ISSN 2219-9365

https://doi.org/10.31891/2219-9365-2026-86-2
УДК 004.9

МЕДЗАТИЙ Дмитро
Хмельницький національний університет
https://orcid.org/0009-0004-3247-6406
e-mail: medzatyi@gmail.com
МАРЦЕНЮК Андій
Хмельницький національний університет
https://orcid.org/0009-0002-2545-3586
e-mail: andry.martsenyk@gmail.com

МЕТОД МОНІТОРИНГУ ПАРАМЕТРІВ КОМП'ЮТЕРНОЇ МЕРЕЖІ В РЕАЛЬНОМУ ЧАСІ

У даній статті проведено комплексне дослідження та розробку методу моніторингу параметрів комп'ютерної мережі в режимі реального часу, що є критично важливим в умовах глобальної цифровізації та зростання складності мережевих архітектур. Автором обґрунтовано, що традиційні підходи, засновані на періодичному зборі статистичних даних, не здатні забезпечити необхідну швидкість реакції на динамічні зміни трафіку та сучасні кіберзагрози. Особливу увагу приділено аналізу недоліків існуючих систем, які використовують повне копіювання пакетів із простору ядра в простір користувача, що призводить до надмірного споживання ресурсів процесора та пам'яті. Наукова новизна роботи полягає у запропонуванні багаторівневої моделі вимірювання характеристик TCP-з'єднань на основі технології eBPF (extended Berkeley Packet Filter). Цей підхід дозволяє здійснювати фільтрацію та первинну агрегацію метрик безпосередньо у просторі ядра операційної системи, мінімізуючи накладні витрати на копіювання даних. Розроблений метод забезпечує точне вимірювання затримки (RTT) двома способами: шляхом аналізу фаз встановлення сесії (SYN/SYN-ACK) та через обробку опцій TCP Timestamp. Математично формалізовано процеси ідентифікації мережевих потоків, визначення напрямку трафіку, оцінки кількості проміжних вузлів через TTL та обчислення рівня втрат пакетів. Програмна реалізація методу базується на комбінованому використанні мов C та Python, де низькорівнева обробка виконується в ядрі, а високорівневий аналіз, діагностика аномалій та інтеграція з системами збору метрик (Prometheus) – у користувацькому просторі. Окремо висвітлено механізми виявлення ретрансмісій та класифікації мережевих аномалій за допомогою набору діагностичних правил. Запропоноване рішення демонструє високу ефективність на пристроях із обмеженими апаратними ресурсами, зокрема на одноплатних комп'ютерах, забезпечуючи стабільність моніторингу навіть за умов високої інтенсивності трафіку.

Ключові слова: моніторинг мережі, реальний час, eBPF, TCP/IP, RTT, простір ядра, простір користувача, мережеві аномалії, ретрансмісія, QoS.

MEDZATYI Dmytro, MARTSENIUK Andrii
Khmelnytskyi National University

METHOD OF MONITORING COMPUTER NETWORK PARAMETERS IN REAL TIME

This article presents a comprehensive study and development of a method for monitoring computer network parameters in real time, which is critically important in the context of global digitalization and the increasing complexity of network architectures. The author substantiates that traditional approaches based on periodic collection of statistical data are unable to provide the necessary speed of response to dynamic traffic changes and modern cyber threats. Particular attention is paid to the analysis of the shortcomings of existing systems that use full copying of packets from kernel space to user space, which leads to excessive consumption of processor and memory resources. The scientific novelty of the work lies in proposing a multi-level model for measuring TCP connection characteristics based on eBPF (extended Berkeley Packet Filter) technology. This approach allows filtering and primary aggregation of metrics directly in the kernel space of the operating system, minimizing the overhead of data copying. The developed method provides accurate measurement of delay (RTT) in two ways: by analyzing the session establishment phases (SYN/SYN-ACK) and by processing TCP Timestamp options. The processes of identifying network flows, determining the direction of traffic, estimating the number of intermediate nodes via TTL, and calculating the packet loss rate are mathematically formalized. The software implementation of the method is based on the combined use of the C and Python languages, where low-level processing is performed in the kernel, and high-level analysis, anomaly diagnostics, and integration with metrics collection systems (Prometheus) are performed in user space. The mechanisms of detecting retransmissions and classifying network anomalies using a set of diagnostic rules are separately highlighted. The proposed solution demonstrates high efficiency on devices with limited hardware resources, in particular on single-board computers, ensuring stable monitoring even under conditions of high traffic intensity.

Keywords: network monitoring, real-time, eBPF, TCP/IP, RTT, kernel space, user space, network anomalies, retransmission, QoS.

Стаття надійшла до редакції / Received 02.03.2026
Прийнята до друку / Accepted 07.04.2026
Опубліковано / Published 31.05.2026

ISSN 2219-9365

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Актуальність розробки та впровадження методів моніторингу параметрів комп'ютерної мережі в реальному часі зумовлена стрімкою цифровізацією всіх сфер людської діяльності, де мережева інфраструктура виступає фундаментальним базисом для обміну даними. У сучасних умовах навіть короткочасна затримка або збій у роботі мережі можуть призвести до критичних фінансових втрат, репутаційних ризиків та зупинки стратегічно важливих бізнес-процесів. Традиційні методи аналізу, що базуються на періодичному зборі статистичних даних, вже не здатні забезпечити необхідну швидкість реакції на динамічні зміни в трафіку, що вимагає переходу до безперервного спостереження [1].

Зростання складності мережевих архітектур, включаючи гібридні хмарні середовища, програмно-конфігуровані мережі (SDN) та інтернет речей (IoT), створює нові виклики для систем адміністрування. Велика кількість різнорідних пристроїв генерує колосальні обсяги даних, які необхідно обробляти миттєво для підтримки стабільності з'єднань. Моніторинг у реальному часі дозволяє виявляти вузькі місця в інфраструктурі ще до того, як вони стануть причиною повної відмови системи, що забезпечує високу доступність сервісів для кінцевих користувачів [2].

Одним із ключових аспектів актуальності є питання кібербезпеки, оскільки сучасні загрози, такі як DDoS-атаки або несанкціоноване проникнення, розвиваються надзвичайно швидко. Тільки інструменти моніторингу в реальному часі здатні зафіксувати аномальні сплески активності або підозрілу поведінку пакетів даних у момент їх виникнення. Це дає можливість автоматизованим системам захисту негайно ізолювати загрозу, мінімізуючи потенційні збитки від витоку конфіденційної інформації чи пошкодження цілісності даних [3].

Розвиток мультимедійних технологій, зокрема потокового відео високої чіткості, IP-телефонії та онлайн-конференцій, висуває жорсткі вимоги до якості обслуговування (QoS). Параметри затримки (latency), джитера (jitter) та втрати пакетів мають критичне значення для комфортної роботи користувача. Постійний контроль цих показників дозволяє динамічно перерозподіляти мережеві ресурси та оптимізувати маршрутизацію трафіку в реальному часі, адаптуючись до поточного навантаження на канали зв'язку [4].

Впровадження концепції Індустрії 4.0 та автоматизація виробничих процесів потребують гарантованої надійності мережевих з'єднань для координації роботи промислових роботів та датчиків. У таких системах часові межі реакції вимірюються мілісекундами, тому будь-який метод моніторингу, що працює з запізненням, є неприйнятним. Актуальність теми підтверджується необхідністю створення інтелектуальних систем, які інтегрують алгоритми машинного навчання для прогнозування стану мережі на основі поточних параметрів [5].

Економічна ефективність експлуатації мереж також напряму залежить від якості моніторингу, оскільки проактивне обслуговування обходиться значно дешевше, ніж аварійне відновлення після катастрофічних збоїв. Можливість бачити повну картину розподілу навантаження в реальному часі дозволяє компаніям більш раціонально планувати масштабування інфраструктури та уникати витрат на надлишкові потужності. Це робить розробку нових методів моніторингу пріоритетним завданням для IT-департаментів великих корпорацій та провайдерів телекомунікаційних послуг [6].

Перехід на віддалений формат роботи по всьому світу значно збільшив навантаження на корпоративні VPN-шлюзи та зовнішні вузли доступу. Забезпечення стабільного та безпечного підключення для тисяч співробітників, які перебувають у різних географічних локаціях, вимагає від адміністраторів інструментів для миттєвої діагностики проблем на будь-якій ділянці ланцюга передачі даних. Моніторинг у реальному часі стає "очима" системного адміністратора, дозволяючи бачити стан віддалених сегментів мережі без затримок [7].

Крім того, сучасні стандарти передачі даних та протоколи стають дедалі складнішими, що ускладнює процес ручного пошуку несправностей. Автоматизовані методи збору та візуалізації параметрів мережі в режимі реального часу допомагають знизити навантаження на технічний персонал, зменшуючи ймовірність людської помилки. Це актуально в контексті загального дефіциту висококваліфікованих кадрів, оскільки дозволяє автоматизувати рутинні операції з перевірки цілісності мережі [8].

З появою технологій 5G та розвитком «Розумних міст» обсяги трафіку продовжують зростати в геометричній прогресії, що робить старі підходи до моніторингу остаточно застарілими. Нові методи мають враховувати специфіку високошвидкісних каналів та здатність обробляти великі масиви даних (Big Data) безпосередньо "на льоту". Науковий пошук у цьому напрямку є критично важливим для створення стійкої цифрової екосистеми майбутнього, де мережа є не просто допоміжним інструментом, а життєво важливою артерією суспільства.

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

Отже, актуальність дослідження методів моніторингу параметрів комп'ютерної мережі в реальному часі зумовлена сукупністю технологічних, безпекових та економічних чинників. Необхідність забезпечення безперервності бізнесу, захисту від кібератак, підтримки високої якості сервісів та підготовки інфраструктури до викликів майбутнього робить цю тему однією з найважливіших у сучасній комп'ютерній інженерії.

Розробка ефективного методу дозволить не лише констатувати стан мережі, а й активно керувати її продуктивністю, гарантуючи надійність у будь-яких умовах експлуатації.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Принцип роботи моніторингу параметрів мережі

В цілому мережевий моніторинг із розвитком IT-інфраструктури став постійною складовою керування й адмініструванням інфраструктури в цілому. Загальне поняття моніторингу враховує аспекти періодичного або постійного збирання, аналізу, обробки метрик про стан взаємодії пристроїв, сервісів із ресурсами або пристроями, що розташовані із ними у одній і тій же локальній мережі, або що розташовані віддалено із доступом через випадкові проміжні вузли глобальних мереж. Поняття фіксованого рівня обслуговування(QoS) порівняно не часто використовується для типових користувацьких мереж та мереж дрібного бізнесу, але для мереж із великими кількостями трафіку поняття є досить актуальним.

Стандарт ISO/IEC 27033 переважно орієнтується на аспекти безпеки при моніторингу, проте ефективна система моніторингу має також забезпечувати опрацювання подій в реальному часі, журналювання, збереження метрик та засоби сповіщення адміністратора.

Засоби, спрямовані на мережеву телеметрію, можна поділити на категорії, класифікуючи їх за способом збирання метрик. Активні і пасивні засоби напряму відрізняються способами.

Активні методи зазвичай генерують тестовий трафік, окремі потоки, на основі яких можна виміряти параметри мережевого з'єднання на шляху до точки призначення. Активні засоби створюють синтетичний трафік.

Досить популярними є засоби для ICMP-тестування, використовуючи утиліти ping, traceroute(частина реалізацій утиліти застосовує саме ICMP). Для визначення маршруту застосовуються traceroute і mtr. Щоб виміряти пропускну здатність часто застосовують iperf3. Окремо можна визначити засоби тестування, що починають свою роботу із прикладного рівня, тим не менш в кінцевому результаті тести прикладного рівня все одно відображають більш низькорівневе тестування на 2-3-4 рівнях моделі OSI.

Активний метод аналізу з'єднання має ряд переваг і недоліків. Щодо переваг, то тестування можна провести у довільний час, адаптуючи вхідні параметри так, щоб результати були найбільш показовими. Водночас недоліком є те, що трафік є повністю синтетичним і за певних умов не покаже проблему, яка існує для справжнього трафіку системи. Таке вимірювання характеризує лише поведінку синтетичного трафіку. Окрім цього, генерування окремих потоків – це додатковий трафік, що циркулює мережею, великі об'єми трафіку при тестах iperf3 можуть завантажити канал і напряму негативно впливати на справжній трафік користувача. Також синтетичний трафік може піддаватись блокуванню від мережевих екранів і засобів вторгнення.

Пасивні засоби спрямовані на аналіз існуючого трафіку, не генеруючи нового. Глибокий аналіз заголовків пакетів можуть дати розуміння процесів, що відбуваються в мережі на основі пакетів, що прибули до пристрою. Аналіз концентруєтеся на трафікові застосунків і сервісів. Пасивні засоби не впливають на канал зв'язку та не спотворюють вибірку потоків синтетичними даними. Вплив мережевих екранів на аналізу також не притаманний для цього способу. Трафік, що циркулює від пристрою, до сервісу, який розташований на сервері має бути дозволеним. Якщо сервер не готовий приймати запити, у випадку TCP ми не побачимо встановленої сесії, у випадку UDP/ICMP – не побачимо відповідей.

Окремо можна виділити гібридні засоби. Гібридні засоби моніторингу можуть застосовувати і активні, і пасивні тестування мережі. Таким чином, коли пасивний аналіз зафіксував аномалію – активний аналіз може допомогти локалізувати проблему, не створюючи надмірне навантаження.

Підвищення рівнів складності в рішеннях мережевої інфраструктури відповідно адаптує задачу моніторингу для контролю за мережевими з'єднаннями і якістю передачі даних. Більш традиційні підходи, що використовують як основні метрики – метрики операційної системи, а саме завантаження каналу мережі, завантаження процесора, кількість вільної оперативної пам'яті, проявляють недолік у вигляді статичності. Особливість полягає в тому, що стан каналу і мережі в цілому – для різних потоків даних може бути різний. Це класична особливість підходу комутації пакетів при передачі даних із точки А в точку Б.

Об'єктом дослідження систем моніторингу є процес передачі даних у мережах TCP/IP на рівні окремих мережевих потоків. Мережевий потік - це послідовність пакетів, у випадку TCP, окремі сегменти можна корелювати в єдиний потік за допомогою номера послідовності, IP адреси джерела та призначення та портами джерела і призначення.

Розглянемо завдання систем моніторингу із векторної точки зору. Нехай $F=\{f1,f2,...,fn\}$ - множина TCP-потоків, де кожен потік *fn* визначається чотирма параметрами IP адреси ініціатора, IP призначення, мережевий порт ініціатора та порт призначення *(src_ip, dst_ip, src_port, dst_port)*.

Для кожного *fn* можна визначити вектор характеристик:

$$Q(fn) = \{RTT,\ \sigma RTT,\ L,\ W,\ H\}$$

де RTT - затримка, σRTT - джитер, L - рівень втрат пакетів, W - розмір вікна прийому, H - кількість проміжних вузлів.

ISSN 2219-9365

Принцип обробки трафіку системами моніторингу, що базуються на пасивному аналізі

Оскільки моніторинг цього типу базується на спостереженні за справжнім трафіком, що передається по каналу зв'язку, то система переважно працює як окремий сегмент отримувача.

Для аналізу можуть застосовуватись або повні копії кадрів, пакетів, сегментів і датаграм, або лише їх заголовки із ключовими полями. На відмінну від систем протилежного типу, де аналізуються синтетичні дані, в системі сегменті-отримувача з'являється ускладнення – є потреба аналізувати потік трафіку в реальному часі, не маючи можливості виконати повторний запит. Іншою проблемою, що потребує вирішення, є обмежений контекст. Проаналізувати TCP сесію, без аналізу етапу встановлення і трьох етапної комунікації досить складно.

Система отримує потік фрагментів, що не відносяться один до одного, в довільній послідовності. Якщо аналізувати саме транспортний рівень моделі OSI сегменти, датаграми несуть лише поля заголовку: порядковий номер, розмір вікна прийому, часові мітки, у випадку TCP – стани сесії. Жодне із цих полів не дозволяють одразу зробити висновок про стан з'єднання. Відповідно система, що аналізує справжні пакети є автоматом відстеження стану з'єднань та передбачає механізм поєднання пакетів за ознаками, щоб встановити їх спорідненість. Вигляд формату TCP заголовка подано на рис. 1.

Переважна більшість існуючих методів застосовують повне копіювання пакетів із простору ядра операційної системи в простір користувача із корисним навантаженням. Пакет потрапляє через мережевий інтерфейс в буфер, інструменти типу Wireshark, Zeek досить часто застосовують цей спосіб. Спосіб має чимало готових до використання бібліотек та є досить універсальним в розрізі використанні для систем моніторингу. Проте має значні недоліки, одним із яких є надмірна надлишковість. Для того, щоб оцінити стан мережі, системі моніторингу не потрібно аналізувати поля із інформацією (поле data в заголовку). Іншим недоліком є копіювання пакету в користувацький простір, при рості використання пропускної здатності, система аналізу із невеликою кількість ресурсів не зможе коректно працювати через виснаження ресурсів процесора, часу використання ядер і оперативної пам'яті. Також при надсиланні надмірної кількості трафіку і виснаження ресурсів, операційна система за стандартним механізмом роботи може розпочати відкидання пакетів. Таким чином, результати вимірів можуть бути спотворені.

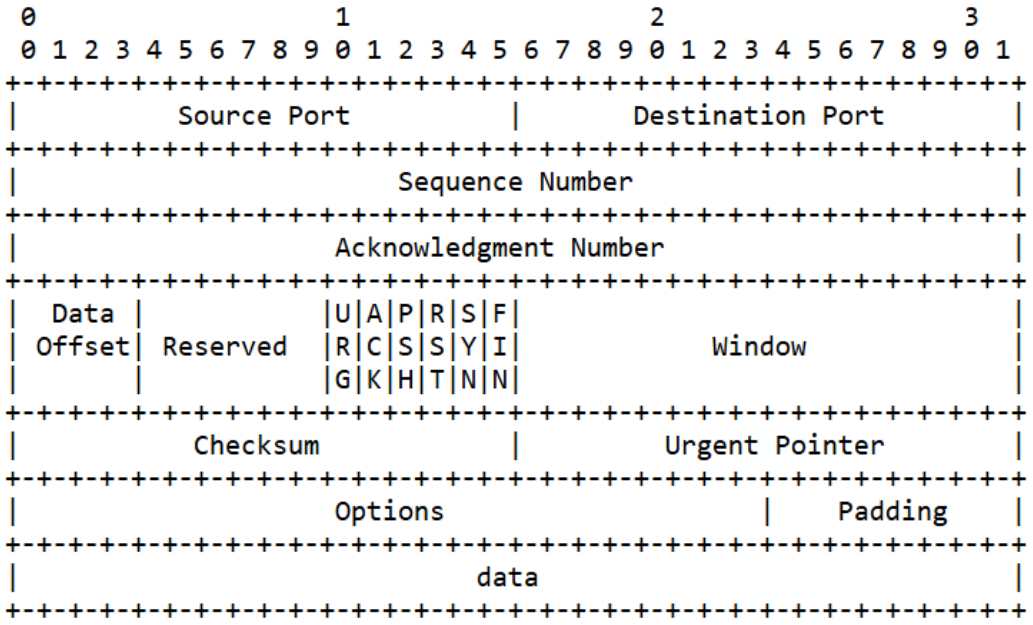

Рис. 1. Вигляд формату TCP заголовка [9]

Можливе також застосування технології TAP (Test Access Point), це фізичний пристрій, що дублює трафік на фізичний інтерфейс, звідки трафік можна зняти для подальшої обробки. Це апаратний пристрій, який здатен захоплювати і передавати кадри із високим рівнем точності. Головні недоліки - фізичне втручання в інфраструктуру, встановлення в кожному сегменті мережі, складність масштабування і створення додаткової точки відмови. Ця технологія чудово підходить для аналізу мережевих загроз та засобів виявлення вторгнень, однак надмірна надлишковість не дасть можливості застосувати технологію із пристроями, що мають мінімальну кількість ресурсів.

Із існуючих способів також існують протоколи типу NetFlow, протоколи агрегованої телеметрії. За допомогою цього методу можна отримати одразу агреговані потоки за класифікаціями, водночас загальна статистика не є надто детальною і її не є достатньо для аналізу параметрів і стану мережі. За допомогою стандартного NetFlow важко визначити аномалії канального і мережевого рівнів.

Порівняно недооціненим методом є захоплення і фільтрація заголовків за допомогою фільтрів eBPF (extended Berkeley Packet Filter). Це код, що виконується напряму у просторі ядра без переведення пакетів в

користувацький простір для аналізу. Аналіз трафіку може відбуватись без виходу за межі простору ядра, а в користувацький простір повертаються лише дані, які були відфільтровані. Розмір пакету зменшується в декілька десятків разів, можливість обробки корисних даних збільшується відповідно. Основні складнощі при створенні реалізацій - це чітко визначений синтаксис, глибокий аналіз логіки, верифікація перед запуском, щоб уникнути неочікуваних умов в коді. Це складність, яка з'являється для реалізацій, але однозначно не недолік, це засіб захисту системи і спосіб забезпечити стабільну роботу операційної системи, недопустивши неочікувані умови до виконання.

Обмеження eBPF полягають в тому, що існує залежність від версії ядра Linux 4.9 (дата офіційного виходу грудень 2016 року), на більш старіших версіях були впровадженні певні функції eBPF, але частина із функцій додавались із оновленнями до версії 4.9.

Таким чином, незалежно від застосованих технологій та специфік тих чи інших реалізацій, основне завдання у випадку відстеження TCP сесій зводиться до відновлення стану TCP-сесій та розбір і пошук спорідненості датаграм UDP із великого потоку даних. Протокол TCP є з'єднано орієнтованим і збір метрик має місце в контексті конкретної сесії. Система може почати працювати із вже встановленими з'єднаннями, без видимості трьох етапного рукостискання(handshake), відповідно такі сесії треба обробляти окремо за іншим механізмом, щоб не створювати хибних метрик.

Глобальні мережі не є ідеальним середовищем із ідеальною точністю передачі даних, відповідно прибування пакетів може відбуватись не в початковій послідовності, а в перевпорядкованому форматі. Це прийнятна поведінка мережі, однак і рішення моніторингу потрібно адаптовувати відповідно.

Також система має спостерігати за сигналами FIN та RST і проактивно очищати записи про завершення сесій.

Попередньо зазначались особливості опрацювання саме TCP потоків трафіку. Однак існує не менш важливий транспортний протокол UDP, що працює без встановлення з'єднання і не має механізму підтвердження доставки датаграми.

Через специфіку роботи протоколу моніторинг має бути побудований із іншим підходом для збирання телеметрії, в цілому моніторинг є більше обмежений в кількості корисних даних, на основі яких можна зробити висновки про роботу мережі.

Кількість UDP датаграм в мережах та їх функції є значними. Велика кількість протоколів використовують саме UDP на транспортному рівні, щоб зменшити затримки при передачі даних за рахунок уникання повторного надсилання даних та відсутності механізму підтвердження. Вигляд заголовку UDP подано на рис 2.

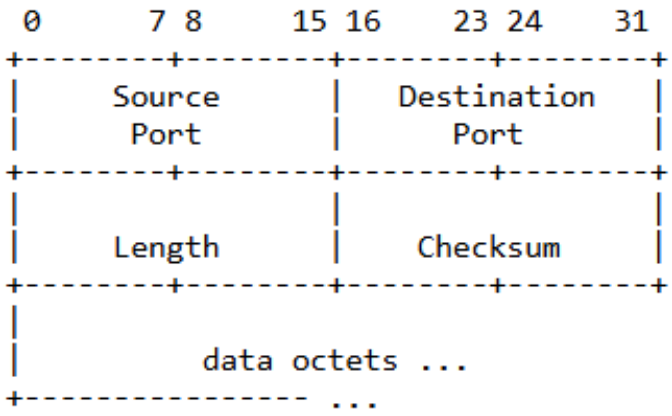

Рис. 2. Вигляд заголовка UDP протоколу [10]

Для UDP датаграм можна також визначити корисну інформацію – це інтенсивність потоку із пропускною здатністю, міжпакетний інтервал. Та однією із найбільш цінних метрик є рівень втрат і зміна послідовність датаграм, що визначена із порядкового номера пакету.

Вимірювання RTT значень для UDP є досить нестандартним завданням, враховуючи відсутність стандартного механізму відповіді на транспортному рівні.

У випадках аналізу трафіку протоколів DNS, NTP, RADIUS на прикладному рівні можна отримати більше корисних деталей, приміром ідентифікатор транзакції.

Використання комбінованих даних із транспортного рівня в парі із даними зібраними на прикладному рівні дає більш детальний результат. Таким чином, можна пов'язати запит-відповідь до транзакції та оцінити стан мережі і проміжних пристроїв через які пройшов UDP трафік до точки призначення.

Значного поширення набув протокол DTLS, призначений для шифрування датаграм. Шифрування датаграми не дозволяє проаналізувати вкладення вище мережевого рівня. Лише доступний заголовок IP рівня,

адреси, порти, розмір пакетів і часові мітки. Набір інформації є досить обмеженим, однак все ще можна визначити пропускну здатність і різницю затримок між пакетами.

У випадку DTLS завдання повністю зводиться до класифікації застосунків і пошуку аномалій без можливості аналізу і розшифрування вмісту датаграми і даних енкапсульованих на вищому рівні. Існує декілька версій DTLS, різниця між DTLS 1.2 і DTLS 1.3 подана на рис. 3.

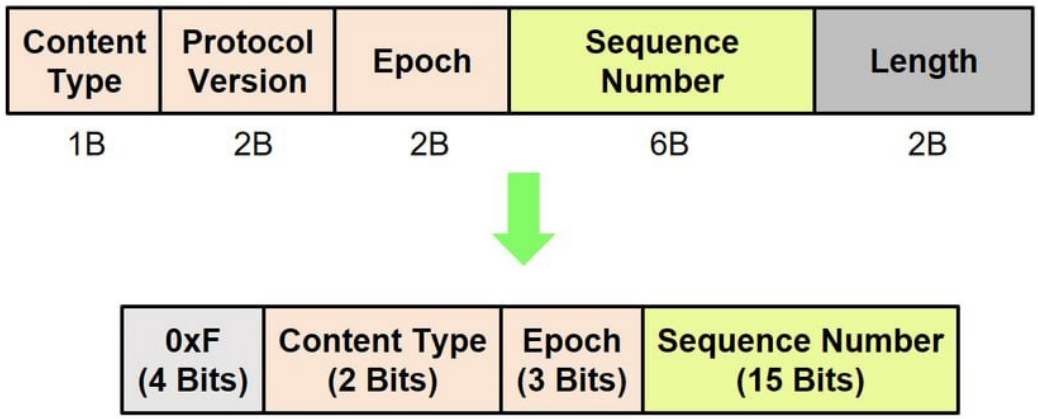

Рис. 3. Вигляд заголовка DTLS протоколу та різниця між версіями DTLS 1.2 і 1.3 [11]

Метод моніторингу параметрів комп'ютерної мережі в реальному часі

Враховуючи результати попереднього аналізу, було вирішено будувати рішення багаторівневої моделі вимірювання характеристик TCP з'єднань на основі аналізу заголовків пакетів у просторі ядра операційної системи засобами eBPF. Основним методом є захоплення і фільтрація заголовків за допомогою фільтрів eBPF. Метод є досить оптимальним для одноплатних комп'ютерів, оскільки керування ресурсами системи здійснюється із порівняно меншими втратами.

Для точності і можливості широкого спостереження за мережею було визначено наступні метрики які необхідно визначати і аналізувати:

1. Ідентифікація потоку.
2. Вимірювання RTT (двома методами для більшої точності).
3. Виявлення повторної передачі сегментів.
4. Оцінка кількості проміжних вузлів при передачі даних від джерела до точки призначення.
5. Визначення напрямку трафіку.
6. Обчислення перцентилів і джитеру у ковзному вікні.
7. Діагностика першопричини аномалій.

В межах етапу проєктування варто формалізувати вимірювання параметрів та подання метрик у математичній формалізації.

Ідентифікація мережевого потоку відбувається за умови, що пакет пройшов через конкретний мережевий інтерфейс та побудувавши ключ на основі набору даних у полях пакету:

$$k = (src\_ip, dst\_ip, sport, dport) \in \mathbb{N}^2 \times \{0..65535\}^2$$

Визначити напрямок пакетів можливо, перевіривши чи належить IP адреса ініціатора трафіку множині IP адрес, налаштованих на локальному інтерфейсі L:

$$dir(k) = \{\, 1\ (OUT), \text{якщо } src\_ip \in L;\ \ 0\ (IN), \text{якщо } src\_ip \notin L \,\}$$

Вимірювання часу RTT можна двома способами. Один із способів - це орієнтування на SYN/SYN-ACK стани TCP сесії, який можна використовувати як резервний, коли мережеве обладнання не підтримує опції TCP Timestamp. При встановлені сесії відправник ініціює встановлення сесії фрагментом SYN, прапорець SYN еквівалентний 1, ACK еквівалентний 0. Фільтр eBPF має зафіксувати проходження фрагменту через інтерфейс і фіксує часову мітку ядра:

$$entry\_timestamp[k] \leftarrow T\_syn = bpf\_ktime\_get\_ns()$$

При надходженні відповіді SYN-ACK сервіс моніторингу має здійснити пошук за оберненим ключем k і здійснює розрахунок RTT як різницю часових міток між фрагментами, один із яких був ініціацією, а інший відповіддю на ініціацію:

$$RTT^1 = T_{synack} - T\_syn,\ \ T\_synack = bpf\_ktime\_get\_ns()$$

Альтернативним засобом виміру є метод, що аналізує TCP Timestamp. Згідно із RFC 7323, якщо обидві сторони підтримують опцію TSOPT, то в TCP заголовку як додаткові поля будуть доступні TSval і TSecr, тобто лічильник відправника і значення, що відповідає за останній отриманий пакет. В реалізації фільтрування і аналізу опцій за допомогою eBPF відбувається ітеративний розбір опцій:

$$RTT_2 = T\_ack - sent\_ns, \text{ якщо } tsopt\_map[k].tsval = TSecr(\text{пакету})$$

Значення RTT може бути коректним за умови:

$$50000 \leq RTT_2 \leq 5000000000 \text{ (нс)}$$

В умові достовірності верхня і нижня межа підібрані експериментально. Нижня межа потрібна, щоб не враховувати хибні спрацювання на фрагменти, що прибули в невпорядкованому форматі. Верхня межа спрямована на те, щоб не враховувати застрілі сесії, які не продовжуються впродовж 5 секунд і, можливо, не є закритими коректно.

Основний спосіб виявлення умов перенадсилання пакетів(ретрансмісій) – це пошук співпадінь порядкових номерів фрагментів, що відносяться до одного і того ж потоку із різницею в часовому проміжку визначеного порогу:

$$is\_ret(p) = 1, \text{ якщо } last\_seq[k].seq = SEQ(p) \wedge \Delta T > T\_ret$$

Значення *T_ret* визначено як 50мс. Це нижня межа часу очікування повторної передачі фрагменту. Це значення дозволяє відрізнити повторну передачу від перевпорядкованих фрагментів.

Оцінку проміжних вузлів можна реалізувати за допомогою дослідження і варіації значень TTL. В залежності від пристрою стандартні початкові значення можуть відрізнятись, приміром Android та Linux мають стандартне значення TTL 64, операційні системи Windows та MacOS має початкове значення 128, також в залежності від реалізацій мережевого стеку операційних систем це значення може бути іншим.

$$H = TTL_0 - TTL(p)$$

Визначається як різниця стандартного значення із тим, яке потрапило у фільтр. Значні стрибки кількості проміжних вузлів на одному і тому ж потокові трафіку можуть свідчити про глобальні зміни маршрутизації.

Існують випадки, коли проміжні пристрої можуть двічі віднімати значення TTL при передачі фрагменту далі, тому різниця у одну-дві одиниці від середнього значення можна вважати як похибку або балансування трафіку на проміжних вузлах.

Щодо оцінювання рівня втрачених фрагментів за інтервал часу, то за одиницю часу спостереження система фіксує кількість повторних передач і загальну кількість пакетів для пристрою, що веде комунікацію. Це співвідношення кількості повторних передач до загальної кількості передач. Рівень втрат можна подати у наступному вигляді:

$$L(\Delta t) = N\_ret \,/\, (N\_total + N\_ret)$$

В контексті втрат також можна визначити рівень порогу, після перетину якого система почне сигналізувати адміністратора.

Розглянувши попередньо опції, які можуть бути використанні для програмної реалізації, оцінивши ризики, було прийнято рішення застосовувати комбіноване рішення із мов програмування C та Python.

Мова програмування C є необхідною для взаємодії саме із ядром і фільтром eBPF, це забезпечує низькорівневу обробку трафіку, без переміщень пакетів у користувацький простір. Швидкодія мови програмування C є перевагою для загальної швидкодії системи.

Мова програмування Python в контексті побудови програмної реалізації вирішує інше завдання. Створення програмних інтерфейсів для міжсервісної взаємодії, обробки і аналізу тих результатів, які були отримані від eBPF, основна функція, на яку планується використати мову програмування Python. Акцент на швидкодії робиться саме у тому місці, де можливе надвисоке навантаження через сплески трафіку.

Технологія eBPF є не зовсім ідентичною до класичних підходів програмування у користувацькому просторі. Так як виконання відбувається у просторі ядра, відповідно синтаксис і доступні конструкції є обмеженими. Обмеження пов’язанні із роботою верифікатора ядра перед безпосереднім завантаженням коду.

Статичний верифікатор ядра перевіряє кожну процедуру і конструкцію до виконання. Верифікатор будує граф потоку управління і симулює виконання всіх шляхів, аналізуючи зміни регістрів і поведінку програми. Будь - яке відхилення від дозволених конструкцій спричинить виведення діагностичного повідомлення і виконання буде неможливим до ре-конфігурації синтаксису до норми. Навіть за умови успішної компіляції код може бути недоступним до фактичного виконання.

Таким чином, при розробці алгоритмів потрібно окремо від вимог компілятора акцентувати увагу і на вимоги верифікатора. Суттєвим і помітним обмеженням є заборона на використання циклів без статичної термінації і чітко визначеної кількості ітерацій виконання. Приміром цикл із break не може бути верифікованим, навіть якщо ми гарантовано впевнені в існуванні умови, при якій цикл закінчиться.

При створенні eBPF програм потрібно передбачити механізм передачі станів. BPF мапа - це структура, що існує в просторі ядра, але водночас доступна із простору користувача. Для хеш-таблиць BPF_HASH доступні декілька операцій - пошук, оновлення і видалення. Особливістю у порівнянні із хеш-таблицями є те, що BPF_HASH таблиці мають статичний розмір, під який виділяється пам’ять у момент запуску. Перелічену специфіку не можна позиціонувати як недолік, це правила, яких варто дотримуватись і враховувати при написанні коду.

У методі, що розробляється ми намагатимемось не переносити надлишкові дані між просторами ядра і користувача, щоб зберегти ресурси системи, однак повного перенесення уникнути не вдасться. Технічно передача даних між просторами можлива завдяки окремим вказівникам запису на рівні ядра і читання на рівні користувача. Тобто, якщо ми хочемо перенести значимі поля пакету розміру 40-80 байт замість повних 1500 байт, то на рівні ядра ми записуємо дані в буфер, виконуючи функцію perf_submit, а для зчитування в просторі користувача виконується функція perf_buffer_pool.

Через надмірну кількість трафіку і високу інтенсивність, яку операційна система не може опрацювати через брак ресурсів, відбувається переповнення буферу, яке може здійснити вплив на загальну статистику.

Для точного обчислення інтервалів часу оптимальним варіантом є використання часу ядра у наносекундах замість сторонніх залежностей. Опитати час ядра можна завдяки функції bpf_ktime_get_ns, із точність до 1нс. Блок-схема алгоритму частини рішення, що виконується у просторі ядра, зображено на рис. 4.

Поділ виконання частин методу на два окремих простори також спрямований на підвищення безпеки системи. Взаємодія між просторами відбувається у формі однонаправленого потоку. З міркувань практичності і безпеки поділ реалізований таким чином, що у просторі ядра відбувається доступ і фільтрація потоків трафіку, а у просторі користувача виконуються частини, для яких необхідна динамічна пам’ять. В користувацькому просторі опрацьовуються складні структури даних, доступна підтримка більшої кількості бібліотек мов програмування. Також у користувацькому просторі більш доцільно проводити агрегацію статистик.

Як інструмент взаємодії між мовами програмування Python і C застосована бібліотека BCC, під час виконання частини Python коду, відбувається компіляція C коду. Такий підхід дозволяє вносити зміни в C код і тестувати результати одразу на пристрої, без необхідності використання бінарних файлів. Для застосування бібліотеки необхідно передавати метадані ядра.

Відлагодження eBPF застосунків є значно складнішим порівняно із звичайним C кодом, для тестування і фіксації виведень застосовується трасування ядра за шляхом /sys/kernel/debug/tracing/trace_pipe. Тобто саме для проведення тестувань скриптів немає можливості виведення результатів відладжування в термінал через засоби стандартного виводу, результати можна побачити після виконання, переглянувши вміст файлу.

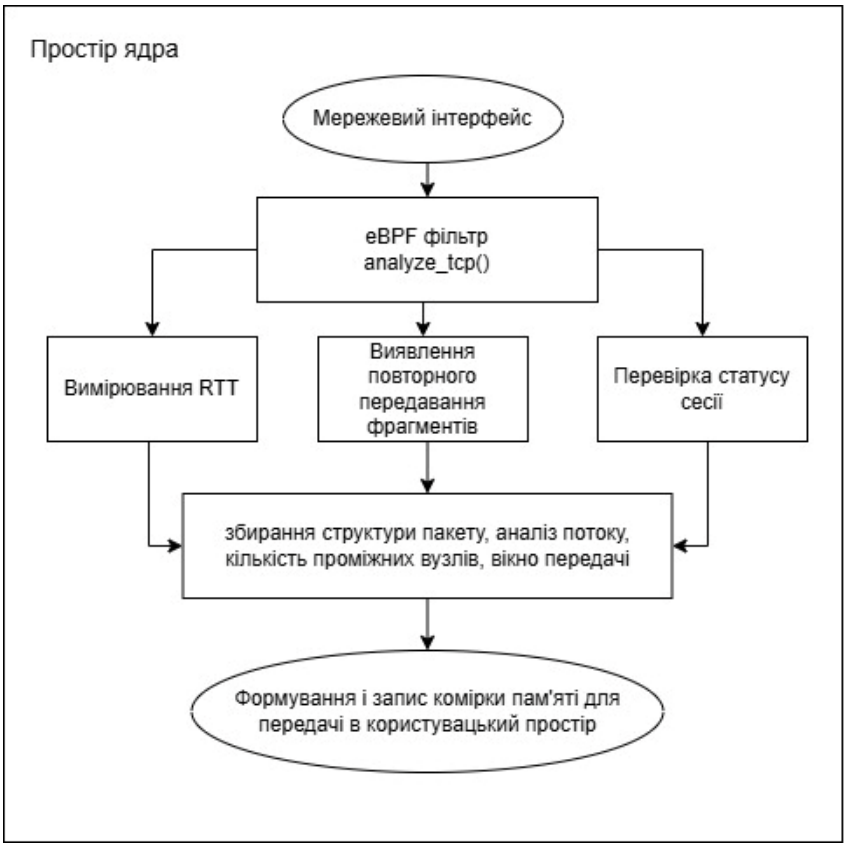

Рис. 4. Блок-схеми алгоритму в частині, що виконується у просторі ядра

Таким чином, побудова методу та системи зведена до первинної фільтрації і агрегації метрик у просторі ядра і доопрацювання в просторі користувача. Даний підхід і вирішує проблему управління ресурсами і забезпечує стабільну роботу на пристроях із невеликою кількістю ресурсів, в тому числі і одноплатних комп’ютерів.

У порівнянні із існуючими системами доступними на ринку, саме цей підхід демонструє приріст у продуктивності та зменшення використання системних ресурсів через вдосконалений засіб обробки потоків трафіку.

Якщо більш детально розглянути іншу частину реалізації методу у просторі користувача, то вона створена на Python3. Під час виконання ця частина запускається як єдиний процес, що взаємодіє із частиною у просторі ядра.

Протягом всього часу виконання, відбувається вичитування записів із буферу ядра через інтервал у 100мс. Функція виконується у однопоточному режимі, механізм псевдо багатопоточності у скрипті застосовується для DNS запитів та сервера метрик Prometheus.

Як було зазначено раніше, на етапі виконання вивчаються локальні IP адреси на основі активних мережевих інтерфейсів, щоб спростити процес ідентифікації пакетів на вхідні і вихідні. Саме у цьому просторі відбувається безпосереднє обчислення і визначення параметрів на основі отриманих даних у необробленому вигляді.

Як додаткова функція, на основі IP адреси отримувача відбувається виконання функцій спрямованих на визначення мережевої автономної системи (ASN). Адміністратор може використати номер автономної системи при пошуку точок, де спостерігаються втрати, та при розслідуванні мережевих аномалій. Дана функція не блокує головний процес від продовження виконання так як DNS запити можуть виконуватись декілька сотень мілісекунд. Блокування головного процесу на час очікування відповіді від DNS сервера може спричинити затримки через зупинку перехоплення потоків.

У просторі користувача також відбувається етап діагностики і класифікації аномалій. Пошук аномалій відбувається на основі чотирьох правил, які перевіряють всі потоки на основі зібраних діагностичних деталей. Правила перевіряються незалежно, на основі умов відбувається формування результату. Це спрямовано на те, щоб адміністратору вказати на можливі аномалії. Передбачено чотири сценарії, окрім штатної роботи мережі: втрати пакетів під час передачі із підвищеними затримками, втрати пакетів без підвищених затримок, нестабільність шляху або перевантаження буферу, також можлива ситуація, коли точка призначення знаходиться на великій географічній відстані, відповідно значення RTT може коливатись, але без втрат пакетів.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

У ході дослідження обґрунтовано, що в умовах стрімкої цифровізації та ускладнення мережевих архітектур, традиційні методи періодичного збору даних стають недостатніми. Перехід до безперервного моніторингу в реальному часі є критично необхідним для забезпечення кібербезпеки, підтримки високої якості обслуговування (QoS) та стабільної роботи інфраструктури в концепції Індустрії 4.0.

Проведений порівняльний аналіз активних та пасивних засобів моніторингу виявив, що пасивний аналіз реального трафіку є більш точним, оскільки не створює додаткового навантаження на канали зв'язку та не спотворює вибірку синтетичними даними. Водночас визначено ключові виклики такого підходу, зокрема необхідність обробки великих обсягів інформації безпосередньо «на льоту» та складність відновлення стану TCP-сесій.

Запропонований метод моніторингу базується на використанні технології eBPF, що дозволяє виконувати фільтрацію та агрегацію метрик безпосередньо у просторі ядра операційної системи. Це вирішує проблему надмірної надлишковості та виснаження ресурсів процесора, притаманну класичним інструментам, які копіюють повні пакети у користувацький простір.

Математична формалізація методу охоплює ідентифікацію потоків за набором ключових параметрів та вимірювання RTT двома способами: через аналіз станів SYN/SYN-ACK та за допомогою опції TCP Timestamp. Такий комбінований підхід забезпечує високу точність вимірювань навіть у складних умовах перевпорядкування пакетів або відсутності підтримки певних опцій мережевим обладнанням.

Програмна реалізація методу, що поєднує мови C та Python, демонструє ефективний розподіл задач: низькорівнева обробка трафіку відбувається у просторі ядра, а складна агрегація, DNS-запити та взаємодія з сервером метрик – у просторі користувача. Це дозволяє мінімізувати копіювання даних між просторами, передаючи лише значущі поля пакетів, що критично важливо для пристроїв з обмеженими ресурсами.

Результати роботи підтверджують, що впровадження інтелектуальної діагностики аномалій на основі зібраних метрик дозволяє не лише констатувати стан мережі, а й оперативно виявляти причини збоїв, такі як перевантаження буферів чи нестабільність маршрутів. Запропонований підхід забезпечує приріст продуктивності та надійність керування сучасною мережевою інфраструктурою.

References

1. Method for Estimating the Convergence Parameters of Dynamic Routing Protocols in Computer Networks / H. Osukhivska et al. 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT), LVIV, Ukraine, 22–25 September 2021. 2021. URL: https://doi.org/10.1109/csit52700.2021.9648792.

2. Method of Restoring Parameters of Information Objects in a Unified Information Space Based on Computer Networks / V. Mukhin et al. International Journal of Computer Network and Information Security. 2020. Vol. 12, no. 2. P. 11–21. URL: https://doi.org/10.5815/ijcnis.2020.02.02.

3. A New Static Cost-Effective Parameter for Interconnection Networks of Massively Parallel Computer Systems / M. M. Hafizur Rahman et al. Soft Computing in Data Analytics. Singapore, 2018. P. 147–155. URL: https://doi.org/10.1007/978-981-13-0514-6_15.

4. Pimenta Junior A. P., Abe J. M., Silva G. C. Determination of Operating Parameters and Performance Analysis of Computer Networks with Paraconsistent Annotated Evidential Logic Eτ. IFIP Advances in Information and Communication Technology. Cham, 2016. P. 3–11. URL: https://doi.org/10.1007/978-3-319-51133-7_1.

5. Kovalenko O. Y., Kuzniuk K. V. Computer network monitoring systems. Mathematical machines and systems. 2023. Vol. 1. P. 50–59. URL: https://doi.org/10.34121/1028-9763-2023-1-50-59.

6. Bodenham D. A., Adams N. M. Continuous Monitoring of a Computer Network Using Multivariate Adaptive Estimation. 2013 IEEE 13th International Conference on Data Mining Workshops (ICDMW), TX, USA, 7–10 December 2013. 2013. URL: https://doi.org/10.1109/icdmw.2013.114.

7. Analysis of Network Parameters Influencing Performance of Hybrid Multimedia Networks / D. Kovac et al. International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems. 2013. Vol. 2, no. 3. URL: https://doi.org/10.11601/ijates.v2i3.69.

8. Shete P. J., Awale R. N., Ket S. Y. Channel quality aware cross-layer design based rate adaptive MAC for improving the throughput capacity of multi-hop ad hoc networks. Ad Hoc Networks. 2017. Vol. 63. P. 45–61. URL: https://doi.org/10.1016/j.adhoc.2017.05.009.

9. Kamtam A., Kamar A., Patkar U. C. Artificial Intelligence approaches in Cyber Security. International Journal on Recent and Innovation Trends in Computing and Communication. 2016. Vol. 4. Issue 4. Pp. 5-9.

10. Kaur P., Gill N. Performance Comparison of UDP and UDP-Lite for Different Video Codecs. International Journal of Computer Applications. 2012. Vol. 54, no. 12. P. 15–22. URL: https://doi.org/10.5120/8617-2479.

11. eeDTLS: Energy-Efficient Datagram Transport Layer Security for the Internet of Things / U. Banerjee et al. 2017 IEEE Global Communications Conference (GLOBECOM 2017), Singapore, 4–8 December 2017. 2017. URL: https://doi.org/10.1109/glocom.2017.8255053.