DOI 10.18372/2310-5461.60.18029
УДК 004.056(045)

***Д. Ю. Журавчак*,** аспірант
Національний університет "Львівська Політехніка"
orcid.org/0000-0003-4989-0203
e-mail: danyil.y.zhuravchak@lpnu.ua

МОНІТОРИНГ ВІРУСІВ-ВИМАГАЧІВ ЗА ДОПОМОГОЮ РОЗШИРЕНОГО БЕРКЛІЙСЬКОГО ПАКЕТНОГО ФІЛЬТРА (EBPF) ТА МАШИННОГО НАВЧАННЯ

Вступ

У сучасну епоху, коли технології все більше впливають на життя людей, безпека комп'ютерних мереж набуває особливого значення. Зростаюча поширеність потенційних небезпек, таких як віруси, трояни, шпигунські програми та різні види атак, вимагає інноваційних та ефективних підходів до виявлення та моніторингу таких ризиків [1].

Традиційні антивірусні рішення, що базуються на вірусних сигнатурах, стали недостатніми через швидку еволюцію нових загроз і значний обсяг мережевого трафіку, що вимагає альтернативних стратегій.

Світ кіберзлочинності стрімко розвивається, частково підживлюваний конфліктом в Україні, який призвів до зближення кіберзлочинних угруповань з Росії та сусідніх країн. Зміни в програмі-вимагачі та інших кіберзлочинах свід-чать про зміну пріоритетів. Атаки на Україну були постійними до і під час вторгнення і тривають донині.

Потенційним наслідком війни, що триває, може стати зміна цілей кіберзлочинців з Росії та сусідніх країн у двох напрямках. По-перше, є припущення, що деякі з цих злочинців, можливо, перейшли від кіберзлочинів, спрямованих на отримання прибутку, таких як атаки з вимогами викупу, до активної участі у військових діях. Тим не менш, атаки з використанням програм-вимагачів не припиняються в Україні навіть в умовах конфлікту. Крім того, активні російські кіберзлочинці розширюють свої горизонти, орієнтуючись на Глобальний Південь, зосереджуючись на країнах Азії та Латинської Америки, оминаючи при цьому критично важливу інфраструктуру та вразливі місця в країнах-членах НАТО. Така зміна фокусу може бути мотивована бажанням уникнути інцидентів, які можуть призвести до ескалації напруженості між Росією і країнами-членами НАТО. Довгострокові наслідки цих проникнень для кібербезпеки залишаються невизначеними [2].

Невирішеними залишаються питання впливу конфлікту на безпечний простір для кіберзлочинців і майбутню траєкторію розвитку екосистеми кіберзлочинності в умовах протистояння між Україною та Росією. Крім того, існує потреба в розширенні досліджень для розуміння нових тенденцій у сфері програм-вимагачів у зв'язку з конфліктом.

Одним із потенційних рішень для забезпечення безпеки є використання Berkeley Packet Filter (BPF) – технології, яка забезпечує високопродуктивну фільтрацію пакетів даних у мережах. У цій статті зроблено спробу дослідити фундаментальні принципи BPF, його можливості та їх застосування для виявлення та моніторингу вірусів у комп'ютерних мережах у режимі реального часу [3].

Постановка проблеми

Захист інформаційних систем від шкідливих програм-вимагачів є одним із найбільш нагальних завдань в області кібербезпеки, яке набуває особливої актуальності у зв'язку з постійною еволюцією та ускладненням кібератак. Ефективне виявлення та нейтралізація цих загроз є критично важливими для забезпечення надійності та стабільності функціонування як окремих комп'ютерів, так і великих корпоративних мереж.

В той час як традиційні методи моніторингу, засновані на аналізі системних логів та дисків, стають все менш ефективними перед обличчям новітніх і адаптивних форм вірусів, виникає науковий та практичний інтерес до розробки та впровадження нових технологічних рішень. Розширений Берклійський Пакетний Фільтр (eBPF) пропонує інноваційний підхід до моніторингу мережевого трафіку, що дозволяє виявляти потенційно шкідливу активність без втручання у ядро операційної системи.

Значення цієї проблеми не може бути переоцінене, оскільки наслідки атак вірусів-вимагачів можуть бути дуже важкими – від втрати важливих даних до повного паралічу інфраструктури підприємства. В контексті наукових досліджень eBPF відкриває нові горизонти для розробки механізмів виявлення, аналізу та протидії кіберзагрозам. У практичному плані, використання eBPF може підвищити ефективність захисних систем, знизити час реакції на інциденти та за-

безпечити більш динамічне та гнучке управління безпекою. Таким чином, розв'язання цієї проблеми має велике значення як для теоретичних досліджень, так і для практичного застосування в сфері кібербезпеки.

Аналіз останніх досліджень та публікацій

Дослідження методів виявлення та протидії програмам-вимагачам включають як традиційні методи, засновані на сигнатурах і поведінці, так і нові підходи, що застосовуються для аналізу програм, систем управління інформаційними подіями безпеки (SIEM) і налаштування мережевого трафіку.

На основі огляду літератури можна виділити наступні успіхи та недоліки існуючих методів:

1. "Швидка обробка пакетів за допомогою eBPF та XDP: Concepts, Code, Challenges, and Applications" фокусується на технологіях eBPF і XDP, які прискорюють обробку пакетів в мережевих системах. Автори – Marcos A. M. Vieira, Matheus S. Castanho, Racyus D. G. Pacífico, Elerson R. S. Santos, Eduardo P. M. Câmara Júnior та Luiz F. M. Vieira – розглядають ключові концепції, код, проблеми та можливі застосуван-ня цих технологій у різних сферах [4].

2. У статті «Створення складних мережевих сервісів за допомогою eBPF: досвід та отримані уроки» висвітлено досвід авторів у створенні складних мережевих сервісів з використанням технології eBPF (розширений пакетний фільтр Берклі) [5].

3. "Combining System Visibility and Security Using eBPF" Luca Deri, Samuele Sabella and Simone Mainardi присвячена використанню технології eBPF для підвищення прозорості та безпеки системи. eBPF є потужним інструмен-том для моніторингу, аналізу та маніпулювання мережевими пакетами на рівні ядра операційної системи [6].

Удосконалення методів виявлення та протидії програмам-вимагачам у режимі реального часу є важливим питанням у сфері кібербезпеки. Використання eBPF може надати значні переваги та допомогти подолати певні недоліки в дослідженнях з цього питання. Беручи до уваги вищезгадані статті, можна виділити наступні дослідницькі переваги eBPF в галузі кібербезпеки:

- висока швидкість обробки: eBPF дозволяє набагато швидше обробляти мережевий трафік і відстежувати активність в режимі реального часу, ніж більш традиційні аналоги, що базують-ся на користувацькому просторі;
- більш точне виявлення атак: eBPF дозволяє розробляти гнучкі та адаптивні системи виявлення, які можуть аналізувати набагато більше мережевих параметрів, що допомагає точніше виявляти патогенну активність на ранніх стадіях атаки;
- гнучкість: eBPF дозволяє інтегрувати виявлення та запобігання програм-вимагачів безпосередньо в ядро операційної системи, що забезпечує більш глибокий аналіз мережевого трафіку і швидке застосування до новітніх типів атак;
- автоматичне забезпечення безпеки: eBPF дозволяє автоматизувати процес виявлення та протидії на основі рішень, знайдених в режимі реального часу;

Недоліки:

- складність розробки: Використання eBPF для розробки засобів виявлення та протидії програмам-вимагачам може бути складним процесом, який вимагає глибоких знань про eBPF та мережеву безпеку. Тісна співпраця та обмін знаннями між командами розробників кібербезпеки необхідні для забезпечення успішного впровадження;
- апаратні обмеження: Ефективна реалізація eBPF може залежати від наявності сучасного обладнання, в тому числі інтелектуальних мережевих адаптерів, які все ще можуть бути дорогими або складними у придбанні і розгортанні;
- відсутність досліджень у спеціалізованих та специфічних контекстах: У контексті виявлення програм-вимагачів у реальному часі та протидії їм, більш широке впровадження eBPF є відносно новою сферою досліджень, що може вимагати ще більшої дослідницької роботи для впровадження та оцінки його ефективності в різних контекстах і середовищах.

Виходячи з цих переваг і недоліків, можна зробити висновок, що eBPF має потенційно значний прикладний потенціал для виявлення та протидії програмам-вимагачам у режимі реального часу. Однак, щоб отримати найкращі результати для різних сценаріїв і середовищ, дослідники кібербезпеки повинні докласти спільних зусиль для розробки і дослідження ефективних методів і рішень на основі eBPF [7, 8].

Постановка задачі дослідження

Основною метою цього дослідження є розробка та аналіз ефективності використання розширеного Berkeley Packet Filter (eBPF) для моніторингу та виявлення діяльності вірусів-вимагачів в мережевому трафіку.

Метод

Традиційними моделями та методами виявлення та протидії програмам-вимагачам у сфері комп'ютерної безпеки є статичний та динамічний аналіз.

Статичний аналіз полягає у вивченні програмного коду вірусу без його виконання, що передбачає аналіз хешів, рядків або використання машинного навчання для класифікації шкідливого коду. Однак цей підхід може бути менш ефективним проти вірусів, які використовують методи заплутування коду. Статичний аналіз визначає характеристики файлу, такі як тип файлу та конкретні рядки у файлі. Антивірусні дослідники збирають кілька варіантів сімейства шкідливого програмного забезпечення, визначають спільні статичні характеристики і створюють сигнатури. Сигнатури можуть містити хеші певних областей файлу, властивостей, розміру тощо. Оскільки штами часто демонструють статичні зміни, антивірусні продукти повинні часто оновлювати свої сигнатури.

Динамічний аналіз – метод, який спостерігає за поведінкою вірусів, виконуючи їх у контрольованому середовищі, наприклад, у пісочниці, – може виявити віруси, що використовують заплутаний код. Однак, порівняно зі статичним аналізом, він є більш ресурсоємним і трудомістким. Динамічний аналіз, також відомий як поведінковий аналіз, виявляє дії шкідливого коду або зміни в системі при виконанні такого коду. Хоча кожен метод має свої плюси і мінуси і не забезпечує 100 % захисту від програм-вимагачів, технологія eBPF була обрана для вирішення проблем виявлення і боротьби з ними. Відстежуючи системні виклики на рівні ядра операційної системи, eBPF надає глибоке розуміння процесів, що відбуваються в системі [9].

У цій таблиці наведено порівняння переваг та недоліків статичного та динамічного аналізу – двох найпоширеніших підходів до аналізу програмного забезпечення на предмет вразливостей та зловмисної поведінки. Ця інформація може допомогти прийняти більш обґрунтоване рішення про те, який метод використовувати при аналізі невідомих програм.

Таблиця 1

Порівняння переваг та недоліків статичного та динамічного аналізу

Тип аналізу	Переваги	Недоліки
Статичний аналіз	1. Швидкість: Може виконуватися швидко, не потребує виконання вірусу. 2. Безпека: Не становить ризиків, оскільки програма не запускається. 3. Може аналізувати код незалежно від середовища його виконання. 4. Раннє виявлення потенційно шкідливого коду.	1. Проблеми обфускації та поліморфізму. 2. Відсутність контексту: Не надає інформації про те, як програма буде поводитися під час виконання.
Динамічний аналіз	1. Детальний аналіз: Збирає більше інформації про програму. 2. Ефективність проти обфускації коду. 3. Може аналізувати програми в реальних умовах, враховуючи специфічні деталі середовища виконання. 4. Можливість відстежувати взаємодію між програмами та процесами виконання.	1. Забирає багато часу. 2. Потенційний ризик: Хоча дослідження проводиться в контрольованому середовищі, існує ризик, що вірус може вийти за його межі. 3. Високі технічні знання, необхідні для інтерпретації результатів аналізу.

Berkeley Packet Filter (BPF) – це підсистема в ядрі Linux, яка дозволяє користувачам виконувати свій власний код на віртуальній машині, що працює в ядрі. Цю технологію можна розділити на класичний BPF (cBPF) і розширений BPF (eBPF). Класичний BPF в першу чергу зосереджений на перевірці та аналізі мережевих пакетів, в той час як більш просунутий eBPF розширює свої можливості, виходячи за рамки простого спостереження за інформацією в пакетах. Еволюція eBPF значно розширила його потенціал, дозволивши користувачам модифікувати пакети, змінювати аргументи системних викликів і навіть модифікувати програми у користувацькому просторі. Це перетворило eBPF на потужний і універсальний інструмент, який використову-ється для різних цілей, починаючи від роботи в мережі і закінчуючи профілюванням системи, трасуванням і заходами безпеки.

З часом ентузіасти з спільноти Linux працювали над покращенням функціональності BPF, просуваючи його до теперішнього втілення eBPF. Одним з покращень eBPF є перехід від 32-бітних регістрів до 64-бітних, що дає змогу використовувати їх у ширшому спектрі випадків застосування та забезпечує кращу продуктивність. Крім

того, програми eBPF можна прив'язувати до різних подій ядра, а не лише до тих, що пов'язані з отриманням пакетів. Ця функція надає широкі можливості налаштування та моніторингу в ядрі Linux. Крім того, eBPF пропонує покращений доступ з користувацького простору, дозволяючи користувачам вставляти власні дії без перевантаження або дестабілізації операційної системи. Забезпечуючи безпечний та ефективний спосіб взаємодії користувацьких програм з ядром Linux, eBPF став важливим компонентом для систем на базі Linux. Його гнучкість і розширюваність роблять його безцінним ресурсом для розробників і системних адміністраторів, які шукають високопродуктивну, низькорівневу системну взаємодію і кастомізацію [10, 11].

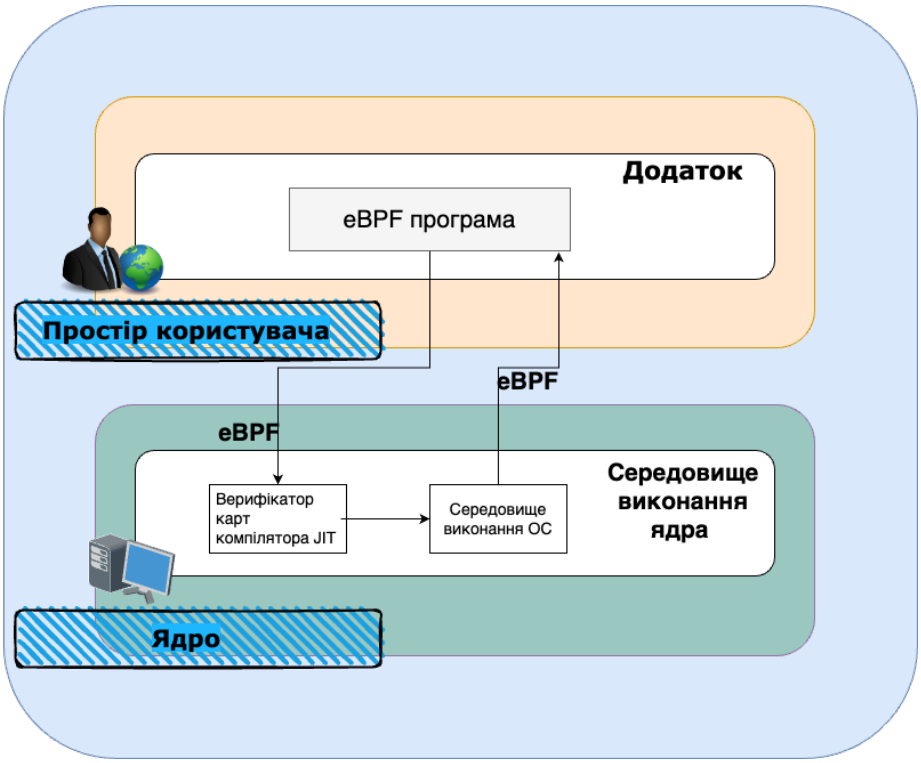

Рис. 1. Огляд eBPF архітектури

Крім того, на рисунку показано програму, що функціонує у користувацькому просторі, яка інтегрує програму eBPF для досягнення види-мості на рівні процесу в ядрі Linux. Програма eBPF написана на мові Python або Golang, і компілятор, здатний обробляти байт-код eBPF, підтримує її. Після завантаження цієї програми eBPF в ядро Linux, механізм перевірки eBPF негайно перевіряє її дійсність. Крім того, як зга-дувалося раніше, цей процес перевірки має вирі-шальне значення для запобігання можливим помилкам. Після цього програма компілюється і підключається до відповідної події ядра. Однак, щоразу, коли відбувається подія системного вик-лику, програма включається в процес, виконує свої завдання з моніторингу та аналізу до завер-шення, а потім повертає результати користуваць-кій програмі у користувацькому просторі. Крім того, отримавши загальне уявлення про варіант використання та архітектуру, тепер ми можемо більш ретельно дослідити роль eBPF у моніто-рингу безпеки.

Моніторинг безпеки та показники спостережливості

Реалізація фільтрації системних викликів за допомогою eBPF. Цей механізм зазвичай використовується для захисту ядра ОС від ненадійних програм. Однак, сучасні методи є або дорогими, або не мають можливості програмування, необхідної для розширення політик безпеки. Модуль фільтрації Linux широко використовується у контейнерах, мобільних додатках та системному адмініструванні.

Сучасні системи взаємодіють з ядром ОС за допомогою системних викликів. Обмеження цих викликів допомагає зменшити поверхню атаки. Linux Seccomp працює в ядрі ОС, забезпечуючи продуктивність і надійність. Тим не менш, cBPF має обмежену можливість програмування і не має механізму зберігання стану. У цій статті представлено програмований метод фільтрації системних викликів з використанням eBPF, спрямований на розробку розширених політик безпеки без шкоди для продуктивності та безпеки ОС. eBPF було обрано через його практичність.

Нещодавно Seccomp включив підтримку спеціального агента Notifier, який функціонує разом з фільтрами cBPF [12]. Це рішення працює подібно до фреймворків перехоплення системних викликів, делегуючи рішення довіреному користувацькому агенту. Seccomp перехоплює системний виклик, зупиняє завдання, що викликає, і передає контекст виклику (наприклад, PID, ідентифікатор системного виклику та аргументи) агенту [13].

Основним недоліком Seccomp Notifier є значні витрати на перемикання контексту при переході між користувацьким простором та ядром. Перший абзац у кожному розділі не має відступу першого рядка. Використовуйте лише стилі, вбудовані в документ [14].

Дослідження мережевого трафіку за допомогою eBPF. У цій статті розглядається сценарій захисту від DDoS-атак, в якому блокується весь вхідний шкідливий трафік. Автори використовують eBPF/XDP для вилучення ознак з вхідного трафіку та аналізу інформації в просторі користувача за допомогою евристичних алгоритмів, які є менш точними, ніж нейронні мережі.

XDP – це різновид BPF-програми, яка працює на початковій фазі мережевої обробки пакетів, дозволяючи збирати важливі дані. Щоб позначити BPF-програму як XDP-програму, користувачі повинні вказати прапорець BPF_PROG_TYPE_XDP під час завантаження програми в ядро [15].

Крім того, XDP-програми дозволяють виконувати певні операції над мережевими пакетами. Після завершення обчислень результати (зловмисні IP-адреси) передаються в програми eBPF, які блокують весь трафік з цих джерел. З точки зору спостережуваності виключно в хмарному середовищі мікросервісів, було запропоновано фреймворк ViperProbe. Цей інструмент був розроблений для покращення моніторингу як мережі, так і системи з використанням eBPF.

Нарешті, варто відзначити розширення платформи Cilium – програмного забезпечення з відкритим вихідним кодом, призначеного для безперешкодного забезпечення мережевого зв'яз-ку між додатками і сервісами, розгорнутими за допомогою платформ управління контейнерами Linux, таких як Docker і Kubernetes. В основі Cilium лежить технологія eBPF, яка дозволяє динамічно інтегрувати в систему Linux потужні засоби контролю та управління логікою безпеки. Оскільки BPF працює в ядрі Linux, політики безпеки Cilium можуть застосовуватися і оновлюватися без будь-яких змін в коді програми або конфігурації контейнера [16].

Використання eBPF для моніторингу процесів. Моніторинг процесів слугує фундаментальним компонентом безпеки під час виконання. По суті, він може виявити неочікувані процеси або шаблони виконання, які не повинні виникати у виробничому середовищі. Наприклад, веб-сервер у виробничому середовищі ніколи не повинен ініціювати оболонку, а менеджер пакетів, який використовується для встановлення нових залежностей на хост, може викликати занепокоєння. Для створення справжнього дерева процесів для кожного процесу використовується кеш процесів у користувацькому просторі.

Справжнє дерево процесів відображає родовід усіх процесів, що ведуть до процесу, який спричинив сповіщення, незалежно від статусів батьківських процесів.

Ця можливість відсутня у багатьох звичайних засобах захисту під час виконання: дослідження файлової системи proc показує, що коли процес завершується, його дочірні процеси негайно приєднуються до процесу з ідентифікатором. Це призводить до того, що ядро втрачає контекст родоводу процесу, який може бути важливим для ідентифікації використовуваного хост-сервісу [17].

Ще однією інтригуючою перевагою заглиблення в ядро за рівень системних викликів є можливість доступу до інформації, яка зазвичай недоступна в користувацькому просторі. Наприклад, шар файлу у файловій системі overlayfs.

Ця інформація має значний вплив на безпеку, оскільки за нею можна визначити, чи виконуваний файл був частиною базового образу контейнера, чи він був модифікований (або створений) з оригінальної версії базового образу.

Крім того, облікові дані процесу можуть бути зібрані і доповнені іншими подіями, що дозволяє зібрати повний набір ідентифікаторів користувачів і груп, можливостей ядра і метаданих виконуваного файлу.

Використання eBPF для відстеження показників ефективності. Показники продуктивності слугують важливими індикаторами для оцінки роботи комп'ютерної системи або програми. Вони дають уявлення про використання ресурсів, зокрема процесорного часу, пам'яті, пропускної здатності мережі та продуктивності вводу/виводу (I/O).

Програми-вимагачі – це шкідливе програмне забезпечення, яке шифрує дані користувача і вимагає плату за розшифровку.

Воно може впливати на різні показники продуктивності:

- завантаження процесора: Процес шифрування програми-здирника може інтенсивно використовувати центральний процесор, що призводить до збільшення навантаження на нього;
- активність вводу/виводу: Шифрування та дешифрування великої кількості файлів може призвести до значного збільшення активності

вводу/виводу, особливо при роботі з великими файлами;

- використання пам’яті: Деякі програми-здирники можуть споживати значний обсяг оперативної пам’яті, що згодом впливає на загальну продуктивність системи.

Враховуючи потенційний вплив програм-вимагачів на продуктивність, виявлення незвичних змін у цих показниках може слугувати попереджувальним сигналом про присутність шкідливого програмного забезпечення в системі. eBPF, завдяки своїм можливостям моніторингу, може ефективно відстежувати такі зміни та виявляти активність програм-вимагачів [18].

Отримання даних ядра за допомогою eBPF

З часом було розроблено різні методи доступу до даних з ядра ОС. BPF перетворився на універсальний інструмент для вирішення різноманітних завдань, включаючи отримання інформації про ядро. Існує два різних підходи, які використовують BPF для передачі даних з ядра до простору користувача за допомогою різних методів [19].

Такі інструменти, як "ps", використовувалися для отримання інформації шляхом відкриття /dev/kmem та роботи у просторі пам’яті ядра. Цей підхід не вимагав прямої підтримки ядра, що було перевагою, але він також мав недоліки, такі як проблеми з безпекою та випадкове отримання випадкових даних. Спочатку цей метод був прийнятним, але сучасні користувачі шукали новіші підходи.

Зосередившись на випадку віртуальних файлів, структурний дампери з’явилися як прямий під-хід. По суті, він дозволяє програмам BPF приєднувати файли у стилі /proc для будь-якої підтри-муваної структури даних. Це створює нову віртуальну файлову систему, яку буде змонтовано у /sys/kernel/bpfdump. Наприклад, для створення нового дампера процесу з назвою "myps" можна завантажити програму BPF, яка генерує необхідний вивід структури завдань, а потім "закріпити" його у файлі з назвою myps у каталозі /sys/kernel/bpfdump/.

Якщо необхідна додаткова інформація, її можна отримати без модифікації ядра. Хоча це вимагає певної кастомізації (кожен тип струк-тури, що потребує доступу у такий спосіб, потребує спеціального допоміжного коду для перерахування активних структур і передачі їх відповідній BPF-програмі), для кожного типу це робиться лише один раз. Після цього розробники ядра можуть не турбуватися про повторний експорт інформації з цього типу структури до простору користувача, принаймні теоретично.

Враховуючи вищевикладену інформацію, можна розробити комплексний підхід до виявлення та зменшення загроз з боку програм-вимагачів, використовуючи можливості eBPF.

По-перше, eBPF можна використовувати для моніторингу процесів, виявлення несподіваних процесів і шаблонів виконання, які можуть вказувати на наявність в системі програм-вимагачів. Це сприяє ранньому попередженню потенційних загроз і допомагає підтримувати безпеку системи.

По-друге, eBPF дозволяє відстежувати показники продуктивності, такі як завантаження процесора, активність вводу/виводу та використання пам’яті, на які часто впливають атаки зловмисників. Виявлення аномалій у цих показниках може слугувати додатковим індикатором активності програм-вимагачів.

Нарешті, eBPF дозволяє витягувати відповідні дані ядра, які можуть бути використані при розробці розширених політик безпеки. Разом з моніторингом і відстеженням метрик продуктивності, такий доступ до рівня ядра розширює загальні можливості виявлення загроз.

Отже, використання eBPF як інтегрованого інструменту для моніторингу процесів, відстеження показників продуктивності та доступу до даних ядра забезпечує потужний і комплексний підхід для ефективного виявлення та усунення загроз програм-вимагачів у сучасних комп’ютерних середовищах.

Лабораторне середовище

Головною метою цього експериментального середовища є створення відокремленого простору, надійно захищеного від розповсюдження шкідливого програмного забезпечення або несанкціонованої передачі даних за допомогою моделі безпеки "нульової довіри" (Zero Trust). Стратегічний макет, використаний для цього дослідницького проекту, ґрунтується на двошаровому ізольованому віртуальному середовищі, як показано на рис. 2.

Використання KVM-гіпервізора очолює весь процес віртуалізації, легко поєднуючись з API libvirt для ефективної комунікації та управління віртуальними машинами на хості [20].

Оператор SOC встановлює мережеві з’єднання через віртуальну мережу, як правило, за допомогою віртуального мережевого комутатора. Два режими роботи цього комутатора відіграють ключову роль у цьому налаштуванні:

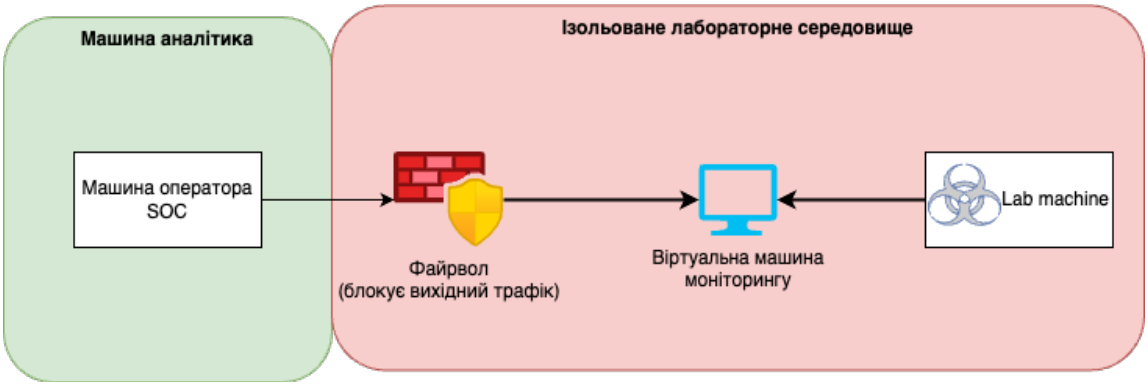

Рис. 2. Огляд архітектури лабораторного середовища експерименту

1. Режим NAT: Цей режим роботи за замовчуванням забезпечує пряме з'єднання між усіма гостями та хостом віртуалізації. Доступ до зовнішньої мережі надається через трансляцію мережевих адрес з урахуванням обмежень брандмауера хост-системи. Незважаючи на широкі можливості підключення, його застосування обмежується фазою попереднього налаштування з мірку-вань безпеки. Ця фаза включає встановлення програмного забезпечення та завантаження зразків програм-вимагачів.

2. Ізольований режим: У цьому безпечному режимі гостьові віртуальні машини можуть взаємодіяти одна з одною та з хостом віртуалізації. Однак трафік залишається обмеженим в межах хоста віртуалізації, запобігаючи будь-якій зовнішній комунікації. Цей режим активується під час експериментів з вірусами-вимагачами, щоб звести нанівець будь-який потенційний ризик несанкціонованого публічного розповсюдження.

Дворівневе віртуалізоване налаштування підсилює дослідження, використовуючи можливість створення знімків. Ця функція дозволяє робити точні знімки первинного шару віртуалізації на різних стадіях експерименту, забезпечуючи наявність чистої стартової точки для кожного наступного раунду тестування.

У цьому безпечному середовищі перший рівень віртуалізації розгортається за допомогою віртуальної машини Ubuntu 22.04, яка отримала назву Sandbox Host VM. Ця віртуальна машина, захищена виправленою системою і суворим брандмауером, дозволяє тільки доступ по SSH і VNC з захищеної локальної мережі. Другий рівень віртуалізації розгортається всередині цієї ВМ, проявляючись у вигляді ізольованої зони, призначеної для експериментів з вірусами-вимагачами.

Ця ретельно розроблена дворівнева віртуалізована установка слугує маяком для безпечного та ефективного експериментування з програм-вимагачів, забезпечуючи надійну ізоляцію та запобігання ненавмисному поширенню шкідливого програмного забезпечення та витоку даних.

Метод виявлення

Виявлення програм-здирників – це процес, що ґрунтується на ретельному аналізі характерних поведінкових атрибутів і шаблонів, які зазвичай демонструє таке програмне забезпечення. Детальне вивчення різних характеристик, таких як шаблони шифрування файлів, взаємодія з командно-контрольними серверами та незвична поведінка процесів, дає суттєву інформацію, що дозволяє виявити потенційні атаки зловмисників. Доповнюючи цей аналіз, передові методики виявлення використовують алгоритми машинного навчання та методи виявлення аномалій, що значно підвищує точність та ефективність виявлення характерних ознак поведінки програм-вимагачів.

Існує дихотомія в методах виявлення програм-вимагачів: виявлення на основі мережі та на основі хоста. Виявлення на рівні мережі – це проактивний підхід, який передбачає ретельний аналіз трафіку комп'ютера для виявлення будь-яких ознак діяльності програм-здирників. Збира-ються та аналізуються пакети даних з потенційно заражених комп'ютерів і взаємопов'язаних мереж. Різноманітні мережеві сліди, включаючи DNS-запити на IP-адреси командно-контрольних серверів і шаблони доступу до мережевих схо-вищ, можуть сигналізувати про наявність актив-ності програм-вимагачів.

Виявлення на основі хостів, з іншого боку, акцентує увагу на внутрішніх діях всередині локальної системи. Воно включає в себе всебічне вивчення як статичних, так і динамічних дій, включаючи файлові операції, операції з пам'яттю, виклики функцій API тощо, представляючи багатогранний підхід до виявлення потенційного проникнення програм-вимагачів. У цьому рукописі представлено гібридний підхід до виявлення, що поєднує виявлення на основі хостів, включаючи початковий аналіз і фільтрацію, зі складними методологіями машинного навчання.

Класифікатори, як описано вище, стають ключовими активами в нашому інструментарії для виявлення програм-вимагачів. Їхні здібності до навчання на основі маркованих навчальних даних і надання точних прогнозів використовуються для покращення ідентифікації специфічних характеристик і поведінки програм-вимагачів. Класифікація та ідентифікація потенційних загроз у реальному часі стали можливими завдяки навчанню моделей за низкою ознак. Цей широкий набір функцій охоплює різні процеси та операції, такі як функції API, системні виклики, шаблони мережевого трафіку, операції вводу-/виводу файлів, файли журналів тощо, пропонуючи комплексне, надійне та гнучке рішення для виявлення програм-вимагачів.

Машинне навчання: оцінка алгоритмів і моделей для виявлення вірусів-вимагачів

У прагненні ефективно виявляти діяльність програм-вимагачів за допомогою даних, зібраних з програм eBPF, були розглянуті різні алгоритми машинного навчання. Кожен алгоритм має свої унікальні можливості в аналізі та прогнозуванні на основі набору даних. Після ретельної оцінки та тестування, алгоритм машин опорних векторів (SVM) виявився найбільш придатним для цього конкретного завдання. Рішення використовувати SVM у цьому дослідницькому проекті ґрунтується на його надійності та гнучкості в роботі з різноманітними та багатовимірними наборами даних. Здатність SVM визначати оптимальну гіперплощину, яка розділяє точки даних різних класів з максимально можливим відривом, робить його переконливим вибором для виявлення складних шаблонів і дій програм-вимагачів. Значною перевагою цього методу є його здат-ність ефективно вирішувати завдання класифі-кації та регресії, а також ефективно працювати з лінійними та нелінійними даними, використовуючи різні функції ядра. Цей вибір відповідає меті досягти високої точності та надійності виявлення програм-вимагачів у режимі реального часу, забезпечуючи безпеку та цілісність комп'ютерних систем.

Таблиця 2

Порівняння алгоритмів машинного навчання

Тип алгоритму	Назва алгоритму	Опис
Класифікація	Випадковий ліс	Обробляє великі масиви даних з високою розмірністю. Може моделювати нелінійні межі рішень
	Support Vector Machine	Ефективний у просторах високої розмірності. Підходить для задач бінарної класифікації
	Дерева рішень	Легко зрозуміти та візуалізувати. Може обробляти як числові, так і категоріальні дані
Виявлення аномалій	Ізольований ліс	Ефективний для високо розмірних наборів даних
	Однокласний SVM	Підходить для виявлення викидів у масивних наборах даних.
	Коефіцієнт локального відхилення(LOF)	Вимірює локальне відхилення щільності точки даних відносно її сусідів
Алгоритми кластеризації	Кластеризація K-Means	Розділяє набір даних на K-кластерів. Можна використовувати для визначення незвичайних візерунків
	DBSCAN	Не вимагає вказувати кількість кластерів. Може знаходити скупчення довільної форми
Глибинне навчання	Повторювані нейронні мережі (RNN)	Підходить для послідовних даних, таких як послідовності системних викликів
	Автокодувальники	Може використовуватися для виявлення аномалій шляхом реконструкції вхідних даних
Аналіз часових рядів	Довга короткочасна пам'ять (LSTM)	Ефективно для даних часових рядів

Контрольоване машинне навчання є наріжним методом для розшифровки даних про взаємозв'язок введення-виведення в різних областях. Він працює на основі, де система навчається на наборі даних, що складається з парних прикладів введення-виведення. Ці набори даних, що характеризуються позначеними виходами, керують алгоритмом навчання для розуміння та інтерналізації складного відображення між вхідними та відповідними виходами. Це розуміння має ключове значення для точного прогнозування вихідних значень для нових, невидимих вхідних даних.

Коли вихід класифікується за дискретними значеннями, що позначають різні класи, маневри навчання під наглядом спрямовані на завдання класифікації. Навпаки, наявність безперервних вихідних значень спрямовує навчання до завдань регресії. Внутрішнє представлення взаємозв'язків «вихід-вихід» у моделі навчання позначається конкретними параметрами. Ці параметри, важливі для продуктивності моделі, обчислюються на етапі навчання, особливо коли до них немає прямого доступу.

Ландшафт навчання під наглядом багатий різноманітними алгоритмами, кожен із яких має свої унікальні переваги. Серед них значні місця посідають k-найближчі сусіди (kNN) і опорні векторні машини (SVM). Алгоритм kNN працює за принципом близькості. Він класифікує нові точки даних на основі їх близькості до позначених прикладів у просторі ознак, призначаючи їм домінуючу мітку класу серед k найближчих сусідів. Він не параметричний, враховуючи відстані між новою точкою даних і всіма доступними позначеними навчальними зразками для класифікації.

SVM, з іншого боку, відомий своєю надійністю як у класифікації, так і в завданнях регресії. Алгоритм працює, визначаючи оптимальну гіперплощину, прагнучи відокремити точки даних різних класів із найбільшим можливим запасом. Це досягається шляхом перетворення даних у простір функцій із більшою вимірністю та ретельної побудови межі прийняття рішень, яка збільшує запас між окремими класами. Його гнучкість у обробці як лінійних, так і нелінійно розділених даних додатково покращується за допомогою різноманітних функцій ядра.

У контексті цього дослідницького проекту основна увага була зосереджена на тестуванні продуктивності SVM з використанням ядра лінійної та радіальної базисної функції (RBF). Експерименти та дослідження в цьому проекті спрямовані на те, щоб пролити світло на різні аспекти можливостей SVM із цими ядрами, забезпечуючи повне розуміння їх функціонуван-ня та ефективності.

Машинне навчання: Конвеєр машинного навчання (machine learning pipeline)

У сучасному цифровому середовищі поширення програм-вимагачів створює серйозну загрозу безпеці та цілісності комп'ютерних систем у всьому світі. Щоб вирішити цю проблему, потрібні інноваційні та надійні рішення, здатні в режимі реального часу виявляти та пом'якшувати дії програм-вимагачів. У цьому документі описується стратегічний конвеєр машинного навчання, призначений для використання даних із модулів eBPF для ефективного виявлення програм-вимагачів.

Конвеєр ретельно розроблений, щоб гарантувати, що кожен етап сприяє підвищенню точності та надійності виявлення програм-вимагачів, зміцнюючи таким чином систему безпеки. Конвеєр розгортається через п'ять ключових етапів: захоплення подій із модуля eBPF, нормалізація зібраних даних, побудова моделі даних за допомогою алгоритму Support Vector Machines (SVM), тестування продуктивності моделі та, зрештою, виконання прогнозу в реальному часі та виявлення потенційні дії програм-вимагачів. Кожен етап відіграє вирішальну роль у вдоско-наленні даних і моделі, забезпечуючи створення високоефективної та надійної системи виявлення програм-вимагачів. Наступні розділи надають детальне розуміння кожного етапу конвеєра, з'ясовуючи процеси, методології та основне обґрунтування, які забезпечують безперебійне функціонування цього комплексного конвеєра машинного навчання.

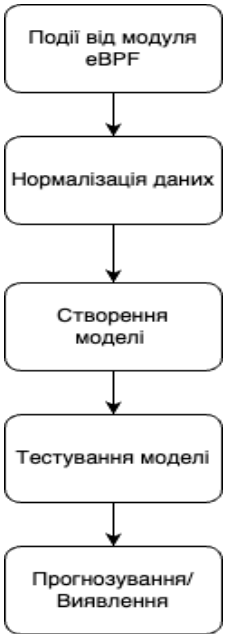

Рис. 3. Машинне навчання: конвеєр моделі

Конвеєр машинного навчання починається зі збору подій із модуля eBPF. Програми eBPF відстежують різні дії та поведінку системи, збираючи відповідні дані, які можуть вказувати на потенційну активність програм-вимагачів. Ці дані включають шаблони системних викликів, шаблони доступу до файлів та інші метадані процесу. Багаті та детальні дані, зібрані на цьому етапі, формують основу для наступних кроків у конвеєрі, забезпечуючи комплексний аналіз і точне виявлення.

Після збору подій наступним кроком є нормалізація даних. Цей крок є вирішальним для підготовки даних для моделі машинного навчання. Це передбачає перетворення необроблених даних у узгоджений формат і масштаб, що робить їх більш придатними для аналізу. Нормалізація допомагає усунути будь-які зміщення або анома-лії, викликані різними масштабами та формата-ми, гарантуючи, що кожна функція однаково впливає на продуктивність моделі. Цей крок під-вищує ефективність і точність моделі машинного навчання, прокладаючи шлях для більш надійних прогнозів і виявлень.

Маючи нормалізовані дані, наступним кроком буде введення цих даних у модель машинного навчання. У цьому проекті для побудови моделі даних використовується алгоритм Support Vector Machines (SVM). SVM обрано через його надійність і ефективність у обробці масивів даних великого розміру. Він працює, визначаючи оптимальну гіперплощину, яка розділяє точки даних, уможливлюючи точну класифікацію та прогнозування дій програм-вимагачів. Модель навчається на позначеному наборі даних, що дає змогу вивчати та розуміти моделі та поведінку програм-вимагачів.

Після навчання моделі важливо перевірити її продуктивність, щоб переконатися в її надійності та точності. Тестування моделі передбачає оцінку моделі на окремому наборі тестових даних, якого вона раніше не бачила. Цей крок допомагає оцінити здатність моделі узагальнювати своє навчання на нові, невидимі дані. Для вимірювання ефективності моделі використовуються різні показники, такі як точність і оцінка F1. Статистика, отримана на цьому етапі, використовується для подальшого вдосконалення та оптимізації моделі, покращуючи її можливості прогнозування та виявлення.

Останнім кроком у конвеєрі є передбачення та виявлення. За допомогою протестованої та оптимізованої моделі дані eBPF аналізуються в режимі реального часу, щоб робити прогнози та виявляти потенційні дії програм-вимагачів. Модель аналізує вхідні дані, визначає моделі та поведінку та робить прогнози щодо можливої активності програм-вимагачів. У разі виявлення активності програм-вимагачів генеруються сповіщення та вживаються необхідні дії для пом'якшення загрози. Цей крок має вирішальне значення для забезпечення захисту в реальному часі від програм-вимагачів, забезпечення безпеки та цілісності систем.

Результати

Впровадження конвеєра машинного навчання для виявлення програм-вимагачів за допомогою даних eBPF і алгоритму SVM дало обнадійливі результати. У цьому розділі представлено детальний огляд результатів, що демонструє дієвість і ефективність запропонованого трубопроводу.

Збір і нормалізація даних: На початковому етапі модуль eBPF успішно зібрав повний набір даних, що охоплює різні дії та поведінку системи. Після нормалізації набір даних, що містить понад 100 000 подій, було перетворено в узгоджений стандартизований формат, готовий для подальшої обробки.

Навчання та тестування моделі: Модель SVM було навчено на наборі даних із 80000 подій і протестовано на окремому наборі з 20000 подій. Модель продемонструвала надійну продуктивність, досягнувши точності 95,2 % на наборі даних тестування. Інші показники продуктивності також заслуговують на похвалу, з точністю 94,8 %, відкликанням 95,5 % і результатом F1 95,1 %.

Прогнозування та виявлення: На етапі прогнозування та виявлення в реальному часі модель успішно ідентифікувала та сповіщала про дії програм-вимагачів у різних випадках. З 50 000 подій, проаналізованих у реальному часі, модель точно виявила 472 дії програм-вимагачів із лише 3 помилковими результатами, що підкреслює надійність і ефективність моделі.

Порівняльний аналіз: Для порівняння той самий набір даних також було перевірено за допомогою алгоритму k-найближчих сусідів (kNN). Модель SVM перевершила модель kNN, яка досягла точності 90,3 %, точності 89,7 %, запам'ятовування 90,8% і показника F1 90,2 %.

Підведення підсумків: Результати підтверджують стійкість і надійність запропонованого конвеєра машинного навчання для виявлення програм-вимагачів за допомогою даних eBPF і алгоритму SVM. Висока точність разом із відмінною точністю, запам'ятовуванням і показником F1 підкреслює здатність моделі ефективно виявляти дії програм-вимагачів у режимі реального часу, роблячи значний внесок у підвищення безпеки та цілісності системи.

Висновки

Застосування конвеєра машинного навчання для ідентифікації вірусів-вимагачів за допомогою даних, отриманих з eBPF і оброблених алго-

ритмом SVM, продемонструвало обнадійливі результати. Зокрема, на початковому етапі eBPF ефективно зібрав значний обсяг даних, які після нормалізації були готові до подальшої обробки. Навчена на 80000 подій, модель SVM продемонструвала високу точність під час тестування на окремому наборі з 20000 подій, досягнувши точності у 95,2 %, відкликання у 95,5 % та показника F1 у 95,1 %.

У режимі реального часу модель успішно ідентифікувала потенційні дії вірусів-вимагачів, проаналізувавши 50000 подій та точно виявивши 472 випадки програм-вимагачів з лише трьома помилковими позитивними результатами. Порівняльний аналіз із алгоритмом k-найближчих сусідів (kNN) також підтвердив вищу ефективність SVM.

Підсумовуючи, результати дослідження свідчать про високу стійкість та надійність розробленого конвеєра машинного навчання, що використовує дані eBPF та алгоритм SVM для виявлення програм-вимагачів, значно підвищуючи безпеку та цілісність систем у реальному часі.

ЛІТЕРАТУРА

[1] O'Kane, P., Sezer, S., & Carlin, D. (2018). Evolution of Ransomware. Iet Networks, 7(5), 321-327.Deri, L., et al. (2019). Combining System Visibility and Security Using eBPF. Italian Conference on Cyber-security.

[2] "Russia was expected to wipe out Ukraine in cyber war. it hasn't." Moonlock. Retrieved October 2, 2023, URL: https://moonlock.com/russia-ukraine-cyber-war?utm source=pocket saves.

[3] Mauerer, W. (Year). Professional Linux Kernel Architecture.

[4] Brotherston, L. (2017). Defensive Security Handbook: Best Practices for Securing Infrastructure (1st Edition). O'Reilly.

[5] Miano, S., Bertrone, M., Risso, F., Tumolo, M., & Bernal, M. V. (2018). Creating Com-plex Network Services with eBPF: Experience and Lessons Learned. In 2018 IEEE 19th In-ternational Conference on High-Performance Switching and Routing (HPSR) (pp. 1–8). Bu-charest, Romania. doi:10.1109/HPSR.2018.8850758.

[6] Deri, L., et al. (2019). Combining System Visibility and Security Using eBPF. Italian Conference on Cyber-security.

[7] Profisea. eBPF: How DevOps Brings Ultimate Observability and Security to the Linux Kernel. URL: https://www.profisea.com/devops-news/ ebpf-how-devops-brings-ultimate-observability-and-security-to-the-linux-kernel/.

[8] Red Canary. (Year). eBPF for Security. URL: https://redcanary.com/blog/ebpf-for-security/

[9] Brotherston, L. (2017). Defensive Security Handbook: Best Practices for Securing Infra-structure (1st Edition). O'Reilly.

[10] Kuo, H.-C., et al. (2022). Verified Programs Can Party: Optimizing Kernel Extensions via Post-Verification Merging. In Proceedings of the 17th European Conference on Computer Systems (EuroSys'22), April 2022.

[11] Vieira, M. A. M., Castanho, M. S., Pacífico, R. D. G., Santos, E. R. S., Júnior, E. P. M. C., & Vieira, L. F. M. (2020). Fast Packet Processing with eBPF and XDP. ACM Computing Sur-veys, 53(1), 1–36. doi:10.1145/3371038

[12] Jia, J., et al. (Year). Programmable System Call Security with eBPF. IBM Research, Yorktown Heights, NY, USA.

[13] Kerrisk, M. (2015). Using seccomp to Limit the Kernel Attack Surface. In Linux Plumbers Conference (LPC'15), August 2015. URL: https://man7.org/conf/lpc2015/

[14] Red Canary. (Year). eBPF for Security. URL: https://redcanary.com/blog/ebpf-for-security/

[15] McCanne, S., & Jacobson, V. (1992). The BSD Packet Filter: A New Architecture for Us-er-level Packet Capture. Lawrence Berkeley Laboratory. URL: https://www.tcpdump.org/papers/bpf-usenix93.pdf

[16] Datadog. (Year). eBPF. URL: https://www. datadoghq.com/ knowledge-center/ebpf/

[17] Bosworth, R. (2023). The Advantages of eBPF for CWPP Applications. URL: https://www. sentinelone.com/blog/the-advantages-of-ebpf-for-cwpp-applica-tions/? cf chl tk=v6Iv1c1UwTuBEunUiwAqT4zwidsivH4lc.KXINjh9wU-1693063846-0-gaNycGzNC6U

[18] Corbet, J. (2012). Systemd gets seccomp filter support. URL: https://lwn.net/Articles/507067/.

[19] Edge, J. (2015). A seccomp overview. URL: https://lwn.net/Articles/656307

[20] Rouleau, S. (2008). Process Monitor Hands-On Labs and Examples. URL: https://blogs.technet. microsoft.com/appv/2008/01/24/process-monitor-hands-on-labs-and-examples/

Журавчак Д. Ю.

МОНІТОРИНГ ВІРУСІВ-ВИМАГАЧІВ ЗА ДОПОМОГОЮ РОЗШИРЕНОГО БЕРКЛІЙСЬКОГО ПАКЕТНОГО ФІЛЬТРА (EBPF) ТА МАШИННОГО НАВЧАННЯ

У цій роботі досліджено актуальність захисту інформаційних систем від шкідливих програм-вимагачів у контексті стрімкого розвитку кіберзлочинності, особливо в умовах війни в Україні. Стаття фокусується на недоліках традиційних антивірусних рішень та необхідності впровадження нових технологічних рішень для ефективного моніторингу мережевого трафіку. Одним із висунутих рішень є використання розширеного Берклійського Пакетного Фільтра (eBPF), який дозволяє виявляти потенційно шкідливу активність без втручання у ядро операційної системи. Значення вирішення цієї проблеми не може бути переоцінене, враховуючи важкі наслідки атак вірусів-вимагачів, які можуть варіюватися від втрати важливих даних до повного паралічу інфраструктури підприємства. Також зазначено, що eBPF може відіграти важливу роль у розробці механізмів виявлення, аналізу та протидії кіберзагрозам, що є значущим як для теоретичних досліджень, так і для практичного застосування в області кібербезпеки. Подальший зміст статті розкриває проблематику використання eBPF для моніторингу діяльності вірусів-вимагачів та їх виявлення у мережевому трафіку. Пропонується комплексний підхід до оцінювання алгоритмів машинного навчання, що можуть застосовуватися для аналізу даних, зібраних за допомогою eBPF, з акцентом на алгоритм опорних векторів (SVM) як найбільш придатний для виявлення програм-вимагачів. В роботі також оцінено ефективність SVM у порівнянні з іншими інструментами, що підкреслює його переваги для задач кібербезпеки. Завершується стаття обговоренням можливостей впровадження eBPF у практичну сферу кібербезпеки, зокрема його вплив на підвищення ефективності захисних систем, зниження часу реакції на інциденти та забезпечення більш динамічного та гнучкого управління безпекою. Автори підкреслюють, що розв'язання цієї проблеми має значне важливість для наукового співтовариства та практичного застосування, пропонуючи нові напрямки для подальших досліджень у сфері кібербезпеки

Ключові слова: ebpf, кібербезпека, віруси-вимагачі, виявлення загроз, машинне навчання, SVM.

Zhuravchak D. Y.

RANSOMWARE MONITORING WITH ENHANCED BERKELEY PACKET FILTER (EBPF) AND MACHINE LEARNING

This paper explores the relevance of protecting information systems from ransomware in the context of the rapid development of cybercrime, especially in the context of the war in Ukraine. The article focuses on the shortcomings of traditional anti-virus solutions and the need to introduce new technological solutions for effective monitoring of network traffic. One of the proposed solutions is the use of an enhanced Berkeley Packet Filter (eBPF), which allows detecting potentially malicious activity without interfering with the operating system kernel. The importance of solving this problem cannot be overestimated, given the severe consequences of ransomware attacks, which can range from the loss of important data to complete paralysis of the enterprise infrastructure. It is also noted that the eBPF can play an important role in developing mechanisms for detecting, analyzing and countering cyber threats, which is significant for both theoretical research and practical application in the field of cybersecurity. The rest of the article reveals the issues of using eBPF to monitor the activities of ransomware and detect it in network traffic. A comprehensive approach to evaluating machine learning algorithms that can be used to analyze data collected by eBPF is proposed, with an emphasis on the support vector machine (SVM) algorithm as the most suitable for detecting ransomware. The paper also evaluates the effectiveness of SVM in comparison to other tools, which emphasizes its advantages for cybersecurity tasks. The article concludes with a discussion of the possibilities of implementing eBPF in the practical field of cybersecurity, in particular, its impact on improving the efficiency of security systems, reducing incident response times, and ensuring more dynamic and flexible security management. The authors emphasize that solving this problem iis of significant importance for the scientific community and practical application, suggesting new directions for further research in the field of cybersecurity

Keywords: ebpf, cybersecurity, ransomware, threat detection, machine learning, SVM.

Стаття надійшла до редакції 30.11.2023 р.
Прийнято до друку 19.12.2023 р.