УДК: 004.712 519.172.4

Методи покращення ефективності для систем високошвидкісної класифікації пакетів

Ю. В. Бойко, К. С. Дєєв
Київський національний університет імені Тараса Шевченка

В статті розглянуті методи та підходи в реалізації класифікаторів мережевого трафіку. Вказані інструменти використовуються як детектори аномальної мережевої активності, які засновані на імплементації алгоритму Aho-Corasick. Основна частина роботи присвячена огляду шляхів підвищення ефективності роботи класифікатора та мінімізації часу обробки мережевого пакету потрібного для визначення його належності до окремого класу трафіку. Висновки, отримані в роботі, можуть бути використані для створення розподіленої системи класифікації пакетів з оптимальною архітектурою.

Ключові слова: *Мережевий моніторинг, аналіз трафіку, класифікація пакетів, виявлення вторгнень.*

В статье рассмотрены методы и подходы в реализации классификаторов трафика. Указанные инструменты используются как детекторы аномальной сетевой активности, которые основаны на имплементации алгоритма Aho-Corasick. Основная часть работы посвящена обзору путей повышения эффективности работы классификатора и минимизации времени обработки сетевого пакета, требующегося для определения его принадлежности к определенному классу трафика. Выводы, полученные в работе, могут быть использованы для создания распределённой системы классификации пакетов с оптимальной архитектурой.

Ключевые слова: *Сетевой мониторинг, анализ трафика, классификация пакетов, обнаружение вторжений.*

The article describes methods and approaches chosen for development of network traffic classifiers. These tools usable as detectors of abnormal activities are based on Aho-Corasick algorithm implementation. The main part of the work is devoted to the overview of ways to improve the classifiers efficiency as well as to minimization of processing time needed to detect traffic class the particular network packet belongs to. Obtained conclusions can be used to create distributed classification system with optimal architecture.

Keywords: *Network monitoring, traffic analysis, packet classifying, intrusion detection system.*

Вступ

За останні роки можна спостерігати велике підвищення загроз безпеці критичних компонентів сучасних мережевих архітектур. Ці системи працюють на основі політик послідовної перевірки ряду наперед сформованих правил та відповідних процедур, які необхідно виконати при співпадінні того чи іншого правила для окремого мережевого пакета чи композиції таких пакетів (режим перевірки сесій). Будемо вважати, що існує набір правил: $\langle R_1, ..., R_n \rangle$, класифікатор ідентифікує набір правил, які мають співпасти для позитивного спрацювання. Таким чином, класифікація пакетів це механізм що встановлює відповідність приналежності пакету до окремого правила шляхом перевірки мережевого пакету чи окремих його заголовків та встановлення як ця відповідність має бути в подальшому відпрацьована для аналізу. Пакетні

класифікатори таких систем мають працювати в режимі реального часу на високошвидкісних інтерфейсах мережевих плат та одночасно мінімізувати чи зовсім відкидати можливість помилкових спрацювань чи втрати пакетів[1]. Більш-того, розвиток технологій вимагає від вказаних систем змін які стосуються процедур обробки пакетних заголовків, а саме, якщо раніше достатньо було перевіряти відповідність IP-адрес та пізніше номерів портів то тепер необхідно аналізувати більш широкий набір параметрів: TCP-ідентифікатори з'єднань, TTL, поля TOS/DSCP. Спроможна здатність є одним з основних показників кількісних характеристик комплексу. Можливість аналізувати корисне навантаження в режимі реального часу потребує мережевих карт з збільшеними пакетними буферами.

Мета даної роботи сформувати швидку і масштабовану систему пакетної класифікації, яка б могла застосовуватись у різних випадках. Проблема класифікації пакетів має дві несхожі компоненти:

- співпадіння на основі пакетних заголовків;
- глибока перевірка пакетного навантаження;

Кожен з підходів має свої недоліки та переваги, які будуть обговорені в слідуючих розділах.

1. Методи аналізу заголовків

Декілька програмних розробок визначають правило яке може застосовуватись до пакету ґрунтуючись на значенні декількох полів IP-заголовку мережевого пакету. Швидкодія систем на основі перевірки списку правил для кожного пакету достатньо істотно деградує із збільшенням кількості правил чи створенні комплексних схем перевірки, які потребують співпадінь по багатьом критеріям. Попередні техніки що використовуються в класифікаторах також залежать від даного ефекту, але використовували різні оптимізації, зокрема представлення шаблону у вигляді бінарного дерева [10], або використання невизначеного автомату станів[2,3].

Але вони не спростовували залежності від кількості правил, що було неодноразово перевірено на практиці, коли кількість правил досягає декількох тисяч, як приклад у системі Snort NG[13]. Емпірична формула показує збільшення часу аналізу в 50 разів на кожні нові 2500 правил, тобто залежність порядку $\sqrt{N}$, де N – кількість нових правил. Більш того, різні додатки потребують різної поведінки при співпадіння пакета за рядом критеріїв. Частина застосунків можуть потребувати лише пакетної фільтрації, іншим може бути необхідна також класифікація. Попередні дослідження намагалися вирішити проблему шляхом перетворення(BPF, DPF, PathFinder) [4], та зосереджувалися на пакетній фільтрації. Опис алгоритму множинної класифікації за ключовими ознаками описано в [13], але слід зазначити що він не підтримує пріоритетів.

2. Постановка проблеми

За основні критерії в визначенні продуктивності скінченого автомата класифікатора в даній роботі є підтримка різних застосунків, час спрацювання та розмірність автомата. Представлений механізм дозволяє проводити класифікацію за допомогою спрощеного інтерфейсу взаємодії.

В алгоритмі використовується операція розкладу на стани, які засновані на понятті залишкового стану по відношенню до іншого. Якщо розглядати аналогії, то це операція подібно до взяття остачі від ділення цілих чисел, так само як поділ забезпечує основу. Такі конструкції важливі в плані оптимізації розташування правил при побудові автомата станів, оскільки в теорії це може мінімізувати розмірність кінцевого автомата, що в свою чергу впливає на швидкість роботи класифікатора[7].

Створення техніки вибору порядку розташування правил має велике значення в побудові, ефективного в плані часу обробки, автомата. Використання такого підходу може вносити похибку позитивного спрацювання може призвести, але в той же час це призводить до значного скорочення часу обробки великих списків правил, тобто зниження розміру автоматів для певних наборів правил. Якщо головною метою є встановлення однозначної відповідності того чи іншого пакета до окремого класу трафіку то необхідно застосовувати автомат побудований з меншим коофіцієнтом невизначених станів.

Використання часу як основної метрики є наслідком невпинного зростання швидкості каналів Інтернет та об'ємів даних що через них передаються. Створення ефективного підходу в організації вимірів мережевого трафіку дозволить встановлювати класифікатори не лише як апаратні системи аналізу трафіку, але як і програмні комплекси пост-обробки перехопленого трафіку.

Дану проблематику розглянуто такими аторами, як L. Bailey, B. Gopal, A. Pagels, L. Peterson, T. Lakshman, D. Stiliadis, Z. Chen, Y. Diao, T. Lakshman та ін.

3. Огляд рішень

Декілька систем мережевого моніторингу та систем мережевої безпеки обробляють набір пакетів як одне ціле в залежності від типу протоколу корисного навантаження верхнього рівня, базуючись на інформації яка біла отримана з пакетних заголовків. Прикладом таких програмних застосунків є система Snort[20], популярна відкрита IDS, яка порівнює пакети засновуючись на сигнатурах описаних через спеціальні правила. Розвиток вказаної системи йшов шляхом створення ефективних методів опису правил аналізу співпадіння за шаблоном па подальшого розпаралелення функції їх перевірки. Ці вирази можуть бути скомпільовані для застосування в ролі автомата станів DFA (Deterministic Finite Automata). Аналіз цих випадків детально розглянуто в [5]. Розроблений алгоритм дозволяє проводити класифікацію пакетів шляхом опису правил подібних до продукту Snort. В окремому випадку система дозволяє проводити класифікацію використовуючи схему співпадіння за регулярним виразом. В подальшому для спрощення опису рішення будемо вважати що, перевірка заголовків буде називатись перевіркою пакета, а перевірка навантаження того ж пакета – пакетною інспекцією.

3.1 Проблема класифікації

Опишемо два визначення для представлення умов створення правил класифікації, будемо їх називати фільтрами. Для спрощення представлення кожен фільтр буде ідентифікувати лише одне правило.

Визначення 1. (Синтаксис)

Форма запису виразу для фільтра має відповідати наступним вимогам:

- містить змінну що перевіряється (***x***) та одну чи декілька констант (***c***);
- дозволяє використання бітових масок(***x&c1=c***);
- підтримує можливість встановлення нерівності полів заголовку (***x≠c***);
- встановлення відношень між елементми заголовку пакета що перевіряється (***x<c***);

Таким чином, правило може бути записане як:

$$(\textbf{dport} = 80)\ \&\&\ (\textbf{sport} > 1024)\ ||\ (\textbf{flags}\&0xb = 0x3) \qquad (1)$$

Фільтр C записаний у вигляді вираза (1) може бути застосований до мережевого пакета P. Відповідна операція може біти записана у вигляді $C(P)$. За співпадіння будемо вважати позитивне логічне «істина», яке базується на застосуванні фільтра до пакетного навантаження.

Визначення 2. (Пріоритетна обробка)

Фільтр F це ланцюжок перевірок C. Набір фільтрів Φ формується з одиничних записів C, які відсортовані за пріоритетом. Пріоритет F оголошується як *pri*(F).

Для набору фільтрів Φ, ми вважаємо що $F \in \Phi$, відповідає співпадінню пакета P і означає: $M_{\Phi}(F,P)$, якщо $F(P)$ істина та $F'(P)$ хибне для $\forall F' \in \Phi$, що має вищий пріоритет ніж F. Таким чином пакет не може бути оброблений фільтром до того часу коли всі пріоритетні правила будуть оброблені. Наприклад, розглянемо набір правил:

- F_1 : (***icmp_type*** = *ECHO*)
- F_2 : (***icmp_type*** = *ECHO_REPLY*) || (***ip_ttl*** = *1*)
- F_3 : (***ip_ttl*** = *1*)

Будемо вважати F_1 відповідає пакету P_1, а F_2 в свою чергу P_2.

Якщо набір би мав не встановлені пріоритети то правило F_1 спрацювало б для P_1, F_2 для P_2, а F_3 перевірялося б для обох пакетів. Тобто,

$M_{\Phi}(P_1) = \{F_1, F_2\}$ та $M_{\Phi}(P_2) = \{F_2, F_3\}$,

якщо $pri(F_1) > pri(F_2) > pri(F_3)$ то $M_{\Phi}(P_1) = \{F_1\}$ та $M_{\Phi}(P_2) = \{F_2\}$,

якщо $pri(F_3) > pri(F_2) > pri(F_1)$ то $M_{\Phi}(P_1) = M_{\Phi}(P_2) = \{F_3\}$,

якщо $pri(F_1) = pri(F_2) > pri(F_3)$ то $M_{\Phi}(P_2) = \{F_3\}$ та $M_{\Phi}(P_1) = \{F_1\} \wedge M_{\Phi}(P_1) = \{F_2\}$.

3.2 Визначення пріоритетності

Пакетна фільтрація може відбуватися за умови встановлення рівного пріоритету для всіх фільтрів. Таким чином співпадіння буде можливим одразу після першого правила що співпадає. Співпадіння за списком також можливе, для цього необхідно встановити пріоритет для кожного правила. Як наслідок перших двох, можливе співпадіння за багатьма критеріями заголовку[11].

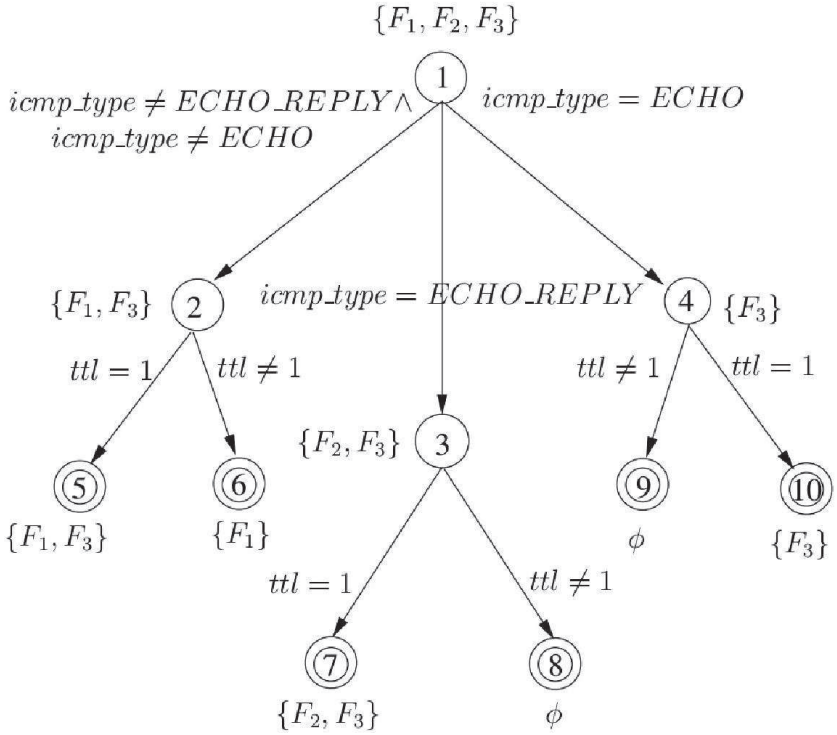

Рис. 1 Автомат скінченних станів

Схеми автоматів для наборів фільтрів з різним пріоритетом співпадіння по шаблону наведено на Рис.1 та Рис. 2. Такі автомати відомі як класифікаційні. Переходи між станами у невизначеному автоматі на Рис. 2 характеризуються наявністю початкових умов в вхідних правилах.

Під час виконання класифікації будемо вважати що деякий мережевий пакет задовольняє умові T_i, наприклад, ***icmp_type*** = *ECHO*. Перехід відбувається якщо T_i, не трапляється раніше у наборі правил або перед ним не стоїть правила з більшим пріоритетом. Перевірку полів заголовка ip-пакета можливо виконувати в будь-якій послідовності. Якщо продовжувати розглядати, наприклад з ***icmp_type,*** то поле буде розташоване за зміщенням 35 байтів, рахуючи від початку ip-заголовка, перед полем ***ttl***, яке розташоване за 13 байт раніше.

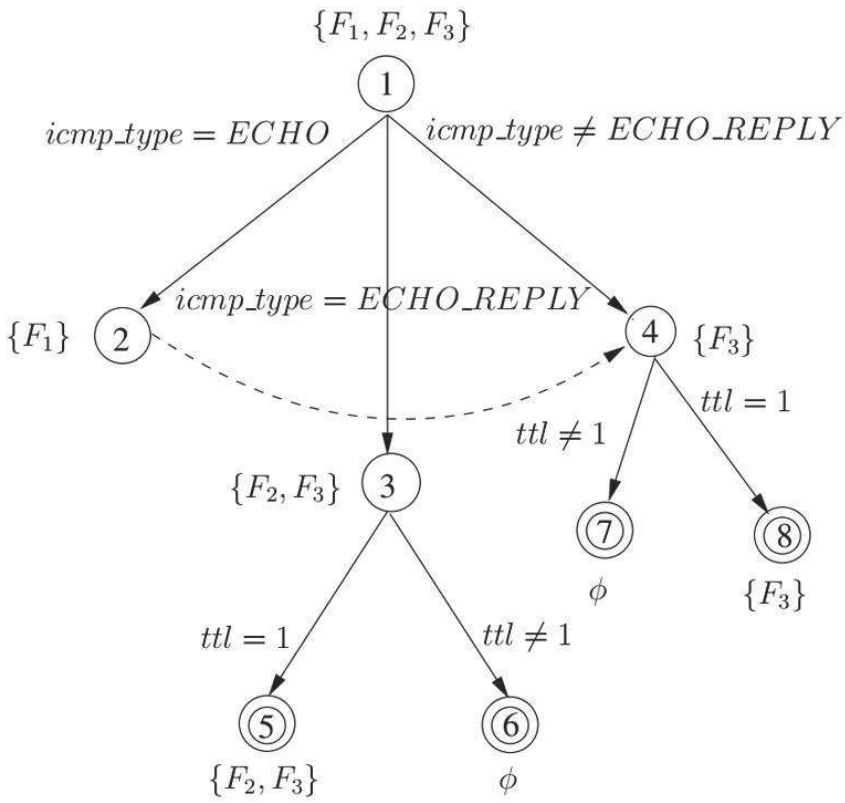

Рис.2 Невизначений автомат збігу станів

Використовуючи формат опису правил подібний до того що описано в [6], процес встановлення правил зводиться до прямого опису зміщень полів які необхідно досліджувати. Перед компіляцією кінцевих правил поводиться аналіз синтаксису на предмет взаємо-виключаючих станів.

Тобто збережена логічна послідовність проведення тестів:
*{**proto** = ICMP} : (**icmp_type** = ECHO),*
перевірка типу повідомлення можлива, якщо тип пакету – ICMP.

Існує дві кількісні характеристики роботи автомата – його розмір та час спрацювання. Загалом більшість фільтрів містить невелику кількість тестів в той час як кількість самих фільтрів досить велика. В результаті довжина шляху проходу в автоматі є короткою в порівнянні з його шириною.

Час спрацювання автомата прямо залежить від його довжини. Середнє значення буде залежати від розподілу пакетів в час проведення тесту.

4. Результати

Якщо застосовувати запис полів за відомим зміщенням, можна проводити аналіз вмісту пакета за допомогою перевірки відповідності текстовому регулярному виразу. Визначення збігу з описом в регулярних виразах є досить

популярним в системах глибокого аналізу пакетів (DPI). Для розширення можливостей обробника необхідно виконання двох умов:

- фіксований результат збігу;
- відома адреса зміщення в пакетному навантаженні;

Виконання першої умови важливо з огляду на попередню компіляцію правил, оскільки це задається жорсткою умовою при формуванні скінченого автомата подій. Виявлення збігу за регулярним виразом найбільш доцільно виконувати за алгоритмом Aho-Corasick [1], основою роботи алгоритму є представлення всіх значень у вигляді дерева. В результаті кінцева схема підтримки пошуку по текстовим полям в пакетному навантаження зводиться до оголошення додаткових перевірок, а саме - пошуку збігу. Процедура генерації фільтра не описується, оскільки залежить від операційної системи, ознайомитись з виразами можна на офіційному сайті документації [12]. Процедура перехоплення мережених пакетів неодноразово описувалась [5], [7]. Аналізатор може використовуватись в режимі IDS або як класифікатор трафіку для виявлення його приналежності до відповідного сервісу (поля DSCP або ToS).

Висновки

Підвищення ефективності роботи комплексів для класифікації мережевих пакетів має велике значення для систем аналізу з'єднань та запобігання вторгнень в мережу. Запропонований алгоритм показує покращення швидкості роботи класифікатора при попередній компіляції наборів правил.

Алгоритм підтримує встановлення пріоритетів для правил (для фільтрів) та можливість роботи у режимі пошуку максимальної кількості співпадінь. Запропонована техніка показує що зменшення розміру скінченого автомата досягається за рахунок мінімізації повторення проходів через класифікатор для правил що стосуються ключового поля IP-пакета. Як результат це також сприяє зменшенню часу спрацювання.

Розглянуті механізми можуть застосовуватись у побудові масштабованих систем мереженої класифікації пакетів. В подальшому планується провести інтеграцію з системою Snort, для можливості роботі системи в режимі IDS, та з набором утиліт класифікатора OpenDPI (набір бібліотек для класифікації мережевих пакетів в режимі реального часу). Застосування оптимізованих правил дозволяє проводити ефективну роботу в напрямку перехоплення пакетів на високошвидкісних каналах зв'язку Інтернет мережі та проводити статистичний аналіз цих даних з метою встановлення розподілів застосування протоколів та проведення планування ємності каналів з метою їх подальшого розширення. В рамках проведеного дослідження встановлено можливість створення програмної реалізації, яка б працювала незалежно від мереженого стеку операційної системи в режимі фільтру, за основу доцільно використовувати комплекс NetGraph (операційна система FreeBSD)[8] або DFA (операційна система LINUX)[9]. Вказані підходи дозволять абстрагуватися від апаратного забезпечення мережевого класифікатора, оскільки процедури обробки пакетів будуть виконуватись в ядрі операційної системи.

ЛІТЕРАТУРА

1. L. Bailey, B. Gopal, A. Pagels, L. Peterson. Pathfinder: A pattern-based packet classifier., Operating Systems Design and Implementation, p. 115-123, 1994.
2. A. Begel, S. McCanne, L. Graham. BPF+: Exploiting global data-flow optimization in generalized packet filter architecture. In SIGCOMM, p. 123-134, 1999.
3. S. Chandra, P. McCann. Packet types. Workshop on Compiler Support for Systems Software (WCSSS), May 1999.
4. P. Gustafsson, K. Sagonas. Efficient manipulation of binary data using pattern matching., J. Funct. Program., p. 16-35, 2006.
5. J. Hopcroft, R. Motwani. Introduction to Automata Theory, Languages, and Computation. Addison Wesley, 2001.
6. C. Kruegel, T. Toth. Using decision trees to improve signature-based intrusion detection., Symposium on Recent Advances in Intrusion Detection (RAID), 2003.
7. T. Lakshman, D. Stiliadis. High-speed policy-based packet forwarding using efficient multi-dimensional range matching. In SIGCOMM, p. 203-214, 1998.
8. S. McCanne, Van Jacobson. The BSD packet filter: A new architecture for user-level packet capture. In USENIX Winter, p. 259-270, 1993.
9. G. Varghes. Packet classification using multidimensional cutting. SIGCOMM, 2003.
10. U. Manber. A fast algorithm for multi-pattern searching. Technical Report TR94-17, 1994.
11. Z. Chen, Y. Diao, T. Lakshman. Fast and memory-efficient regular expression matching for deep packet inspection. In Architectures for Networking and Communications Systems, p. 93-100, 2006.
12. FreeBSD Kernel Interfaces Manual [Електронний ресурс] . – Режим доступу: http://www.freebsd.org/cgi/man.cgi?bpf%284%29
13. Snort - open source network IDS/IPS [Електронний ресурс] . – Режим доступу: http://www.snort.org/

Надійшла у першій редакції 02.06.2014, в останній - 30.10.2014.