DOI 10.34185/1562-9945-3-152-2024-04
УДК 004.051

В.С. Горбатов, А.О. Журба

ДОСЛІДЖЕННЯ ВПЛИВУ ІСНУЮЧИХ АЛГОРИТМІВ ОПТИМІЗАЦІЇ ОБРОБКИ ПРАВИЛ НА ШВИДКОДІЮ СИСТЕМИ ВИЯВЛЕННЯ МЕРЕЖЕВИХ ВТОРГНЕНЬ SNORT 3

Анотація. Системи виявлення вторгнень у мережу (NIDS) є ключовим компонентом кібе рбезпеки, працюючи на попередженні, виявленні та реагуванні на потенційні загрози в мережі. Вони аналізують мережевий трафік для виявлення аномальних або зловмисних дій, таких як спроби несанкціонованого доступу, віруси, експлуатація програмного забез печення та інше. Для високої ефективності системи виявлення вторгнень мають вико нувати інспекцію пакетів на швидкості кабелю або близько до неї. Швидкість роботи систем виявлення вторгнень має вирішальне значення, оскільки вона дозволяє вчасно ви явити потенційні кіберзагрози, забезпечуючи безперервну роботу бізнес процесів. Snort 3 є розвитком однієї з найпопулярніших систем виявлення вторгнень Snort, і є відкритою багатопотоковою системою виявлення вторгнень, яка працює в операційних системах подібних до UNIX.

У цьому дослідженні розглянута архітектура системи Snort 3, а також основні алгорит ми оптимізації обробки правил та їх вплив на швидкодію системи в різних сценаріях. Швидкодія системи вимірювалася за часом обробки запису мережевого трафіку, який міс тить як звичайні робочі пакети, так і шкідливі, на двох різних конфігураціях.

Ключові слова: Snort 3,NIDS, швидкодія, Fast Pattern, Виявлення атак, Правило, Сигнатура, Оптимізація, Алгоритм, Протокол.

Постановка задачі. У світі швидкісних мереж і складних атак, швидкодія систем виявлення мережевих вторгнень(NIDS) стає ключовим фактором у забезпеченні безпеки та функціональності цифрових інфраструктур. Системи виявлення вторгнень сканують мережевий трафік для виявлення аномальних або зловмисних активностей, таких як спроби несанкціонованого доступу, та використання шкідливого програмного забезпечення та експлойтів. Щоб бути високоефективним, NIDS повинні аналізувати пакети вхідного трафіку з максимальною швидкістю, близькою до швидкості каналу передачі даних, тобто мільйони байт в секунду. Швидкодія таких систем виявлення вторгнень є критич-

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

ною, оскільки затримки у роботі корпоративних мереж можуть призвести до зниження ефективності роботи цілого підприємства.

Системи виявлення вторгнень на основі сигнатур, працюють за принципом аналізу мережевого трафіку та виявлення в ньому певних шаблонів атак, які відомі за попередніми досвідом або базою даних[1]. Вони використовують правила (сигнатури) атак для визначення потенційно небезпечних дій у мережі і надають можливість реагувати на ці загрози у реальному часі. Кожне правило складається з набору умов та текстових шаблонів, які використовуються для перевірки кожного мережевого пакета.

Ці правила організовані у деревоподібні структури, де кожен вузол представляє собою опцію, а тіло правила, яке визначає додаткові умови, розташоване на листках цього дерева. З урахуванням можливості наявності десятків тисяч правил, застосовуються різні техніки оптимізації для звуження області пошуку та забезпечення ефективного функціонування системи. Знання специфіки роботи і впливу цих алгоритмів на систему допоможе досягти максимальної швидкодії системи NIDS при побудові інфраструктури.

Аналіз останніх досліджень і публікацій. Одним із найпоширеніших та визнаних інструментів у сфері NIDS є система виявлення вторгнень Snort, що вже зарекомендувала себе як потужний засіб захисту мереж. Snort 3 є оновленою версією цієї системи, і володіє багатопотоковістю, підвищеною швидкодією в порівнянні зі Snort, більшою модульністю та іншими перевагами[2], тож розглядати у контексті цієї статті будемо саме її.

Задача оптимізації роботи NIDS стоїть дуже гостро та розглянута у багатьох проекціях. Через варіативність та багатофункціональність існуючих систем є широке поле для аналізу та покращення ефективності NIDS як для специфічних задач так і для задач широкого профілю. Так багато робіт розглядають швидкодію Snort 3 у порівнянні з іншими системами виявлення вторгнень[3] у різних типах інфраструктур, що допоможе користувачу знайти найкращий варіант для себе.

Загалом, для звичайного користувача шляхами покращення швидкодії системи є конфігурація, набори правил та написання своїх правил. Більшість авторів надають рекомендації щодо оптимізації саме набору правил, використовуючи різні набори[4] або навіть пропонуючи методи Data Mining для покращення набору правил[5]. Одним з важливих кроків також є системна оптимізація мережевих взаємодій, один із видів якої наведено у статті[6].

Алгоритмічні оптимізації, схожі на ті що описані у цій статті, пропонуються у декількох роботах. Так деякі автори змогли інтегрувати Snort з нейронними мережами[7] або евристичними алгоритмами для зменшення кількості правил що перевіряються[8].

Мета дослідження - розглянути три основні алгоритми оптимізації обробки правил що використовуються у системі Snort 3, а саме Fast Pattern, групування що базується на портах та групування що базується на протоколах. Для них буде описана базова реалізація, модифікації вихідного коду, що необхідні для вимкнення алгоритму а також вплив алгоритму на загальну швидкодію системи.

Результати дослідження можуть бути використані для оптимізації роботи системи при її локальному розгортуванні, при розробці нових алгоритмів оптимізації обробки правил, а також для інтегрування описаних алгоритмів у інші системи.

Викладення основного матеріалу дослідження.

Обробка пакетів у системі Snort 3 є гнучкою за допомогою підходу, керованого подіями[9], діаграма якого показана на рис. 1.

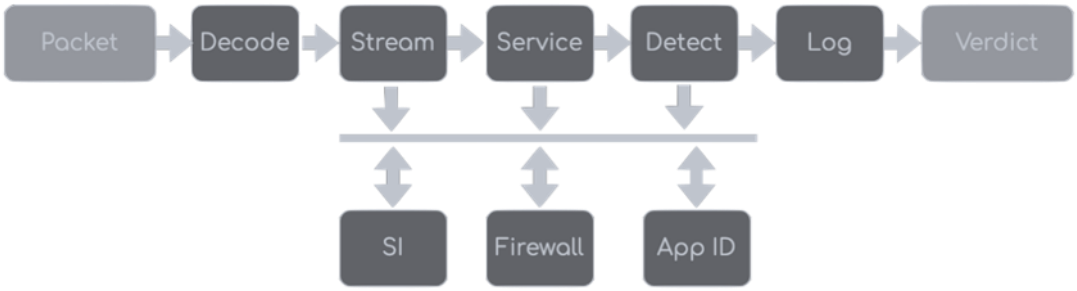

Рисунок 1 – Схема модульної архітектури Snort 3

Основні кроки обробки:

- Декодинг кожного пакету, щоб визначити основні його характеристики.
- Попередня обробка кожного пакету, для дефрагментації повного набору даних (пересбірка TCP сесії).
- Виявлення атак – це етап порівняння правил з бази даних з даними пакету.
- Логування – де система зберігає будь-яку відповідну інформацію, отриману в результаті попередніх кроків[9].

У контексті цієї роботи, нас цікавить саме етап виявлення атак. Так у ньому для кожного шаблону правила створюється дерево параметрів виявлення,

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

яке дозволяє Snort ефективно обробляти набір правил. Нелистові вузли в цьому дереві посилаються на екземпляр опції для перевірки. А листові вузли описують тіло правила. До листа приєднаний один або декілька RTN, які представляють заголовок правила. Критерії RTN оцінюються в останню чергу, щоб визначити, чи потрібно генерувати подію[10].

Через те що кількість правил у системі Snort 3 може досягати десятків тисяч, кожне з яких містить у собі декілька опцій, загальна кількість нелисткових вузлів може досягати сотень тисяч штук. Якщо кожен пакет буде потребувати обробки кожного нелисткового вузла, швидкодія системи буде явно не задовільна. Для цього у Snort 3 є декілька алгоритмів звуження пошуку

Основним методом є попередня фільтрація використовуючи Fast Pattern. Fast Pattern це маленький текстовий шаблон, що націлений не на виявлення атаки а на пошук пакета у якому потенційно може бути специфічна атака або набір атак. Fast Pattern є частиною сигнатури і може бути призначений як користувачем так і системою виявлення вторгнень.

Під час конфігурації кожне правило розбирається, і з неї виділяються всі шаблони. Якщо користувач вказав конкретний шаблон для використання як fast pattern, то він буде обраний; в іншому випадку обирається найдовший шаблон. Усі обрані шаблони додаються до загального масиву шаблонів fast pattern, які компілюються в MPSE дерево.

MPSE (багатошаблонний пошуковий двигун) - це система, яка може одночасно шукати декілька шаблонів у даних. У Snort 3 доступні декілька таких систем, зокрема AC_BNFA та Intel Hyperscan. Усі шаблони fast pattern додаються до цієї системи та компілюються у таблицю пошуку за алгоритмом Ахо-Корасіка для AC_BNFA та у константну базу даних для Hyperscan. Після компіляції MPSE дерева для кожного fast pattern формується дерево правил.

Під час роботи, кожен пакет перед пошуком атаки проходить через MPSE, де шукається fast pattern. Якщо шаблон було знайдено, то для цього пакету обробляється відповідне дерево перевірок; в іншому випадку можна припустити, що у пакеті атака відсутня [11].

Також Snort 3 зберігає дерево правил що не мають fast pattern опцій(а саме content, regex або sd_pattern). Вони виконуються на кожен пакет і потребують багато ресурсів.

Розподіл правил на групи за портами використовується для запобігання перевірці правил, RTN яких точно що не може відповідати пакету. Наприклад, усі правила TCP, які використовують порт 80, потраплять до однієї групи, а всі

правила, які використовують порт 22, – до іншої групи[9]. Ці групи правил скомпільовані в MPSE, таким чином система містить в собі не одну велику, а N малих пошукових систем, що працюють базуючись на характеристиках пакету, які є простою класифікуючою функцією.

Основний процес розподілу проходить на етапі конфігурації, де створюється набір таблиць відповідності ідентифікаторів правил та унікального порту або напрямку який ці правила очікують. Далі, після пошуку в правилах Fast Pattern, на основі таблиці створюються порт групи, де до кожного унікального порту додається MPSE система, скомпільована під шаблони Fast Pattern що зустрічаються у правилах які очікують цього порту[10].

Також, система оперує додатковою групою any-any, для правил що націлені на все порти, без залежності на напрямок. Такі правила виділені у групу і перевіряються для кожного пакету.

Розподіл правил на групи за протоколами також використовується, адже порт не завжди може точно характеризувати тип трафіку (адже ніхто не заважає HTTP серверу працювати на проту відмінному від 80). Так NIDS проводить передобробку пакетів, це необхідно з декількох причин, таких як нормалізація даних, виявлення аномалій, та ін. Тож, система завжди знає справжній протокол до якого відноситься пакет, без залежності на порт. Це досягається за допомогою пошуку характеристик у реальних даних пакету. Тож, класифікація правил за протоколом є більш ефективною ніж просто за портами.

У Snort 3 є декілька способів класифікувати правило за протоколом:

- вказати протокол у заголовці правила;
- використати спеціальну опцію service;
- використати один з буферів, специфічних для деякого протоколу.

Далі система використовує реалізацію на основі порт груп, одначе замість порту використовується внутрішній ідентифікатор протоколу. Тобто для кожного протоколу та напрямку будується група, кожен пакет перед перевіркою звертається до набору груп і отримує відповідну своєму протоколу групу правил.

Для вимірювання впливу вищеописаних алгоритмів на швидкодію системи, було проведено деякі зміни у кодовій базі системи Snort 3, що дозволило вимкнути оптимізації.

Також була проведена валідація змін. Очікується, що кількість спрацювань правил на одному тому ж трафіку та з однаковою конфігурацією не повинна змінитися. Перевірка виконувалася на записі трафіку DEFCON-2003, що

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

містить як атаки так і звичайний трафік. У якості конфігурації використовувався пакет Talos LightSPD з конфігурацією Maximum Detection[12]. Порівняння відбувалося на Snort 3 версії 3.1.78.0 та версії з відповідними змінами, базуючись на 3.1.78.0.

Для вимкнення розподілення по портам, простіше всього скласти усі правила у одну групу. Для цього можна створити окрему групу, або використати вже існуючу групу any-any. У цій роботі була використана група any-any, адже вона виконується на кожен пакет, тож не потребує багато кодових змін. Тож, була змінена функція формування таблиць портів, так щоб усі правила складались у групу any-any. Валідація показала ідентичність результатів.

Для вимкнення функціоналу протокольних груп був використаний схожий підхід. На етапі створення таблиці сервісів, код був модифікований таким чином, щоб складати усі сервіси в одну групу. Також був модифікований код пошуку групи, що виконується при обробці пакета, так він одну повертає статичну групу.

Валідація показала зміну в кількості спрацювань, на 1434 з 29496 спрацювань. Подібні зміни не є критичними і обумовлені злиттям схожих правил.

Для вимкнення функціоналу Fast Pattern треба зазначити, що створення MPSE для Fast Pattern перевірок проходить після парсингу усіх правил. Для того щоб Snort зрозумів чи є у правилі опція що підтримує Fast Patter, він звертається до вектора шаблонів правила. У випадку коли вектор не пустий, система підбирає відповідний патерн з вектора і додає його в MPSE. Для вимкнення Fast Pattern код було модифіковано таким чином, що перевірка вектора завжди повертає нульове значення, тож правило додається лише у список Non Fast Pattern правил та перевіряється на кожному пакеті.

Валідація показала збільшення кількості спрацювань на 530 штук. Така зміна є очікуваною, адже кількість правил що перевіряються на пакеті зросла, тож вірогідно що деякі додаткові правила спрацюють.

Для замірів швидкодії було підготовлено:

- Сервер для отримання результатів;
- Конфігурації системи виявлення вторгнень;
- Записи трафіку, що включають атаки та звичайний трафік.

У якості машини для проведення бенчмарків був використаний сервер Cisco UCSC C220 M5SX, системні характеристики якого наведено у табл. 1.

Таблиця 1

Конфігурація серверу для замірів швидкодії

CPU	2 × Intel Xeon Gold 5218 CPU 2.30GHz по 16 ядер, 32 потоки
RAM	128 Gb, 2666 MT/s, 4 канали
OS	Fedora Linux 39 (Server Edition)

Для досягнення максимально точних результатів, системні процеси на сервері було максимально скорочено, та їх обробка була призначена на CPU 1. Таким чином вдалося ізолювати системні процеси на потоках 0-15,32-47 а процес для якого замірялася швидкодія на потоках 16-31,48-63. Також для того щоб виключити можливе вузьке місце у вигляді пропускної здібності диску, усі записи трафіку попередньо було скопійовано у RAM диск зі швидкістю доступу 21.3 GB/s.

Snort 3 було зібрано використовуючи компілятор clang++ версії 17.0.4 з пакету LLVM через те, що алгоритмічна опимізація аплікацій зібраних використовуючи clang у більшості випадків краща ніж GCC[13]. Також був використаний максимальний рівень оптимізацій компілятора (O3) і усі плагіни використовували динамічне зв’язування, для зменшення числа кеш-промахів.

Для кожної варіації запису трафіку та конфігурації виконувалось по 50 прогонів. У результаті підготовки серверу стандартне відхилення результатів вдалося знизити до 0.01%.

У якості конфігурацій використовувались Balanced Security And Connectivity та Maximum Detection з пакету Talos lightSPD версії 2024-02-07-001. Цей пакет надає можливість вибрати конфігурації Snort, які більше спрямовані на швидкість або більше на виявлення та глибину перевірки[12]. Перша надає збалансовану конфігурацію, що включає правила для більшості відомих атак, друга, відповідно, надає максимальний рівень захисту з усіма наявними правилами.

Записи трафіку що використовувалися, включаються як звичайну роботу в мережі так і атаки. Для цього були використані наступні записи:

- 2000 DARPA Intrusion Detection Scenario, що включає в себе набір атак на Windows NT [14];
- National CyberWatch Mid-Atlantic Collegiate Cyber. Defense Competition 2012, запис 00008, що містить атаки на вбудовані системи медичного обладнання[15];

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

• Відокремлену частину(9:20 - 10:42) запису CIC-IDS2017-Thursaday-Working-Hours, що включає Brute Force, XSS та SQL Injection атаки[16].

Далі наведені результати замірів швидкодії системи Snort 3 зі змінами описаними вище. Для кожного варіанту було отримано базовий результат та результат зі змінами. Відсоткова зміна вирахувана за формулою (1).

$$\% = \frac{n_2 - n_1}{|n_1|} * 100\% \quad (1)$$

У табл. 2 наведені результати вимірювання швидкодії NIDS з вимкненим функціоналом порт груп. Як можна побачити, у випадку зі збалансованою конфігурацією та трафіком DARPA, швидкодія зросла на майже 5%. Скоріш за все це відбулося через специфіку трафіку і те, що кількість правил що надходила на доволі мала, тож виграш від видалення пошуку групи переважив втрати на MPSE пошук. У всіх інших випадках спостерігається падіння швидкодії до 30%, що зростає при переході на конфігурацію з більшою кількістю правил.

Результати з вимкненим функціоналом протокольних груп наведено у табл. 3. Тут зберігається покращення швидкодії при вимкнених групах у випадку зі збалансованою конфігурацією. На мою думку, причини збігаються з попереднім дослідженням. Також можна побачити падіння швидкодії на 30% та більше у випадку з великою кількістю правил. Таке велике падіння швидкодії логічне, адже кількість правил що використовують протокольні групи більша ніж тих шо використовують порт групи.

Таблиця 2

Результати замірів швидкодії з вимкненим функціоналом порт груп

Трафік	Конфігурація Balanced Security and Connectivity			Конфігурація Maximum Detection		
	Базовий результат, с.	Цільовий результат, с.	Різниця, %	Базовий результат, с.	Цільовий результат, с.	Різниця, %
DARPA-2000-Win-NT-attacks	11,051	10,554	-4,71	13,484	19,44	30,64
CIC-IDS2017-Thursaday-attacks	7,554	8,407	10,15	15,412	20,435	24,58
maccdc2012_00008	32,471	42,442	23,49	130,642	163,328	20,01

Таблиця 3

Результати швидкодії з вимкненим функціоналом протокольних груп

Трафік	Конфігурація Balanced Security and Connectivity			Конфігурація Maximum Detection		
	Базовий результат, с.	Цільовий результат, с.	Різниця, %	Базовий результат, с.	Цільовий результат, с.	Різниця, %
DARPA-2000-Win-NT-attacks	11,044	10,72	-3,02	13,447	21,271	36,78
CIC-IDS2017-Thursaday-attacks	7,563	7,104	-6,46	15,423	23,372	34,01
maccdc2012_00008	32,618	32,756	0,42	130,892	188,739	30,65

Вимкнення Fast Pattern, призвело до сильного падіння швидкодії, що можна побачити у табл. 4.

Таблиця 4

Результати замірів швидкодії з вимкненим Fast Pattern

Трафік	Конфігурація Balanced Security and Connectivity			Конфігурація Maximum Detection		
	Базовий результат, с.	Цільовий результат, с.	Різниця, %	Базовий результат, с.	Цільовий результат, с.	Різниця, %
DARPA-2000-Win-NT-attacks	10,985	116,285	958.57	13,481	296,176	2096.98
CIC-IDS2017-Thursaday-attacks	7,539	22,487	198.27	15,389	150,565	878.39
maccdc2012_00008	31,9	148,511	365.55	130,904	2120,38	1519.79

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Падіння лінійно збільшується в залежності від кількості правил. Специфіка трафіку теж сильно впливає на падіння швидкодії, адже чим більший середній розмір пакету, тим більше шанс того що при перевірці на ньому будуть оброблятися вузли опцій.

Висновки. Система Snort 3 є однією з найефективніших NIDS на ринку, це досягається шляхом використання великої кількості цікавих алгоритмів оптимізації. У цій роботі були розглянуті три основні алгоритми оптимізації обробки правил(порт та протокольні групи, Fast Pattern) та їх вплив на швидкодію системи при обробці трафіку. Результати показали значний їх вплив а також пряму залежність швидкодії на кількість правил що використовуються.

Деякі результати показали незначне покращення швидкодії при відключенні алгоритмів оптимізації, це відбувається на конфігураціях з невеликою кількістю правил. У більшості випадків помітно явне падіння швидкодії від 10% і вище. Найбільше погіршення швидкодії відбувається при відключенні операцій Fast Pattern, без цього алгоритму погіршення може досягати 20 разів.

Результати роботи можуть бути використані при написанні правил та конфігурацій для NIDS Snort 3, оптимізації самої системи а також для інтеграції цих алгоритмів у інші системи. Схожі алгоритми можуть бути інтегровані у пошукові системи, бази даних, системи аналізу даних, т.д.

ЛІТЕРАТУРА

1. Martin R. Snort: Lightweight intrusion detection for networks. LISA '99 : матеріали Міжн. наук. конф., м. Вашингтон, 7–12 груд. 1999 р. 1999. С. 229–238.

2. Comparing Snort 2 and Snort 3 on Firepower Threat Defense – Cisco [Електронний ресурс]. – Режим доступу:
https://www.cisco.com/c/en/us/support/docs/security/
firepower-ngfw/217617-comparing-snort-2-and-snort-3-on-firepow.html.

3. Thorarensen C. A Performance Analysis of Intrusion Detection with Snort and Security Information Management : магістерська робота. Linköping, 2021. 84 с.

4. Granberg N. Evaluating the effectiveness of free rule sets for Snort : магістерська робота. Linköping, 2022. 53 с.

5. Khamphakdee N., Benjamas N., Saiyod S. Improving intrusion detection system based on snort rules for network probe attacks detection with association rules technique of data mining. Journal of ICTResearch & Applications. 2015. Т. 8, № 3. С. 11–21.

6. Salah K., Kahtani A. Improving Snort performance under Linux. IET Communications. 2009. Т. 3, № 12. С. 1883–1895.

7. Elshafie H. M., Mahmoud T. M., Ali A. A. Improving the Performance of the Snort Intrusion Detection Using Clonal Selection. International Conference on Innovative Trends in Computer Engineering (ITCE) : матеріали Міжн. наук. конф., м. Асуан, 2–4 лют. 2019 р. 2019. С. 104–110.
8. Singh P., Behal S., Kumar K. Performance enhancement of a Malware Detection System using score based prioritization of snort rules. 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), м. Greater Noida, Delhi, India, 8–10 жовт. 2015 р. 2015. С. 1150-1155.
9. Snort 3 User Manual [Електронний ресурс]. – Режим доступу: https://usermanual.wiki/Document/snortmanual.760997111/view
10. Snort++ Developers Guide [Електронний ресурс]. – Режим доступу: https://www.snort.org/downloads/snortplus/snort_devel.html#_detection.
11. Горбатов В. С., Журба А. О. Метод попередньої фільтрації сигнатур для прискорення пошуку атак системою виявлення мережевих вторгень. Сучасні Інформаційні та комунікаційні технології на транспорті, в промисловості та освіті: матерівли Міжн. наук. конф., м. Дніпро, 13–14 груд. 2023 р. Дніпро, 2023. С. 136–137.
12. Munshaw J. Soft Release: lightSPD, the new rules package for Snort 3. Snort Blog. [Електронний ресурс]. – Режим доступу: https://blog.snort.org/2020/12/soft-release-lightspd-new-rules-package.html.
13. Performance comparison of GCC and LLVM on the EISC processor / C. Park та ін. 2014 International Conference on Electronics, Information and Communications (ICEIC), м. Kota Kinabalu, Sabah, Malaysia, 15–18 січ. 2014 р. 2014. С. 96-102.
14. 2000 DARPA Intrusion Detection Scenario Specific Datasets | MIT Lincoln Laboratory [Електронний ресурс]. – Режим доступу: https://www.ll.mit.edu/r-d/datasets/2000-darpa-intrusion-detection-scenario-specific-datasets
15. 2012 MACCDC – MACCDC [Електронний ресурс]. – Режим доступу: https://maccdc.org/maccdc-2012/
16. IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB [Електронний ресурс]. – Режим доступу: https://www.unb.ca/cic/datasets/ids-2017.html

REFERENCES

1. Martin Roesch, “Snort: Lightweight intrusion detection for networks”, Lisa 99, vol. 1, 229–238, 1999.
2. Comparing Snort 2 and Snort 3 on Firepower Threat Defense – Cisco. URL: https://www.cisco.com/c/en/us/support/docs/security/firepower-ngfw/217617-

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

comparing-snort-2-and-snort-3-on-firepow.html.
3. C. Thorarensen, “A Performance Analysis of Intrusion Detection with Snort and Security Information Management”, Independent thesis Advanced level, Linköping University-Department of Computer and Information Science, 2021.
4. Granberg, N.: "Evaluating the effectiveness of free rule sets for Snort". Independent thesis Advanced level, Linköping University-Department of Computer and Information Science, 2022.
5. N. Khamphakdee, N. Benjamas and S. Saiyod, “Improving intrusion detection system based on snort rules for network probe attacks detection with association rules technique of data mining”, Journal of ICTResearch & Applications, vol. 8, no. 3, pp. 11–21, 2015.
6. Salah, K.; Kahtani, A., “Improving Snort performance under Linux”, IET Communications, vol. 3, no. 12, p. 1883-1895, 2009.
7. H. M. Elshafie, T. M. Mahmoud, A. A. Ali, "Improving the Performance of the Snort Intrusion Detection Using Clonal Selection", 2019 International Conference on Innovative Trends in Computer Engineering (ITCE), Aswan, Egypt, 2019, pp. 104-110, 2019.
8. P. Singh, S. Behal and K. Kumar, "Performance enhancement of a Malware Detection System using score based prioritization of snort rules", 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), Greater Noida, India, 2015, pp. 1150-1155, 2015.
9. Snort 3 User Manual.
URL: https://usermanual.wiki/Document/snortmanual.760997111/view
10. Snort++ Developers Guide.
URL: https://www.snort.org/downloads/snortplus/snort_devel.html#_detection.
11. Gorbatov V. S., Zhurba A. O. The method of pre-filtering signatures to speed up the search for attacks by the network intrusion detection system. Modern information and communication technologies in transport, industry and education: materials of the international science conference, Dnipro, December 13–14. 2023. Dnipro, 2023. P. 136–137.
12. Munshaw J. Soft Release: lightSPD, the new rules package for Snort 3. Snort Blog. URL: https://blog.snort.org/2020/12/soft-release-lightspd-new-rules-package.html.
13. C. Park, M. Han, H. Lee, M. Cho, S. W. Kim, “Performance Comparison between LLVM and GCC Compilers for the AE32000 Embedded Processor,” IEIE Transactions

on Smart Processing and Computing, vol. 3, no. 2. The Institute of Electronics Engineers of Korea, 96–102, 2014.
14. 2000 DARPA Intrusion Detection Scenario Specific Datasets | MIT Lincoln Laboratory. URL: https://www.ll.mit.edu/r-d/datasets/2000-darpa-intrusion-detection-scenario-specific-datasets
15. 2012 MACCDC – MACCDC. URL: https://maccdc.org/maccdc-2012/
16. IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB. URL: https://www.unb.ca/cic/datasets/ids-2017.html

Received 05.04.2024.
Accepted 08.04.2024.

Influence of existing rule processing optimizations on the performance of the snort 3 network intrusion detection system

Network intrusion detection systems (NIDS) are a key component of cybersecurity, working to warn, detect, and respond to potential network threats. They analyze network traffic to detect anomalous or malicious activity such as breach attempts, viruses, use of software exploits, and more. Intrusion detection systems should perform packet inspection at or near cable speed to be highly effective. The speed of intrusion detection systems is critical because it allows timely mitigation of potential cyber threats, ensuring uninterrupted operation of business processes.

One of the most common and recognized tools in the field of NIDS is the intrusion detection system Snort, which has already proven itself as a powerful means of protecting networks. Snort 3 is an updated version of this system, and has multithreading, increased speed compared to Snort, greater modularity and other advantages[2], so we will concentrate on it in the context of this article.

The task of optimizing the operation of NIDS is very acute. Due to the variability and multifunctionality of existing systems, there is a wide field for analyzing and improving the efficiency of NIDS both for specific tasks and for tasks of a broad profile. So many works look at the performance of Snort 3 compared to other intrusion detection systems[3] in different types of infrastructures, which will help the user to find the best option for himself.

The purpose of the study is to consider the three main rule processing optimization algorithms used in the Snort 3 system, namely Fast Pattern, port-based and protocol-based clustering. For them, the basic implementation, modifications of the source code, which are necessary to disable the algorithm, as well as the impact of the algorithm on the overall speed of the system, will be described.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Some results have shown a slight performance improvement when the optimization algorithms are disabled, this is on configurations with a small number of rules. In most cases, a clear drop in performance of 10% or more is noticeable. The biggest deterioration in performance occurs when Fast Pattern operations are disabled, without this algorithm the deterioration can reach 20 times.

Горбатов Віталій Сергійович - аспірант кафедри Інформаційних технологій і систем ІПБТ УДУНТ.
Журба Анна Олексіївна - доцент кафедри Інформаційних технологій і систем ІПБТ УДУНТ.

Gorbatov Vitalii - post-graduate student, Department of information technology and systems, Institute of industrial and business technologies, Ukrainian state university of science and technologies.
Zhurba Anna - assistant professor, Department of information technology and systems, Institute of industrial and business technologies, Ukrainian state university of science and technologies.