https://doi.org/10.31891/2219-9365-2025-82-58
УДК 004.056:004.852:004.75

СТЕЦЮК Юрій
Хмельницький національний університет
https://orcid.org/0000-0003-0312-2276
e-mail: yuriy.stetsuk@khmnu.edu.ua

МЕТОД КІЛЬКІСНОЇ ОЦІНКИ РІВНЯ СТІЙКОСТІ ОПЕРАЦІЙНОЇ СИСТЕМИ ДО ВИТОКУ КОНФІДЕНЦІЙНОЇ ІНФОРМАЦІЇ

Часто виникає необхідність визначити яка із складних систем є кращою, має вищий рівень при вирішені нею певного виду задач. При цьому достовірність має бути якомога вищою, бо ціна питання може бути досить високою. В нашому випадку такою задачею є забезпечення ОС високого рівня стійкості інформаційної системи, що працює під її управлінням, до не витоку конфіденційної інформації.

При проведені порівняльних аналізів будь яких складних систем загалом, чи їх окремих підсистем постає питання, відповідь на яке є зовсім нетривіальною. В таких умовах отримати однозначну відповідь на подібне запитання практично неможливо, особливо, якщо йдеться про використовувані, діючі системи.

Система або її частина, яка однозначно була б гіршою за інші системи в усіх відношеннях, просто не могла б існувати. В дійсності має місце типова багатокритеріальна задача, коли є кілька вибраних критеріїв якості, по яких мають оцінюватись порівнювані системи, а також деяка множина систем, які випереджають одна одну за одними критеріями і, зазвичай поступаються по іншим. Тому знаходження методу вирішення таких задач є перспективним напрямом дослідження і актуальною науковою задачею. В роботі запропоновано удосконалення відомого метода з метою підвищення його чутливості та застосування для кількісної оцінки порівнюваних складних програмних систем.

Напрямами подальших досліджень буде розширення класу вирішуваних ним задач.

Ключові слова: конфіденційна інформація, стійкість до витоку, зловмисне програмне забезпечення, метод порівняння.

STETSYUK Yuriy
Khmelnytskyi National University

METHOD FOR QUANTITATIVE ASSESSMENT OF THE LEVEL OF OPERATING SYSTEM RESISTANCE TO CONFIDENTIAL INFORMATION LEAKAGE

It is often necessary to determine which of the complex systems is better, has a higher level when solving a certain type of problem. At the same time, the reliability should be as high as possible, because the price of the issue can be quite high. In our case, such a task is to ensure the OS a high level of stability of the information system operating under its control, to prevent the leakage of confidential information.

When conducting comparative analyses of any complex systems in general, or their individual subsystems, a question arises, the answer to which is completely non-trivial. In such conditions, it is practically impossible to get an unambiguous answer to such a question, especially when it comes to used, operating systems.

A system or part of it that would be clearly worse than other systems in all respects simply could not exist. In reality, a typical multi-criteria problem occurs when there are several selected quality criteria by which the compared systems should be evaluated, as well as a certain set of systems that are ahead of each other by some criteria and, usually, inferior by others.

Therefore, finding a method for solving such problems is a promising direction of research and a relevant scientific problem.

The paper proposes to improve the known method in order to increase its sensitivity and use it for quantitative assessment of compared complex software systems.

The directions of further research will be to expand the class of problems solved by it.

Keywords: confidential information, leak resistance, malicious software, comparison method.

Стаття надійшла до редакції / Received 14.04.2025
Прийнята до друку / Accepted 11.05.2025

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

В процесі дослідження шляхів удосконалення та розробки методів виявлення вразливостей в складних програмних системах потрібні інструменти, які опираючись на наукові методи дозволяють виконувати порівняння досліджуваних систем по заданих інтегральних параметрах. Особливо це стосується систем безпеки серверних ОС.

Для проведення аналізу реальних ОС на предмет можливості забезпечення ними рівня стійкості до витоку, оброблюваної під їх управлінням, інформації, необхідно визначитись із множиною критеріїв, по яких буде оцінюватись кожна з ОС, що аналізується. Вони мають бути суттєвими, що забезпечить максимальну достовірність аналізу. Порівняльна важливість критеріїв залежить від призначення системи та умов її роботи. В цій роботі в якості об'єкта дослідження виступають серверні ОС, а саме їх рівень стійкості до витоків

інформації. Така ситуація виникає тоді, коли необхідно з кількох наявних фізичних систем вибрати одну, яка найкраще вирішує певний клас задач, або дослідити результат її удосконалення.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

З метою вирішення даної проблеми було виконано чимало наукових досліджень. Прогнозування дефектів програмного забезпечення основаного на правилі «точно вчасно» (JIT-SDP), як найбільш детальному методі, що використовується для прогнозування дефектів на рівні змін програмного забезпечення розглядається в [1], а також основаних на методі ройової оптимізації [2]. В [3] пропонується застосування статистичних ознак системи та її ознак з попередньо оброблених даних в поєднанні з алгоритмом оптимізації Pelican з метою вибору найкращих ознак (CMPOA) порівняння.

Огляду методів динамічного аналізу програмного забезпечення, зосереджуючись головним чином на методах, орієнтованих на виявлення рівня безпеки та надійності системного програмного забезпечення присвячені роботи [4, 5] та автоматизації методів верифікації та валідації програм [6, 7], розгляду варіацій багатокомп'ютерних систем основаних на методах адаптивної централізації [8, 9, 10].

В [11] представлено новий підхід до порівняння складних програмних систем шляхом обчислення робустної відстані Хаусдорфа між семантичними вбудовуваннями вихідного коду окремих програмних компонентів.

Адаптивність методів нечіткого тестування та тестування на основі моделей до змінних платформ та мов блокчейну з використанням порівняльного аналізу модульного складу систем розглянуто в [12].

Порушенню семантичних та структурних принципів при порівнянні модулів програмного забезпечення на основі антишаблонів присвячена робота [13]. Запровадженню підходів відновлення архітектури програмного забезпечення (SAR) для аналізу залежностей між програмними модулями та їх автоматичної кластеризації для досягнення високої модульності з наступним порівнянням методів [14]. Прогнозування вразливостей програмного забезпечення на основі порівнянь їх версій розглянуто в [15, 16].

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

При формуванні множини критеріїв слід уникати до включення до неї інтегральних параметрів – тих, що є похідними від більш простих критеріїв, тобто таких, чиє значення забезпечується сумою значень більш атомарних, вище приведених, критеріїв. Такими критеріями при оцінці ОС є надійність, ефективність та інші.

Таким чином, першим кроком дослідження ОС є побудова скінченої множини критеріїв оцінки (3):

$$M_k = (k_1, k_2, \dots, k_n), \tag{1}$$

де n – число критеріїв; $k_1, k_2, \dots, k_n$ - критерії оцінки деякої фізичної чи змодельованої ОС.

Не зважаючи на те, що всі критерії є суттєвими, все ж вони по різному впливають на результат дослідження. Тому, для врахування цієї обставини спів ставимо кожен критерій з його ваговим коефіцієнтом.

При цьому, відмітимо, що сума всіх вагових коефіцієнтів критеріїв її оцінки дорівнює одиниці, що визначається формулою (2).

$$\sum_{i=1}^{n} k_i = 1, \tag{2}$$

де n – число застосованих критеріїв оцінки ОС.по деякому інтегральному параметру; k_i - ваговий коефіцієнт і-го критерія оцінки.

Слід зазначити, що незважаючи на ту обставину, що для всіх ОС, для яких необхідно визначити рівень стійкості до не витоку інформації ми використовуємо одну і туж множину критеріїв M_k, при цьому кожна фізично існуюча ОС має власний розподіл вагових коефіцієнтів критеріїв її оцінки, що до рівня потенціалу стійкості до витоків інформації. Це пов'язано з тим, що в кожній окремій існуючій ОС використані для її оцінки критерії в силу різниці в архітектурі, різниці в функціональній взаємодії між модулями ОС, різниці в алгоритмах вирішення задач і т.і.

При виконані оцінок абстрактних ОС, часто користуються формальними підходами в оцінці вагових коефіцієнтів критеріїв. Так, величину значень вагових коефіцієнтів приймають однаковою, та визначають як величину, що зворотна кількості критеріїв $1/n$ у множині критеріїв M_k, що зручно для подальших маніпуляцій з ними.

Але, коли ми маємо справу з реальними, фізично існуючими системами такий підхід неприпустимий, тому, що гарантовано призведе до значного викривлення результатів аналізу і хибних рішень.

Тому наступним кроком побудови методу кількісної оцінки рівня стійкості ОС до не витоку інформації буде визначення способу задання значень вагових коефіцієнтів.

Визначення вагових коефіцієнтів критеріїв для оцінки рівня стійкості ОС до витоку інформації. В реальних умовах, дуже часто виникає необхідність обрати кращий варіант з кількох можливих. Наприклад необхідно вибрати кращу ОС по показнику стійкості до витоку оброблюваної в комп'ютерній системі (КС) інформації. Причому сама по собі кожна ОС є складною системою з власними архітектурою, алгоритмами

керування та між модульної взаємодії. При чому в силу різних об'єктивних та суб'єктивних факторів складові різних ОС проявляють себе по різному. Одна й та ж система може прекрасно показати себе, наприклад в управлінні файловою системою і, в той же час бути не в змозі забезпечити режим реального часу при реакції на якусь подію в системі чи зовні.

Тому при побудові КС з заданими параметрами приходиться виконувати вибір ОС, яка забезпечить найбільш високі параметри експлуатації, що відповідають заданим. Відповідь на питання, яка з можливих кандидаток краща чи гірша практично неможливо отримати без побудови деякого узагальнюючого, інтегрального показника, яким в нашому випадку, буде рівень стійкості ОС до витоку конфіденційної інформації, що обробляється в комп'ютерній системі під її управлінням. Після оцінки його рівня для кожної ОС – претендентки буде легко знайти кращу. Но це завершальний крок, а йому передує ще кілька кроків.

Першим кроком є побудова набору критеріїв, що характеризують деяку, наперед задану, цільову сторону роботи ОС. Він може бути реалізований багатьма доступними способами в залежності від основного завдання. Результатом буде множина M_k (формула (1)).

Оскільки критерії, які характеризують деяку сторону роботи ОС, мають різний вплив на забезпечення необхідної функціональності, то їм у відповідність треба поставити вагові коефіцієнти. Таким чином можна врахувати вклад кожного критерія в підтримання такого інтегрального параметра, як забезпечення стійкості ОС до витоків конфіденційної інформації. Це буде другий крок.

Для визначення вагових коефіцієнтів можна скористатись формальними методами, але в цьому випадку отриманий результат буде наперед менш точним, в силу того, що обчислення їх значень скоріше ніяк не буде враховувати специфіку ОС, її архітектуру, алгоритми управління, тощо. ОС, як система, або навіть деяка її підсистема є досить складними об'єктами, щоб можна було надіятись отримати множину вагових коефіцієнтів, яка буде знаходитись поблизу варіанта, що може існувати в природі речей.

Тут кращий результат, на противагу формальними методам, забезпечить експертний підхід. Число і кваліфікаційний рівень експертів залежить від важливості вирішуваної задачі. Зрозуміло, що чим вища кваліфікація експертів і більше їх число, результат оцінки вагових коефіцієнтів і результат загалом буде достовірніший.

Хочеться звернути увагу, що класичний підхід при вирішені подібних задач передбачає визначення вагових коефіцієнтів для кожного критерія множини M_k. Але коли ми в якості об'єкта аналізу маємо такий об'єкт як ОС, то цей метод [17, 18] не забезпечить максимальну достовірність результату, через те, що упускаються нетривіальні, неповторювані особливості найскладніших програмних систем, таких як ОС.

Визначення значень вагових коефіцієнтів по кожному критерію для кожної конкретної ОС дозволить покращити отримання максимально достовірного результату. Щоб скористатись таким підходом, необхідно удосконалити класичний метод [17, 18]. Покращення полягає в заміні множини значень вагових коефіцієнтів, однакових для всіх об'єктів аналізу (формула (1)) на матрицю значень вагових коефіцієнтів (формула (4)).

$$M_{vk} = \begin{bmatrix} vk_{1.1} & vk_{1.2} & ... & vk_{1.j} \\ vk_{2.1} & vk_{2.2} & ... & vk_{2.j} \\ ... & ... & ... & ... \\ vk_{i.1} & vk_{i.2} & ... & vk_{i.j} \end{bmatrix}, \quad (4)$$

де: $vk_{i.j}$ - ваговий коефіцієнт j-го критерія i-ї ОС; i – кількість ОС, що аналізуються; j – кількість критеріїв, що використовуються в аналізі.

При цьому вагові коефіцієнти одного рядка матриці M_{vk}, мають задовольняти таким умовам:

$$\sum_{j=1}^{n} vk_j = 1, \quad \text{та } vk_{i.j} \geq 0 \quad (5)$$

де: $vk_{i.j}$ - ваговий коефіцієнт j-го критерія i-ї ОС; j – кількість критеріїв, що використовуються в аналізі деякої ОС; i – кількість ОС, що аналізуються.

В результаті, кожна ОС буде співставлятись зі своєю власною множиною вагових коефіцієнтів, яка буде представлена відповідним рядком елементів матриці вагових коефіцієнтів. Якщо всі рядки матриці вагових коефіцієнтів (формула (4)) отримають однакові значення, то ми, фактично, повернемось до класичного підхіду (формула (1)), де значення вагових коефіцієнтів залежить тільки від критерія, з яким він зв'язаний.

Кількісна оцінка стійкості ОС до витоку конфіденційної інформації. Маючи на меті отримання кількісних оцінок стійкості ОС перейдемо до наступного кроку методу, який передбачає побудову матриці значень критеріїв, кожен елемент якої буде визначено в результаті експерної оцінки кожного критерію для кожної ОС в відносних балах. Матриця значень критеріїв M_{zk} в загальному вигляді буде виглядіти так:

$$M_{zk} = \begin{bmatrix} zk_{1.1} & zk_{1.2} & ... & zk_{1.j} \\ zk_{2.1} & zk_{2.2} & ... & zk_{2.j} \\ ... & ... & ... & ... \\ zk_{i.1} & zk_{i.2} & ... & zk_{i.j} \end{bmatrix}, \quad (6)$$

де: $zk_{i.j}$ - значення j -го критерія і -ї ОС;

Матриці (формула (4)) та (формула (6)) містять всю необхідну вихідну інформацію, яка потрібна для виконання кількісної порівняльної оцінки заданої множини складних об'єктів, якими тут є ОС.

Наступний крок передбачає нормування значень критеріїв оцінки, з метою приведення їх у взаємну відповідність до масштабу вимірювання їх значень, що зробить їх порівнювальними і абстрагованими від одиниць вимірювання. Для нормування скористаємось формулою:

$$F(M_{zk}) = \frac{f_{zk_j}(x) - f_{zk_j}^{min}}{f_{zk_j}^{min}} \quad (7)$$

де: $f_{zk_j}^{min}$ - мінімальне значення критерію zk_j множини його значень, отриманих в ході експертної оцінки; $f_{zk_j}(x)$ – фактичне значення критерію zk_j множини всіх його значень.

Застосувавши формулу (7) до кожного елемента матриці (формула(6)) отримаємо матрицю нормованих значень критеріїв M_{nzk}:

$$M_{nzk} = \begin{bmatrix} nzk_{1.1} & nzk_{1.2} & ... & nzk_{1.j} \\ nzk_{2.1} & nzk_{2.2} & ... & nzk_{2.j} \\ ... & ... & ... & ... \\ nzk_{i.1} & nzk_{i.2} & ... & nzk_{i.j} \end{bmatrix}, \quad (8)$$

де: $nzk_{i.j}$ - нормоване значення j-го критерія і-ї ОС;

Наступним кроком є розрахунок рівня стійкості кожної ОС $R_{S_{OSi}}$ для чого скористаємось формулою (9):

$$R_{S_{OSi}} = \sum_{j=1}^{n} vk_{i.j} \cdot nzk_{i.j} \quad (9)$$

В результаті виконання розрахунків над даними матриць M_{vk} та M_{nzk} за формулою (9) отримаємо матрицю рівнів стійкості ОС:

$$M_{RS} = \begin{bmatrix} R_{S_{OS1}} \\ R_{S_{OS2}} \\ ... \\ R_{S_{OSi}} \end{bmatrix}, \quad (10)$$

де: $R_{S_{OSi}}$ - рівень стійкості і-ї ОС до витоку інформації у відносних одиницях.

Заключним кроком цього методу є знаходження елемента $f_{R_{S_{OSi}}}^{max}$ з максимальним значенням у стовпчику матриці M_{RS}. Воно буде відповідати ОС, в якої стійкість до витоку конфіденційної інформації, що обробляється в інформаційній системі, під її керівництвом, найвища.

Враховуючи, що на цільову функцію $F_{nzk_i}(x)$ впливає лише окремий критерій оптимальності, якому в деякій точці j-го критерію відповідає найменше значення функціоналу $f_{zk_j}^{min}$, що означає проведення розрахунку по найгіршій ситуації. При цьому по значенню $F_{nzk_i}(x)$ можна визначити гарантовану нижню оцінку для всіх функціоналів $f_{zk_j}^{min}$. Це означає, що отриманий кількісний результат стійкості ОС до не витоку інформації, для самої критичної ситуації, буде знаходитись на границі області можливих рішень.

Вибір ОС для експериментальних розрахунків. З метою підтвердження робото здатності запропонованого методу необхідно виконати експериментальні розрахунки рівня стійкості вибраних ОС. З метою перевірки точності методу розрахунки виконаємо у двох варіантах – з індивідуальними ваговими коефіцієнтами для кожної ОС та з загальними, що співставленні тільки із оціночними критеріями.

В якості ОС, що будуть досліджуватись по параметру стійкості до витоку інформації були вибрані реально працюючі, достатньо відомі, ОС: I_1 - **CentOS 7;** I_2 - FreeBSD 10.0; I_3 -Windows NT 10.0.

Вибір саме цих систем зумовлений тим, що всі вони є серверними ОС, появились на ринку приблизно в один час, працюють на одних і тих же апаратних платформах з використанням самих сучасних 64-розрядних процесорів. Не зважаючи на різні архітектури, що в них реалізовані, вони створені для вирішення одного і того ж класу задач, а тому є порівнюваними. Пройшов вже деякий час з моменту початку їх експлуатації, що

дало змогу накопичити достатньо достовірної інформації про її результати, яка дозволить спів ставити результати розрахунків не тільки між собою, но і пересвідчитись, в тому, що вони не суперечать інформації, отриманій в ході реальної роботи ОС.

Підготовка даних для експериментів. В процесі аналізу, були відібрані наступні критерії, які найбільше впливають на достовірність результату в частині стійкості ОС до витоків інформації: J_1 - тип ядра ОС; J_2- наявність вбудованих засобів протидії ЗПЗ; J_3- схема управління пам'яттю системи; J_4- підтримувані типи файлових систем; J_5- підтримка захищеного режиму процесора; J_6- відкритість вихідного коду; : J_7- масштабування системи.

Тепер можна приступити до наступного кроку, який полягає у визначенні значень вагових коефіцієнтів критеріїв оцінки. Результати їх роботи представлені в таблицях 1 – 3.

Таблиця 1.

Вагові коефіцієнти критеріїв оцінки стійкості ОС $I_1 - I_3$ до витоків інформації отримані методом експертної оцінки для удосконаленого методу

ОС	Вагові коефіцієнти критеріїв						
	vkJ_1	vkJ_2	vkJ_3	vkJ_4	vkJ_5	vkJ_6	vkJ_7
I_1	0,15	0,25	0,08	0,2	0,14	0,1	0,08
I_2	0,15	0,25	0,08	0,2	0,14	0,1	0,08
I_3	0,14	0,13	0,12	0,16	0,09	0,05	0,14

Оскільки ОС I_1 та I_2 належать одного Linux-сіметва, що зумовило багато архітектурних спільностей, то експерти прийняли вагові коефіцієнти їх критеріїв рівними (таблиця 1).

Таблиця 2.

Вагові коефіцієнти критеріїв оцінки $J_1 - J_7$ для класичного методу

Вагові коефіцієнти критеріїв						
vkJ_1	vkJ_2	vkJ_3	vkJ_4	vkJ_5	vkJ_6	vkJ_7
0,15	0,25	0,09	0,2	0,13	0,08	0,1

Таблиця 3.

Значення експертної оцінки рівня критеріїв $J_1 - J_7$ оцінки для ОС $I_1 - I_3$ відповідно

ОС	Критерії оцінки, бали						
	J_1	J_2	J_3	J_4	J_5	J_6	J_7
I_1	67	88	71	62	60	18	75
I_2	75	81	80	64	58	24	70
I_3	81	86	75	52	75	17	81

Дані таблиць 1, 2 та 3 були підготовлені на підставі зведених даних по вибраних для розрахунку ОС по результатах проведених експериментів. Скористаємось ними для побудови матриці вагових критеріїв оцінки $ОСМ_{vk}$, відповідно до формули (4):

$$M_{vk} = \begin{bmatrix} vk_{1.1} & \cdots & vk_{1.7} \\ \vdots & \ddots & \vdots \\ vk_{3.1} & \cdots & vk_{3.7} \end{bmatrix} = \begin{bmatrix} 0{,}15 & 0{,}25 & 0{,}08 & 0{,}2 & 0{,}14 & 0{,}1 & 0{,}08 \\ 0{,}15 & 0{,}25 & 0{,}08 & 0{,}2 & 0{,}14 & 0{,}1 & 0{,}08 \\ 0{,}14 & 0{,}13 & 0{,}12 & 0{,}16 & 0{,}09 & 0{,}05 & 0{,}14 \end{bmatrix}.$$

Також побудуємо матрицю значень критеріїв оцінки ОС відповідно до формули (6):

$$M_{zk} = \begin{bmatrix} zk_{1.1} & \cdots & zk_{1.7} \\ \vdots & \ddots & \vdots \\ zk_{3.1} & \cdots & zk_{3.7} \end{bmatrix} = \begin{bmatrix} 67 & 88 & 71 & 62 & 60 & 18 & 75 \\ 75 & 81 & 80 & 64 & 58 & 24 & 70 \\ 81 & 86 & 75 & 52 & 75 & 17 & 81 \end{bmatrix}.$$

По даних матриці M_{zk} побудуємо множину мінімальних значень критеріїв оцінки ОС:

$$M_{zk}^{min} = \{zk_1^{min}, zk_2^{min}, \dots, zk_7^{min}\} = \{67, 81, 71, 52, 52, 58, 17, 70\}.$$

Вона буде використана для нормування значень критеріїв. Оскільки вони задані в балах із застосуванням різних шкал, то для подальшого їх використання потрібно їх привести до одного масштабу. Виконаємо цей крок за допомогою формули (формула (7)), де $f_{zk_j}^{min}$ - відповідні значення з множини M_{zk}^{min}. Її потрібно застосувати до кожного елемента матриціM_{zk}. В результаті обчислень отримаємо матрицю нормованих значень критеріїв оцінки $ОСМ_{nzk}$:

$$M_{nzk} = \begin{bmatrix} nzk_{1.1} & \cdots & nzk_{1.7} \\ \vdots & \ddots & \vdots \\ nzk_{3.1} & \cdots & nzk_{3.7} \end{bmatrix} = \begin{bmatrix} 0 & 0{,}09 & 0 & 0{,}19 & 0{,}03 & 0{,}13 & 0{,}07 \\ 0{,}12 & 0 & 0{,}13 & 0{,}23 & 0 & 0{,}25 & 0 \\ 0{,}21 & 0{,}06 & 0{,}06 & 0 & 0{,}12 & 0 & 0{,}16 \end{bmatrix}$$

Наступним кроком є обчислення цільових функцій по кожній ОС, або іншими словами рівня стійкості ОС. Для цього скористаємось формулою (9). Її треба застосувати до кожного рядка матриці M_{nzk}. В результаті було отримано матрицю рівнів стійкості ОС:

$$M_{RS} = \begin{bmatrix} Rs_{FizOS1} \\ Rs_{FizOS2} \\ Rs_{FizOS3} \end{bmatrix} = \begin{bmatrix} 0{,}453 \\ 0{,}0994 \\ 0{,}0776 \end{bmatrix}$$

Останнім кроком є знаходження максимального значення у стовпці матриці M_{RS}. Ним є рядок $Rs_{FizOS2} = 0б994$, що відповідає ОС FreeBSD 10.0. Як підсумок, можна зробити висновок, що отриманий результат прекрасно корелюється з даними накопиченими під час їх експлуатації.

Тепер виконаємо ще один розрахунок, із застосуванням канонічного підходу. Дані значень вагових коефіцієнтів тепер візьмемо з таблиці 2, що є спільними для всіх задіяних в експерименті ОС.

Оскільки всі кроки, що передують побудові матриці рівнів стійкості ОС M_{RS} залишаються тими ж, то ми можемо зразу приступити до виконання передостаннього кроку. Для його виконання також скористаємось формулою (9), але дещо модифікованою:

$$R_{S_{OSi}} = \sum_{j=1}^{7} vk_j \cdot nzk_{i.j} \qquad (11)$$

де: vk_j- значення j-го вагового коефіцієнта, не зв'язаного з ОС, j – число критеріїв оцінки рівня стійкості ОС.

Її також треба застосувати до кожного рядка матриці M_{nzk}. В результаті отримаємо матрицю:

$$M_{RS} = \begin{bmatrix} Rs_{FizOS1} \\ Rs_{FizOS2} \\ Rs_{FizOS3} \end{bmatrix} = \begin{bmatrix} 0{,}081 \\ 0{,}0983 \\ 0{,}0859 \end{bmatrix}.$$

З оцінки значень матриці M_{RS}, видно, що рядок $Rs_{FizOS2} = 0б983$ вказує, що ОС FreeBSD 10.0 і в цьому варіанті розрахунку має вищий потенціал стійкості до витоку інформації, що не суперечить попередньому результату.

Якщо ж порівняти кількісні показники матриць обох варіантів розрахунків, то видно, що перший варіант представляє одні і ті ж дані в більш ширшому числовому діапазоні, що говорить про його більшу чутливість, при оцінювані складних, разом з тим дуже похожих об'єктів. Ця властивість методу дозволить його застосовувати для дослідження близьких по архітектурі ОС.

Результати експериментальних досліджень підтвердили ефективність розробленого методу оцінки стійкості ОС до витоку інформації, оброблюваної в КС під її управлінням.

ВИСНОВКИ З ДАНОГО ДОСЛІДЖЕННЯ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ РОЗВІДОК У ДАНОМУ НАПРЯМІ

Особливістю основних кроків розробленого методу є використання вагових коефіцієнтів критеріїв оцінки ОС, які знаходяться в залежності не тільки від критеріїв, як це прийнято, але додатково, залежать ще і від самого об'єкта оцінки. Використання ускладнених залежностей, дозволило підвищити чутливість методу оцінки, що особливо має значення при порівнянні досить похожих об'єктів, які при традиційному підході можуть виглядати однаковими.

Особливістю запропонованого методу є кількісна оцінка рівня стійкості ОС до витоку інформації, що дозволяє бачити, як зміна значень тих чи інших критеріїв впливає на рівень стійкості, що розширює застосування методу при наукових дослідженнях ОС.

Розроблений метод може застосовуватись як до абстрактних моделей ОС в наукових дослідженнях, так і для оцінки реальних, фізично існуючих ОС. В цьому полягає практичне значення розробленого методу.

Проведені експериментальні дослідження із залученням сучасних ОС. Виконані оціночні розрахунки підтверджують високу ефективність розробленого методу оцінки стійкості ОС до витоку конфіденційної інформації, яка обробляється в КС під її управлінням, оскільки кількісне представлення рівня досліджуваного параметра дає можливість більш точного судження про нього.

Виконані розрахунки рівня стійкості серверних ОС підтвердили підвищену чутливість запропонованого методу, в порівняні з стандартним підходом.

Метод може бути використаний не тільки для оцінки стійкості ОС до витоків інформації. Його підхід може бути розширений до використання при оцінці складних програмних об'єктів по інших інтегральних параметрах, що стане напрямами подальших досліджень застосування методу.

Література

1. Wu, Q., Wang, X. & Wei, D. (2025). Just-in-time software defect prediction method for non-stationary and imbalanced data streams. Software Qual, J 33, 14.

2. Malhotra, R., Khan, K. (2024). A novel software defect prediction model using two-phase grey wolf optimisation for feature selection. Cluster Comput 27, 12185–12207.
3. Shyamala, C., Mohana, S. & Ambika, M. (2024). Hybrid deep architecture for software defect prediction with improved feature set. Multimed Tools Appl 83, 76551-76586.
4. Haque, M., Ahmad, N. (2025). logistic software reliability model with Loglog fault detection rate. Iran, J Comput Sci 8, 1–7.
5. Kuliamin, V. A (2024). Survey of Software Dynamic Analysis Methods. Program Comput Soft 50, 90–114.
6. Taromirad, M., Runeson, P. (2025). Assertions in software testing: survey, landscape, and trends. Int J Softw Tools Technol Transfer, 1-20
7. Kashtalian, A., Lysenko, S., Kysil, T., Sachenko, A., Savenko, O., & Savenko, B. (2025). Method and Rules for Determining the Next Centralization Option in Multicomputer System Architecture. International Journal of Computing, 24(1), 35-51.
8. Stetsyuk, Y., Stetsyuk, M., Kyrylo, V., Paiuk, V., Kvassay, M. (2024). A model of a centralized security system, as an information technology for the synthesis of an OS architecture protected against the leakage of confidential information. CEUR-WS. 3899, 224-233.
9. Kashtalian, A., Lysenko, S., Sachenko, A., Savenko, B., Savenko, O., & Nicheporuk, A. (2025). Evaluation criteria of centralization options in the architecture of multicomputer systems with traps and baits. Radioelectronic and Computer Systems, 2025(1), 264-297.
10. Stetsyuk, Y., Stetsyuk, M., Savenko B., Kwiecien, A., Kopania L. (2024). Method of random dynamic control transfer between network nodes for a partially centralized OS security system / CEUR-WS. 3963 264-283.
11. Karakatic, S., Milosevic, A. & Hericko, T. (2022). Software system comparison with semantic source code embeddings. Empir Software Eng 27, 70.
12. Elakaş, A., Sözer, H., Şafak, I. (2024). A systematic mapping on software testing for blockchains. Cluster Comput 27, 7111–7126.
13. Kalhor, S., Keyvanpour, M.R. & Salajegheh, A. (2024). A systematic review of refactoring opportunities by software antipattern detection. Autom Softw Eng, 31, 42.
14. Elyasi, M., Simitcioglu, M.E., Saydemir, A. (2023). Genetic algorithms and heuristics hybridized for software architecture recovery. Autom Softw Eng 30, 19.
15. Malhotra, R. (2025). Vidushi Hybrid feature selection module for improving performance of software vulnerability severity prediction model on textual dataset. Computing 107, 66.
16. Kashtalian, A., Lysenko, S., Savenko, O., Nicheporuk, A., Sochor, T., & Avsiyevych, V. (2024). Multi-computer malware detection systems with metamorphic functionality. Radioelectronic and Computer Systems, 2024(1), 152-175
17. Akhtar, Z.B. (2024). Securing Operating Systems (OS): A ComprehensiveApproach to Security with Best Practices and Techniques. International Journal of Advanced Network, Monitoring and Controls 2024, 09(01), 100-111.
18. Pathak, P., Nadeem, M., & Ansar, S. (2024). Security assessment of operating system by using decision making algorithms. Int. j. inf. tecnol. 2024; 8, 17.

References

1. Wu, Q., Wang, X. & Wei, D. (2025). Just-in-time software defect prediction method for non-stationary and imbalanced data streams. Software Qual, J 33, 14.
2. Malhotra, R., Khan, K. (2024). A novel software defect prediction model using two-phase grey wolf optimisation for feature selection. Cluster Comput 27, 12185–12207.
3. Shyamala, C., Mohana, S. & Ambika, M. (2024). Hybrid deep architecture for software defect prediction with improved feature set. Multimed Tools Appl 83, 76551-76586.
4. Haque, M., Ahmad, N. (2025). logistic software reliability model with Loglog fault detection rate. Iran, J Comput Sci 8, 1–7.
5. Kuliamin, V. A (2024). Survey of Software Dynamic Analysis Methods. Program Comput Soft 50, 90–114.
6. Taromirad, M., Runeson, P. (2025). Assertions in software testing: survey, landscape, and trends. Int J Softw Tools Technol Transfer, 1-20
7. Kashtalian, A., Lysenko, S., Kysil, T., Sachenko, A., Savenko, O., & Savenko, B. (2025). Method and Rules for Determining the Next Centralization Option in Multicomputer System Architecture. International Journal of Computing, 24(1), 35-51.
8. Stetsyuk, Y., Stetsyuk, M., Kyrylo, V., Paiuk, V., Kvassay, M. (2024). A model of a centralized security system, as an information technology for the synthesis of an OS architecture protected against the leakage of confidential information. CEUR-WS 3899, 224-233.
9. Kashtalian, A., Lysenko, S., Sachenko, A., Savenko, B., Savenko, O., & Nicheporuk, A. (2025). Evaluation criteria of centralization options in the architecture of multicomputer systems with traps and baits. Radioelectronic and Computer Systems, 2025(1), 264-297.
10. Stetsyuk, Y., Stetsyuk, M., Savenko B., Kwiecien, A., Kopania L. (2024). Method of random dynamic control transfer between network nodes for a partially centralized OS security system / CEUR-WS 3963 264-283.
11. Karakatic, S., Milosevič, A. & Hericko, T. (2022). Software system comparison with semantic source code embeddings. Empir Software Eng 27, 70.

12. Elakaş, A., Sözer, H., Şafak, I. (2024). A systematic mapping on software testing for blockchains. Cluster Comput 27, 7111–7126.
13. Kalhor, S., Keyvanpour, M.R. & Salajegheh, A. (2024). A systematic review of refactoring opportunities by software antipattern detection. Autom Softw Eng, 31, 42.
14. Elyasi, M., Simitcioglu, M.E., Saydemir, A. (2023). Genetic algorithms and heuristics hybridized for software architecture recovery. Autom Softw Eng 30, 19.
15. Malhotra, R. (2025). Vidushi Hybrid feature selection module for improving performance of software vulnerability severity prediction model on textual dataset. Computing 107, 66.
16. Kashtalian, A., Lysenko, S., Savenko, O., Nicheporuk, A., Sochor, T., & Avsiyevych, V. (2024). Multi-computer malware detection systems with metamorphic functionality. Radioelectronic and Computer Systems, 2024(1), 152-175
17. Akhtar, Z.B. (2024). Securing Operating Systems (OS): A ComprehensiveApproach to Security with Best Practices and Techniques. International Journal of Advanced Network, Monitoring and Controls 2024, 09(01), 100-111.
18. Pathak, P., Nadeem, M., & Ansar, S. (2024). Security assessment of operating system by using decision making algorithms. Int. j. inf. tecnol. 2024; 8, 17.