBPF TO IDENTIFY RANSOMWARE THAT USE DGA DNS QUERIES

In today's world, where the Internet has become an integral part of the functioning of government and corporate institutions, the integrity and availability of information is becoming a key issue for many organizations and individual users. The issue of protection against crypto viruses and attacks, in particular, using DGA (Domain Generation Algorithms), a method used by attackers to automatically generate domain names for client-server (Command & Control) communication in the DNS-based virus ecosystem, is particularly relevant, making it difficult to detect and block them due to the way DNS is used in modern computer networks.

Given the growing number of attacks that use DGA, there is a need to develop new methods that are faster and can analyze large traffic flows in real time and provide functionality for detecting and blocking them. eBPF (Extended Berkeley Packet Filter) is a modern tool that allows you to create small programs to monitor and analyze various aspects of the system in real time, including network traffic. These programs are executed directly in the operating system kernel and/or at the network card level. In this study, we consider the possibility of using eBPF to detect DGA activity in DNS traffic. The goal is to determine the effectiveness of real-time ransomware detection. We developed a ransomware analysis lab environment where we developed eBPF-based modules, tested them, and simulated an attack. In addition, a cloud-based data analysis environment based on Splunk was set up and rules for detecting a DGA attack were developed based on this analysis. This article presents the results of developing an eBPF-based program for analyzing DNS traffic, conducting DGA attacks, and methods for detecting them. These results can be an important contribution to the development of strategies to protect against malicious attacks in the network.

Keywords: eBPF, DGA, DNS, malware, IDS, monitoring, kernel, cybersecurity.

Журавчак Даниїл Юрійович, аспірант, асистент кафедри захисту інформації Національного університету “Львівська політехніка”, Львів Україна, ORCID 0000-0003-4989-0203, danyil.y.zhuravchak@lpnu.ua.

Кійко Едуард Віталійович, студент Національного університету “Львівська політехніка”, Львів, Україна, ORCID 0009-0004-5108-0511, eduard.kiiko1@gmail.com.

Дудикевич Валерій Богданович, д.т.н., професор, завідувач кафедри захисту інформації Національного університету “Львівська політехніка”, Львів Україна, ORCID 0000-0001-8827-9920, valerii.b.dudykevych@lpnu.ua.

Zhuravchak Danyil, postgraduate student, teaching assistant, Department of information security, Lviv Polytechnic National University, Lviv, Ukraine.

Kiiko Eduard, student, Department of information security, Lviv Polytechnic National University, Lviv, Ukraine.

Dudykevych Valeriy, doctor of engineering, professor, head of the Department of information security, Lviv Polytechnic National University, Lviv, Ukraine.