МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

СУЧАСНІ ПРОБЛЕМИ НАУКОВОГО ЗАБЕЗПЕЧЕННЯ ЕНЕРГЕТИКИ

Матеріали XXII Міжнародної науково-практичної конференції молодих вчених і студентів
м. Київ, 22–25 квітня 2025 року

ТОМ 2

Київ- 2025

УДК 620.9(062)+621.311(062)
С91

Сучасні проблеми наукового забезпечення енергетики. У 2-х т. : Матеріали XXII Міжнар. наук.-практ. конф. молод. вчених і студ., м. Київ, 22–25 квіт. 2025 р. – Київ : КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2025. – Т. 2. – 284 с.

ISBN _____(Заг.)
ISBN _____(Т. 2)

Подано тези доповідей XXII Міжнародної науково-практичної конференції молодих вчених і студентів «Сучасні проблеми наукового забезпечення енергетики» за напрямами: автоматизація теплоенергетичних процесів, аспекти розвитку інженерії прогорамного забезпечення в енергетиці, комп'ютерний еколого-економічний моніторинг та геометричне моделювання процесів і систем, інформаційні технології та комп'ютерне моделювання.

Друкується в авторській редакції за рішенням Вченої ради Навчально-наукового інституту атомної та теплової енергетики Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» (протокол № 10 від 31 березня 2025 р.)

ISBN _____(Заг.)
ISBN _____(Т. 2)

СЕКЦІЯ № 10

Інформаційні технології та комп’ютерне моделювання

УДК 004.94

[1] Бакалаврант 4 курсу Ковальов О.О.
[1] Асист. Кардашов О.В.
https://scholar.google.com.ua/citations?user=gtnZz4EAAAAJ&hl=uk

[1] КПІ ім. Ігоря Сікорського

ПОРІВНЯННЯ ТЕХНОЛОГІЙ ЗАХОПЛЕННЯ МЕРЕЖЕВОГО ТРАФІКУ ДЛЯ РОЗРОБКИ СИСТЕМИ ГЛИБОКОГО АНАЛІЗУ ПАКЕТІВ

Постановка проблеми та її актуальність. У сучасному світі інформаційних технологій зростає потреба у високопродуктивних та безпечних системах аналізу мережевого трафіку, особливо коли йдеться про застосування технологій deep packet inspection (DPI, глибокий аналіз або інспекція пакетів, де пакет є будь яким блоком даних відносно рівнів системи OSI) для забезпечення кібербезпеки, моніторингу мереж і оперативного виявлення аномалій. Використання DPI полягає не лише в обробці заголовків, а й самих даних, payload (корисне навантаження) та визначення протоколів.

Основною задачею є захоплення потоків пакетів із високою швидкістю та їх подальша обробка, що вимагає не лише високої продуктивності, але й надійності та відсутності вразливостей, пов'язаних з помилками управління пам'яттю. Особливо актуальною стає проблема безпеки доступу до пам'яті, оскільки традиційні реалізації на мовах, таких як C або C++, часто стикаються з ризиком переповнення буфера, витоків пам'яті та інших помилок, які можуть бути використані зловмисниками. Сучасні вимоги до програмного забезпечення також передбачають відсутність залежності від збору сміття (відсутністю Garbage Collector) з метою мінімізації накладних витрат у реальному часі, що є особливо важливим при обробці великого обсягу даних у мережевих застосунках. Згідно аналізу [1] Агентства з кібербезпеки та захисту інфраструктури США, рекомендується використовувати «безпечні» мови, такі як Rust, Java, C#, Go і так далі. Лише мова Rust з цього переліку не має Garbage Collector. Таким чином, при виборі технології треба орієнтуватись на два основних фактори: швидкодія та можливість використовувати мову Rust.

Аналіз останніх досліджень. За останнє десятиліття було розроблено кілька підходів до захоплення пакетів, серед яких найбільш відомими є використання pf_ring, eBPF та бібліотек для роботи з pcap, а саме реалізації на основі Rust (libpnet).

pf_ring – це технологія, яка використовує спеціалізований драйвер для забезпечення високошвидкісного захоплення пакетів, дозволяючи обробляти величезні об'єми мережевого трафіку. Її продуктивність підтверджується широким застосуванням у високонавантажених мережах [2], проте реалізація на базі традиційних мов програмування не гарантує повної безпеки доступу до пам'яті, що може створювати ризики при експлуатації системи.

eBPF, або Extended Berkeley Packet Filter, інтегрований безпосередньо в ядро сучасних Linux-систем, відкриває нові можливості для безпечного та гнучкого аналізу мережевого трафіку. Завдяки вбудованій віртуальній машині, яка перевіряє виконуваний байткод перед його запуском, eBPF забезпечує високу ступінь захищеності, що є критично важливим при розгортанні систем глибокого аналізу пакетів. Проте, незважаючи на безпечність у виконанні, eBPF може вимагати певного часу на адаптацію та навчання розробників, оскільки синтаксис та моделі виконання відрізняються від традиційних підходів. Завдяки даній технології можна ефективно обробляти фрейми – читати їх, відхиляти, тощо. [3]

Бібліотека pcap (libpcap [4]), зокрема її реалізація libpnet (написана на Rust), поєднує популярність класичного підходу з перевагами сучасної мови програмування Rust:

володіння пам’яттю, перевірка типів під час компіляції та відсутність залежності від збирача сміття. Це означає, що навіть у високонавантажених умовах система може ефективно працювати без ризику витоків пам’яті або інших помилок, характерних для реалізацій на C. Ці переваги особливо актуальні з огляду на сучасні вимоги до безпеки програмного забезпечення, де будь-яка вразливість може призвести до серйозних наслідків. Аналіз останніх досліджень демонструє, що інтеграція «memory-safe» бібліотек у системи DPI може не тільки підвищити надійність, але й забезпечити модульність та масштабованість рішень.

Формулювання мети. Метою даної роботи є порівняння технологій захоплення мережевого трафіку – pf_ring, eBPF та pcap із реалізацією через Rust libpnet – з акцентом на пошук оптимального балансу між високою продуктивністю та забезпеченням безпеки доступу до пам’яті. Основною задачею є порівняння підходів до перехоплення мережевих фреймів для подальшої обробки, у даному випадку – для розробки системи глибокого аналізу пакетів, що накладає певні обмеження, а саме здатність обробляти великі обсяги даних без компромісів у безпеці та стабільності роботи програмного забезпечення.

Основна частина. Розглядаючи pf_ring, відразу слід зазначити, що ця технологія реалізована виключно мовою C та працює у ядерному режимі. Такий підхід дозволяє забезпечити максимальну швидкодію завдяки прямому доступу до мережевого інтерфейсу та мінімальним накладним витратам. Однак, використання мови програмування C, як було зазначено вище, створює безпекові ризики, які можуть бути неприпустимими в певних проектах.

pf_ring опитує фрейми від мережевих адаптерів за допомогою Linux NAPI (механізм обробки подій). NAPI копіює пакети з мережевого адаптера в циклічний буфер, а потім програма користувача зчитує пакети з кільця. У цьому сценарії є два клієнти які займаються полінгом (опитуванням), це програма розробника та NAPI (Рис. 2.)[5]. Перевага полягає в тому, що дана технологія може розподіляти вхідні пакети на кілька кілець (отже, на кілька програм) одночасно. Таким чином, досягається висока швидкодія – дану технологію можна використовувати в промислових масштабах, якщо немає серйозних вимог до кібербезпеки, наприклад, в локальних мережах. Хоча, загалом, технологія перевірена часом, але все ж, з огляду до попередніх вимог вона не може вважатись безпечною в плані доступу до пам’яті. Реалізація на Rust відсутня.

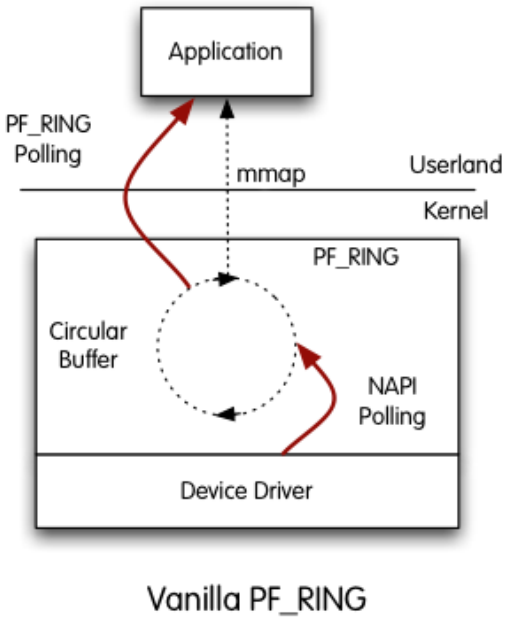

Рисунок 1 – Загальна архітектура технології PF_RING

В eBPF, який також функціонує в ядерному режимі, надає розробникам можливість завантажувати програми безпосередньо в ядро Linux. Завдяки верифікації байткоду перед виконанням, eBPF забезпечує високий рівень безпеки при обробці мережевого трафіку. Він працює у привілейованому контексті та використовує набір допоміжних функцій для ефективної обробки мережевих подій і трасування системних процесів у реальному часі. Пісочниця використовується для безпечного й ефективного розширення можливостей ядра без необхідності змінювати вихідний код ядра чи завантажувати модулі ядра [6]. Бібліотека aya-rs дозволяє використовувати eBPF у Rust, проте значна частина коду позначена як unsafe (небезпечний код, не має гарантій від компілятору), що створює потенційні ризики роботи з пам'яттю, хоч і забезпечує високу швидкодію. eBPF можна вважати компромісом між продуктивністю та гарантіями безпеки, однак технологія працює лише на Linux.

У свою чергу, бібліотека pcap з реалізацією на Rust (libpnet) працює в просторі користувача. На відміну від інших вищевказаних технологій, робота з фреймами в libpnet не може вестись «на льоту» – тому, умовний брандмауер написати з нею неможливо. Робота ведеться зі скопійованими даними, тобто, їх можна лише читати. Це, звичайно ж, впливає на швидкодію – до схожих механізмів роботи як у інших технологій, додається input/output затримка. Також, можна формувати власні фрейми, пакети, сегменти тощо, і відправляти їх, але це не стосується поставленої задачі. Є значна перевага – бібліотека може використовуватись на різних платформах завдяки чітко визначеному API. Наприклад, можна використовувати WinPCap або npcap для роботи на Windows.

Висновки. Порівняння цих технологій дозволяє зробити висновок, що вибір платформи для захоплення мережевого трафіку має враховувати компроміс між продуктивністю та гарантіями безпеки. У сучасних умовах, коли безпека доступу до пам'яті є критичною вимогою для запобігання експлуатації вразливостей, рішення на базі libpnet можуть бути привабливими для розробки систем глибокої інспекції пакетів, незважаючи на деяке зниження швидкодії. Хоча, якщо планується використовувати систему в умовах високого навантаження, варто розглянути інші варіанти, або компромісний – eBPF, або загалом безпечний для використання pf_ring, але не рекомендований до використання в системах, де є жорсткі вимоги до безпеки даних.

Перелік посилань:

1. Cybersecurity and Infrastructure Security Agency. Exploring memory safety in critical open source projects. www.cisa.gov. URL: https://www.cisa.gov/sites/default/files/2024-06/joint-guidance-exploring-memory-safety-in-critical-open-source-projects-508c.pdf (date of access: 27.02.2025).
2. The Improvement of Network Performance by Using the Technique of PF-RING Zero-copy to Optimize Spark Streaming / F. LIU et al. International Conference on Computer Networks and Communication Technology (CNCT 2016), Xiamen, China, 16–18 December 2016. Paris, France, 2017. URL: https://doi.org/10.2991/cnct-16.2017.9 (date of access: 25.02.2025).
3. Zhuravchak D., Kiiko E., Dudykevych V. Using EBPF to identify ransomware that use DGA DNS queries. Collection "Information Technology and Security". 2023. Vol. 11, no. 2. P. 166–174. URL: https://doi.org/10.20535/2411-1031.2023.11.2.293760 (date of access: 02.03.2025).
4. Snort 2.0 intrusion detection / J. Faircloth et al. Syngress, 2003. 550 p.
5. ntop. PF_RING - high-speed packet capture, filtering and analysis. https://www.ntop.org. URL: https://www.ntop.org/products/packet-capture/pf_ring/ (date of access: 19.02.2025).
6. eBPF Foundation. EBPF documentation. URL: https://ebpf.io/what-is-ebpf/ (date of access: 22.02.2025).