КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 1 (33), 2026
ISSN 2663 – 4023
DOI 10.28925/2663-4023.2026.33.1254
УДК 004.62

Четвертуха Назарій Романович
аспірант кафедри захисту інформації
Національний Університет «Львівська Політехніка», Львів, Україна
ORCID: 0009-0001-0944-2599
nazarii.chetvertukha.asp.2025@lpnu.ua

Отенко Віктор Іванович
к.т.н., доцент кафедри захисту інформації
Національний Університет «Львівська Політехніка», Львів, Україна
ORCID: 0000-0003-4781-7766
viktor.i.otenko@lpnu.ua

РОЗРОБЛЕННЯ БАГАТОВИМІРНОЇ ФАСЕТНОЇ ТАКСОНОМІЇ ТЕХНІК ОБФУСКАЦІЇ ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Анотація. У статті розглянуто проблему систематизації обфускаційних технік шкідливого програмного забезпечення, що ускладнюється структурними обмеженнями наявних класифікаційних схем у галузі та відсутністю інтегрованого підходу до опису класичних обфускаційних технік та технік протидії аналізу. У роботі поняття обфускації подається у широкому сенсі, що охоплює як класичні обфускаційні техніки, так і техніки протидії аналізу, об'єднані спільним призначенням приховування шкідливої поведінки програми від аналітичних інструментів. Проаналізовано класичні та сучасні таксономії обфускації, для яких виявлено спільні системні обмеження, що включають неможливість опису багатоаспектних технік за кількома незалежними властивостями, відсутність інтегрованого підходу до класичних обфускаційних технік та технік протидії аналізу, а також відсутність використання систематичної методології при побудові наявних класифікаційних схем. Метою роботи є розроблення багатовимірної фасетної таксономії технік обфускації ШПЗ, що усуває зазначені обмеження. Для досягнення поставленої мети використано метод побудови таксономій Nickerson, Varshney та Muntermann із чергуванням емпірико-концептуальних та концептуально-емпіричних ітерацій. У результаті сформовано фасетну таксономію з п'ятьма незалежними вимірами, а саме рівнем спостережуваності, масштабом впливу, механізмом перетворення, фазою прояву ефекту та поведінковою інваріантністю. Запропоновану таксономію застосовано для опису 16 представницьких обфускаційних технік сучасного ШПЗ. Окремо запропоновано вимір поведінкової інваріантності, який операційно відрізняє класичні обфускаційні техніки від технік протидії аналізу та забезпечує об'єднання двох напрямів технік приховування шкідливої поведінки у спільну класифікаційну рамку зі збереженням структурного розрізнення між ними. Запропонована таксономія створює методологічну основу для систематизації обфускаційних технік та проектування систем виявлення обфускованого ШПЗ.

Ключові слова: обфускація шкідливого програмного забезпечення, фасетна таксономія, протидія аналізу, класифікаційна схема, поведінкова інваріантність, виявлення шкідливого програмного забезпечення, методологія Nickerson, багатовимірна класифікація, шкідливе програмне забезпечення.

ВСТУП

Упродовж останніх років цифрова трансформація зробила інформаційні технології центром операційної діяльності державних структур, бізнесу, та об'єктів критичної інфраструктури. Зростання складності інформаційних систем, розширення цифрової взаємодії та поява нових платформ виконання коду спричиняють постійне розширення поверхні потенційних атак зловмисниками. У такому середовищі шкідливе програмне забезпечення (ШПЗ) зберігає статус домінуючого вектора кіберзагроз, оскільки будь-який елемент цифрової інфраструктури – від персональних робочих комп'ютерів до розподілених хмарних сервісів являє собою потенційний об'єкт атаки.

Сучасні зразки ШПЗ принципово відрізняються від аналогів попередніх десятиліть як своєю різноманітністю, так і рівнем технічної складності. При створенні ШПЗ, зловмисники активно застосовують поліморфні та метаморфічні перетворення, упакування виконуваного коду, шифрування

рядків та інструкцій, додавання надлишкового коду, віртуалізацію логіки та комбінації цих прийомів. Усі ці техніки об’єднані поняттям обфускації – перетворенням програмного коду, що ускладнює його аналіз, зберігаючи при цьому функціональність програми.

Постановка проблеми. Обфускація стала невід'ємним елементом побудови ШПЗ. За даними AV-TEST Institute, щоденно реєструється понад 450 000 нових артефактів шкідливих програм та потенційно небажаних застосунків, а сумарна база станом на 2025 рік перевищує 1,56 млрд зразків [1]. При цьому пакувальники й обфускація коду залишаються провідним інструментарієм авторів ШПЗ для уникнення виявлення [15], що зумовлює стрімкий приріст унікальних артефактів при незмінній функціональній основі та підкреслює потребу в систематичному розумінні цих перетворень [3]. Таким чином, одним із основних механізмів породження різноманіття зразків ШПЗ є саме обфускація, а не зміна їхньої функціональності та внутрішньої логіки.

У відповідь на зростання кількості та різноманітності загроз, класичний сигнатурний антивірусний захист поступово доповнювався механізмами ізольованого виконання підозрілих об'єктів у пісочницях, алгоритмами машинного навчання, а також рішеннями класу Endpoint Detection and Response (EDR). Попри суттєвий прогрес у розробці цих технологій, обфускація продовжує залишатися провідним чинником зниження ефективності систем виявлення. Сигнатурні методи втрачають чутливість до зразків, які при кожному запуску породжують нову бінарну форму [3, 4]; статичні класифікатори на основі машинного навчання виявляють обмежену здатність розпізнавання ШПЗ, обфускованого прийомами відмінними від тих, які застосовані до навчальної вибірки [11]; системи динамічного аналізу набувають вразливості до технік протидії аналізу, які модифікують поведінку програми при виявленні аналітичного середовища [14, 15]. Спільною рисою цих обмежень є залежність результативності засобу виявлення ШПЗ від конкретного типу обфускаційної техніки, використаної у зразку [4].

Саме ця залежність робить систематичну класифікацію обфускаційних технік необхідною передумовою проектування ефективних систем виявлення обфускованого ШПЗ. Проте наявні класифікаційні схеми обфускаційних технік мають три спільні системні обмеження.

По-перше, наявні класифікаційні схеми не дозволяють описати техніку одночасно за кількома її властивостями. Багато обфускаційних технік є багатоаспектними, тобто одночасно проявляються на кількох рівнях програми, наприклад на рівні метаданих файлу, структури коду та середовища виконання. У наявних класифікаціях такі техніки отримують неповний опис. Одновимірні таксономії змушують обрати один з цих аспектів як основу класифікації, а багатошарові схеми, хоча пропонують концептуальну незалежність рівнів, не надають формального правила розміщення техніки одразу в кількох рівнях.

По-друге, у літературі сформувалися два паралельні напрями класифікації технік, що приховують шкідливу поведінку програми від аналітичних інструментів. Перший напрям охоплює техніки, що належать до класичного визначення обфускації, – перетворення програмного коду зі збереженням його функціональної семантики (упакування, поліморфне шифрування, зміна потоку керування). Другий напрям охоплює техніки протидії аналізу – протидію відлагодженню (anti-debugging), протидію віртуалізації (anti-virtualization), прив'язку до середовища виконання (environmental keying) та інші. Інтегрованої класифікаційної схеми, яка охоплювала б обидва напрями зі збереженням структурного розрізнення між ними, у галузі не запропоновано.

По-третє, у домені обфускації ШПЗ наявні класифікаційні схеми побудовані без застосування методології побудови таксономій. Унаслідок цього повнота охоплення обфускаційних технік та несуперечливість класифікаційних рубрик у наявних схемах залишаються не підтвердженими емпірично.

Для подолання цих обмежень пропонується багатовимірна фасетна таксономія, що класифікує техніку одночасно за кількома незалежними характеристиками, охоплює обидва напрями технік приховування у спільній рамці зі структурним виокремленням між ними, та є побудована за верифікованою методологією.

Аналіз останніх досліджень і публікацій. Обфускаційні техніки ШПЗ систематизовано у низці класифікаційних схем, які відрізняються за принципом побудови та сферою охоплення. У праці Collberg, Thomborson та Low [5] запропоновано розрізняти техніки за чотирма видами обфускаційних трансформацій – на рівні розмітки, даних, потоку керування та превентивними трансформаціями, спрямованими на ускладнення роботи автоматичних деобфускаторів. Також, автори подають чотири критерії оцінювання якості: потенціал, стійкість, вартість і непомітність. Це дослідження заклало основу для побудови подальших таксономій, проте з часом стало очевидним, що зазначена таксономія має кілька суттєвих обмежень. По-перше, вона описує тільки трансформації, які застосовуються на етапі компіляції, тоді як техніки динамічної генерації коду, самомодифікації та віртуалізації коду залишаються поза класифікацією. По-друге, таксономія орієнтована переважно на вихідний код та проміжні представлення, що робить її менш застосовною до коду низького рівня. По-третє, криптографічні методи

у ній розглядаються лише поверхово, хоча сучасне ШПЗ активно використовує їх як основний засіб обфускації.

Більш сучасний напрям представлений роботою Galloro, Polino, Carminati, Continella та Zanero [4], яка є одним із найбільш комплексних досліджень технік протидії аналізу для ШПЗ на платформі Windows. Автори систематизували 92 техніки, які спрямовані на виявлення та нейтралізацію інструментованих середовищ аналізу (відлагоджувачів, віртуальних машин, sandbox-систем), та структурували їх за тематичними групами відповідно до типу цільового артефакту перевірки. Особливістю роботи, що вирізняє її серед попередніх систематизаційних оглядів, є наявність емпіричної компоненти. У її межах виконано експеримент на корпусі з 45 375 зразків Windows-ШПЗ, результати якого зіставлено з контрольною вибіркою легітимних Windows-програм. На основі такого зіставлення сформульовано характеристики поширеності та еволюції технік протидії аналізу впродовж досліджуваного періоду, що дозволило зафіксувати динамічні характеристики галузевого тренду, які відсутні в оглядових роботах. Водночас зазначена робота обмежена доменом технік протидії аналізу і не охоплює трансформацій коду (упакування, поліморфних двигунів, шифрування рядків), які були предметом роботи [5].

У нещодавньому дослідженні Xu, Zhou, Ming та Lyu [10] запропоновано підхід шарової таксономії обфускацій (layered obfuscation). У роботі використано принцип шарової безпеки (layered security), запозичений із практики управління ризиками. Використовуючи його, автори пропонують чотирирівневу таксономію за об'єктом захисту: шар елементів коду (code-element layer), що охоплює перетворення компонування, потоку керування, даних, функцій та класів, шар програмних компонентів (software-component layer), що охоплює трансляцію коду, віртуалізаційні підходи виконання, шар міжкомпонентної взаємодії (inter-component layer), та шар специфічних застосунків (application layer), що охоплює підходи, розроблені для конкретних предметних сфер. Також підкреслено, що гілки таксономії є взаємно незалежними і техніки з різних шарів можуть поєднуватися в єдине комплексне рішення захисту, яке автори позначають як шарову обфускацію. Однак, аналіз роботи виявив два обмеження. По-перше, заявлена незалежність шарів залишається тільки на рівні опису – у запропонованій схемі кожна техніка віднесена лише до одного шару і не сформульовано критерій вибору шару для технік, що впливають одразу на кілька рівнів. Прикладом такої техніки є упакування виконуваного файлу, яке одночасно змінює метадані файлу, спосіб завантаження програми у пам'ять та структуру самого коду, однак із запропонованої таксономії не випливає яким саме шаром такі техніки слід описувати. По-друге, таксономію розроблено для розробників, які захищають власне програмне забезпечення від реверс-інжинірингу, а не для аналітиків, які класифікують обфускацію у виявленому ШПЗ. Як наслідок, у даному дослідженні техніки протидії аналізу не виокремлено, оскільки вони виходять за межі функціональної рамки захисту легітимного коду, для якої таксономію було побудовано.

Окремо варто згадати MITRE ATT&CK – стандарт, який широко застосовується в індустрії кібербезпеки для опису поведінки зловмисників під час реальних кіберінцидентів. У ньому міститься матриця, де елемент обфускації подається як техніка (Obfuscated Files or Information (T1027) [2]), що об'єднує близько двох десятків підтехнік, серед яких упакування програмного забезпечення, стеганографія, HTML Smuggling та обфускація командного рядка. Однак, оскільки основним призначенням MITRE ATT&CK є визначення поведінки зловмисників, даний стандарт не містить систематизації обфускаційних технік за їхніми структурними характеристиками.

Робота Aboaoja, Zainal, Ghaleb, Al-Rimy, Eisa та Elnour [11] систематизує методи аналізу ШПЗ (статичний, динамічний, гібридний), з прив'язкою кожного методу до типових артефактів, які отримуються із зразка для подальшого аналізу. Вибір методу аналізу визначається даними, отриманими із артефактів, а не властивостями обфускаційної техніки, яка застосовується у зразку, незважаючи на те, що структурна природа техніки обфускації потенційно дозволяє визначити, який метод аналізу буде проти неї оптимальним. У роботі не сформульовано зв’язок застосованої техніки обфускації з оптимальним методом аналізу, хоча обфускацію зафіксовано як ключовий фактор зниження ефективності всіх трьох методів.

Сучасну таксономію типів пакувальників і методів виявлення упакованих PE-файлів подано у циклі праць Alkhateeb, Ghorbani та Habibi Lashkari [16, 17]. Автори систематизували види пакувальників, запропонували життєвий цикл обробки упакованого ШПЗ для антивірусних систем та продемонстрували можливості ідентифікації типу пакувальника за статичними ознаками. Зазначені роботи показують, що упакування є комплексною технікою, яка має вплив одночасно на метадані файлу, код і середовище виконання. Це дозволяє зробити висновок, що упакування за своїм впливом важко помістити в одну гілку ієрархічної таксономії, воно потребує одночасного опису за кількома незалежними вимірами.

Окремим завданням літературного огляду став пошук методологічної основи для побудови таксономії. Така основа має забезпечувати відтворюваність класифікації та можливість її незалежної

перевірки. Як показало дослідження De Sutter, Schrittwieser, Coppens та Kochberger [19], які провели аналіз 571 публікації з оцінювання захисного програмного забезпечення, у галузі обфускації наразі немає загальноприйнятої методологічної бази. Автори роблять висновок, що наявні класифікаційні схеми будуються переважно інтуїтивно, без посилання на формальний метод побудови таксономії. Це становить самостійний методологічний недолік, який створює потребу у застосуванні верифікованого методу побудови таксономій у дослідженнях, які претендують на систематичність та відтворюваність класифікаційних результатів.

Водночас у інформаційних системах та суміжних класифікаційних дослідженнях широко застосовується метод Nickerson, Varshney та Muntermann [7], з оновленнями Kundisch та співавторів [18]. Він полягає у чергуванні емпіричних і концептуальних кроків із визначеними умовами побудови, як об'єктивними (виміри незалежні, разом покривають усі об'єкти, кожен об'єкт отримує свою класифікацію), так і суб'єктивними (лаконічність, можливість розширення, пояснювальна сила). У кібербезпеці цей метод застосовується для побудови таксономій кіберполігонів, шкідливого мережевого трафіку та методів виявлення вторгнень. Однак у домені обфускації ШПЗ його застосування досі не зафіксовано, тому метод обрано як методологічну основу цього дослідження.

Мета статті. Метою дослідження є розроблення багатовимірної фасетної таксономії технік обфускації ШПЗ, яка усуває системні обмеження наявних класифікаційних схем. Запропонована таксономія забезпечує опис багатоаспектних технік за кількома незалежними характеристиками, інтегрує два паралельні напрями технік приховування шкідливої поведінки у спільну класифікаційну рамку зі структурним виокремленням між ними та побудована за верифікованою методологією побудови таксономії.

Для досягнення поставленої мети сформульовано такі завдання:

1. Обґрунтувати спільний розгляд класичних обфускаційних технік та технік протидії аналізу як єдиного предмета класифікації, з огляду на їхню спільну функцію приховування шкідливої поведінки програми від аналітичних інструментів.
2. Застосувати метод Nickerson, Varshney та Muntermann [7] як методологічну основу побудови таксономії.
3. Визначити та охарактеризувати незалежні виміри таксономії.
4. Сформувати алгоритм класифікації обфускаційних технік за запропонованою таксономією.
5. Застосувати таксономію до представницького набору обфускаційних технік та перевірити її розрізнювальну здатність і виконання об'єктивних умов завершеності.

МЕТОДИКА ДОСЛІДЖЕННЯ

Поняття обфускації у семантичному розумінні охоплює трансформації, які зберігають функціональність програми, тоді як техніки протидії аналізу (anti-debugging, anti-virtualization, anti-sandbox) формально змінюють поведінку програми залежно від властивостей середовища виконання. Однак, у контексті систем виявлення ШПЗ обидва класи технік виконують спільну функцію – приховування шкідливої поведінки програми від аналітичних інструментів. Сучасні зразки ШПЗ, як правило, поєднують трансформації коду з техніками протидії аналізу, і стійкість зразка до виявлення визначається саме як об'єднання впливів обох класів [15]. З огляду на це, у статті прийнято розширене розуміння обфускації, що охоплює як техніки трансформації коду, так і техніки протидії аналізу. Вибраний підхід узгоджується з іншими релевантними дослідженнями. Наприклад, огляд методів виявлення ШПЗ Aboaoja та інші [11] використовує узагальнене поняття «технік ухилення від виявлення», у якому не відображено формального розрізнення між трансформаційними та поведінково-залежними техніками. Також, варто згадати дослідження Опірського, Дзьобана та Василишина [9], присвячене методам обходу систем виявлення на кінцевих точках (EDR) у поєднанні з системами управління інформацією та подіями безпеки (SIEM). Автори розглядають техніки обфускації коду, методи Living-off-the-Land, маніпуляції з журналами та атаки на рівні ядра операційної системи у межах єдиної класифікаційної рамки методів ухилення від виявлення, не виокремлюючи трансформаційні та поведінково-залежні компоненти у різні структури.

Як методологічну основу дослідження обрано метод побудови таксономій, запропонований Nickerson, Varshney та Muntermann [7]. Даний метод був розроблений для предметних областей, де таксономія має водночас бути теоретично обгрунтованою та емпірично верифікованою. У його основі лежить використання двох типів циклів. Емпірико-концептуальний цикл (E2C) полягає у огляді конкретних об’єктів предметної області, з яких далі виводяться узагальнення, як потенційні характеристики таксономії. Концептуально-емпіричний цикл (C2E) відображає зворотну дію – характеристики таксономії формулюються спочатку на основі теоретичних даних, а потім

підтверджуються на конкретних об’єктах. Цикли виконуються по черзі, доки не буде досягнуто об'єктивних умов завершеності. Ці умови полягають у тому, що нові об'єкти не додають нових характеристик, виміри є взаємно незалежними, а кожен об'єкт отримує визначений класифікаційний кортеж. Також, додатково перевіряються суб’єктивні критерії, такі як концептуальна повнота, лаконічність, розширюваність, поясненість, стійкість. На рисунку 1 наведено принцип побудови таксономії за Nickerson.

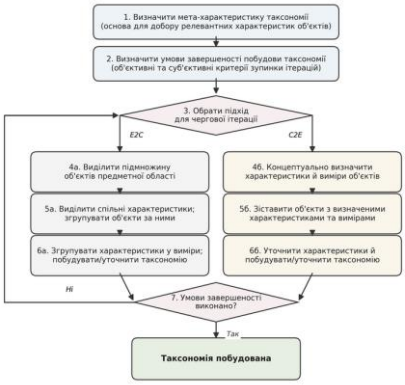

Рис. 1. Принцип побудови таксономії за Nickerson.

Оскільки обфускаційні техніки є багатоаспектними об'єктами, доцільно розглянути вибір саме фасетного, а не ієрархічного оформлення таксономії. Кожна техніка одночасно характеризується кількома незалежними властивостями, зокрема на якому рівні вона проявляється, яку частку програми охоплює, який тип операції виконує, коли активується та як змінюється поведінка програми у різних середовищах. При використанні ієрархічної структури, потрібно вибрати одну із цих властивостей як основну, що призводить до неоднозначного розміщення багатоаспектних технік. Натомість, фасетний підхід зберігає опис властивостей техніки, фіксуючи їх значення за кожним виміром [10, 15]. Ключовим елементом, який визначає, які саме властивості об’єктів релевантні для класифікації, є метахарактеристика. Для даної таксономії, такою метахарактеристикою є спосіб, у який техніка обфускації перешкоджає виявленню ШПЗ, оскільки саме цей аспект практично цікавить розробників систем захисту.

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Виміри таксономії отримано за три ітерації методу Nickerson [7]. Під час ітерації 1 (емпірико-концептуальний цикл) було проаналізовано опис обфускаційних технік у класичних та сучасних оглядах [3, 5, 6, 8, 10, 12, 15] і виокремлено повторювані аспекти, за якими автори описують та порівнюють техніки. Зведення цих повторюваних аспектів до взаємно непокривних категорій дало чотири попередні характеристики таксономії: рівень спостережуваності, масштаб впливу, механізм перетворення, фаза прояву ефекту.

На ітерації 2 (концептуально-емпіричний цикл) сформульовано гіпотезу, згідно з якою для розрізнення класичних обфускаційних технік та технік протидії аналізу необхідний окремий вимір, що фіксує залежність поведінки програми від середовища виконання. Гіпотезу перевірено на семи техніках – двох з прив'язкою поведінки до параметрів цільового середовища (Environmental Keying, Targeted Decryption), двох з активним виявленням аналітичного середовища (Anti-Debugging, Anti-Virtualization) та трьох класичних обфускаційних техніках (Control Flow Flattening, String Encryption, Subroutine Reordering). Класифікація тестових технік розподілила їх на три групи. До першої групи увійшли техніки, що поводяться однаково у будь-якому середовищі, а саме Control Flow Flattening, String Encryption та Subroutine Reordering. До другої групи увійшли техніки, що змінюють поведінку при виявленні аналітичного середовища, а саме Anti-Debugging та Anti-Virtualization. До третьої групи увійшли техніки, поведінка яких визначається параметрами цільового середовища, а саме Environmental Keying та Targeted Decryption. Отриманий розподіл виявився інформативним та незалежним від жодної з чотирьох характеристик першої ітерації. На підставі даної структури вимір поведінкової інваріантності зі значеннями «Повна», «Часткова» та «Відсутня» включено до таксономії.

Під час ітерації 3 (повторний емпірико-концептуальний цикл), усі п'ять вимірів, отриманих у попередніх ітераціях, застосовано до розширеного набору з 16 обфускаційних технік, відібраних за критерієм покриття вимірів, згідно з яким кожне значення кожного виміру отримало щонайменше одного

представника у вибірці. У результаті ітерації виконано перевірку об'єктивних умов завершеності таксономії за методологією Nickerson [7].

Перша умова полягає у насиченості значень і встановлена через демонстрацію того, що жодна з 16 технік розширеного набору не зумовила потреби у введенні нових значень у жодному з п'яти вимірів. Усі техніки розподілилися між уже визначеними значеннями.

Друга умова полягає у незалежності вимірів. Для кожної пари вимірів у вибірці існують техніки, що збігаються за значенням одного виміру пари, але різняться за значенням іншого, що доводить відсутність взаємної залежності між вимірами. Зазначена умова стосується незалежності вимірів у межах усієї вибірки і не вимагає унікальності кортежу для кожної окремої техніки.

Третя умова, полягає в однозначності класифікації, тобто чи отримує кожна техніка визначений 5-кортеж і чи не виникає при цьому суперечливих або невизначених класифікацій. У межах розширеного набору з 16 технік усі техніки отримали однозначно визначені кортежі, проте у одному випадку кортежі двох технік виявилися ідентичними, оскільки обидві ці техніки виявляють аналітичне середовище через виклики до системних API у фазі ініціалізації виконання, і потребують однакового методу виявлення у вигляді інструментованого динамічного аналізу та реалізують ту саму обфускаційну функцію.

Структуру таксономії за п'ятьма вимірами подано в таблиці 1.

Таблиця 1

Виміри запропонованої фасетної таксономії

Символ	Назва виміру	Допустимі значення	Визначення
S	Рівень спостережуваності	• Код • Дані • Середовище • Метадані	На якому артефакті аналітик може виявити ефект обфускації
C	Масштаб впливу	• Вузький • Середній • Широкий	Яку частку програми охоплює трансформація
M	Механізм перетворення	• Заміна • Вставка • Реструктуризація • Кодування • Видалення • Втручання	Тип операції, що виконується над програмою
P	Фаза прояву ефекту	• Статичний • Завантаження • Умовний • Постійний	Момент, коли ефект обфускації стає активним
I	Поведінкова інваріантність	• Повна • Часткова • Відсутня	Визначає різницю в поведінці програми у цільовому й аналітичному середовищах

Перший виміром таксономії є рівень спостережуваності (Scope), що представляє рівень на якому зміни є видимими для аналізу, а також встановлює зв’язок між класом техніки та методом її виявлення. Запропонована у даній роботі таксономія, на відміну від шарової схеми Xu та ін. [10], упорядковує техніки не за рівнем розгортання захисту, а за рівнем прояву обфускаційного ефекту, і виокремлює чотири значення цього виміру: рівень коду (зміни структури інструкцій та потоку керування), даних (трансформація рядків, констант, масивів), середовища виконання (взаємодія з ОС, debugger, sandbox) та метаданих файлу (заголовки PE/ELF, таблиці імпорту). Значення «Код» і «Дані» були виокремлені ще у класичній таксономії Collberg та ін. [5] як два з чотирьох основних класів трансформацій. Значення «Метадані» з'являється у пізнішій літературі [16], присвяченій пакувальникам, як рівень заголовків виконуваних файлів і таблиць імпорту. Значення «Середовище» додано у запропонованій таксономії. Жодна з наявних класифікаційних схем не виокремлює його як окремий рівень прояву ефекту, який застосовується переважно до технік протидії аналізу, що оперують саме на цьому рівні [4, 6].

Аналізуючи значення виміру рівня спостережуваності, можна встановити зв'язок між класом техніки та методом її виявлення, відповідність між значенням виміру та потрібним типом аналізу. Значення «Код» вказує на придатність статичного аналізу структури коду [5, 8], значення «Дані» – на

статичний аналіз вмісту секцій з оцінюванням ентропії [3], «Метадані» – на статичний аналіз структури виконуваного файлу та ідентифікацію пакувальника [16], а «Середовище» – на необхідність динамічного або гібридного аналізу, оскільки ефект проявляється лише під час взаємодії програми із середовищем виконання [4, 6]. Таким чином, значення виміру безпосередньо переходить у вибір типу аналізу, що усуває потребу в інтуїтивному виборі методу виявлення.

Другий вимір представляє масштаб впливу (Transformation Coverage), який відповідає за те, яку частину програми охоплює трансформація. У літературі простежується розрізнення трьох масштабів: вузький (окремі інструкції чи операнди), середній (базові блоки або функції) та широкий (модуль або програма загалом) [5, 10]. Усі три значення цього виміру згадуються в літературі, зокрема Collberg та інші [5] використовують споріднену чотирирівневу шкалу від локального до міжпроцесного масштабу, а Xu та інші [10] використовують схоже розрізнення у основі чотирирівневої шарової таксономії. Цей вимір необхідний як для повноти опису, так і для оцінки трудомісткості як обфускації, так і деобфускації, оскільки техніки вузького масштабу простіше нейтралізуються локальними перетвореннями, тоді як техніки широкого масштабу потребують більш глобального аналізу.

Третім виміром є механізм перетворення (Transformation Mechanism), який фіксує тип операції, що виконується над програмою. На основі узагальнення механізмів, описаних у класичних і сучасних оглядах [3, 5, 8], виокремлено шість типів операцій: заміна (елементи замінюються еквівалентними), вставка (додаються нові елементи), реструктуризація (змінюється порядок існуючих елементів), кодування (шифрування або стиснення), видалення (усунення символів, метаданих) та втручання (вплив на сам аналітичний інструмент). Ця вибірка природно узгоджується з шістьма базовими механізмами обфускації коду, описаними у дослідженні [8] та доповнюється операціями «видаленням» і «втручанням», які ширше представленими у літературі з протидії аналізу [6].

Четвертим класифікаційним виміром запропонованої таксономії є фаза прояву ефекту (Effect Manifestation Phase), що визначає момент активації обфускаційного ефекту у життєвому циклі програми. Систематизація фаз активації механізмів протидії аналізу, наведена у [6], виокремлює чотири значення цього виміру. Значення «Статичний» відповідає ефекту від техніки, який повністю присутній у файлі на диску. Значення «Завантаження» відповідає ефекту, що активується у момент завантаження програми в пам'ять. Значення «Умовний» відповідає ефекту, що активується лише за певної події, зокрема виявлення відлагоджувача. Значення «Постійний» відповідає ефекту, що залишається активним протягом усього виконання програми. Зазначений вимір необхідний для розрізнення обфускаційних технік, що мають подібний механізм перетворення, проте активуються у різні моменти і потребують відповідно різних методів виявлення. Значення «Статичний» та «Умовний» виокремлено у систематизації Afianian та інші [6] як основні фази активації механізмів протидії аналізу. Значення «Завантаження» детально описане у літературі, присвяченій пакувальникам [16], як момент розпакування та передачі керування виконуваному коду. Значення «Постійний» у наявних класифікаційних схемах не наводиться, його включено до запропонованої таксономії для класифікації обфускаційних технік поліморфного шифрування та віртуалізації коду [10].

П'ятим класифікаційним виміром запропонованої таксономії є поведінкова інваріантність (Behavioral Invariance), що надає механізм розрізнення між двома функціональними класами обфускаційних технік у межах інтегрованої класифікаційної рамки. Концептуальну основу зазначеного виміру становить інтеграція двох теоретичних понять, що раніше використовувалися у літературі для опису різних класів обфускаційних технік.

Для забезпечення відтворюваності класифікації за виміром поведінкової інваріантності нижче подано його формальне означення. Нехай E_t позначає цільове середовище, у якому шкідлива програма буде виконуватися. Нехай E_a позначає аналітичне середовище, таке як ізольована пісочниця, відлагоджувач або віртуальна машина. Нехай β (P, E) позначає функціональну поведінку програми P у середовищі E, операційно визначену як послідовність системних викликів та змін стану операційної системи, ініційованих програмою. Значення виміру поведінкової інваріантності визначаються через властивості функції β.

Значення «Повна» надається техніці, якщо програма, що використовує цю техніку не містить механізму, що ставить її функціональну поведінку у залежність від властивостей середовища виконання. Для такої техніки β (P, E_a) збігається з β (P, E_t) незалежно від конкретного аналітичного середовища.

Значення «Часткова» надається техніці, якщо поведінка програми, що використовує цю техніку залежить від результату перевірки ознак аналітичного середовища. Тоді існує таке середовище E_a, для якого β (P, E_a) відрізняється від β (P, E_t).

Значення «Відсутня» надається техніці, якщо коректна шкідлива функціональність програми активується лише за умови відповідності параметрів середовища виконання заздалегідь заданому набору значень, а за невідповідності цих параметрів шкідлива функціональність не активується. На відміну від

часткової інваріантності, тут програма не виявляє ознак аналітичного оточення, а її поведінка визначається безпосередньо параметрами середовища. Структура таксономії та алгоритм класифікації. Запропонована таксономія задається п'ятьма незалежними вимірами, у кожного з яких є своя множина допустимих значень. Кортеж $T = (S, C, M, P, I)$ описує структуру таксономії, де S позначає рівень спостережуваності, C позначає масштаб впливу, M позначає механізм перетворення, P позначає фазу прояву ефекту та I позначає поведінкову інваріантність. Домени зазначених вимірів визначені у такій формі:

$$S \in \{Код, Дані, Середовище, Метадані\},$$

$$C \in \{Вузький, Середній, Широкий\},$$

$$M \in \{Заміна, Вставка, Реструктуризація, Кодування, Видалення, Втручання\},$$

$$P \in \{Статичний, Завантаження, Умовний, Постійний\},$$

$$I \in \{Повна, Часткова, Відсутня\}.$$

Класифікація обфускаційної техніки τ задається відображенням:

$$class: \mathcal{T} \rightarrow S \times C \times M \times P \times I, \quad (1)$$

Для кожної обфускаційної техніки $\tau \in \mathcal{T}$ функція class визначає 5-кортеж (s, c, m, p, i) зі значень п'яти вимірів таксономії, що відповідають класифікованій техніці.

Загальна кількість теоретично можливих 5-кортежів визначається добутком кількості значень п'яти вимірів та обчислюється у такій формі:

$$|S| \cdot |C| \cdot |M| \cdot |P| \cdot |I| = 4 \cdot 3 \cdot 6 \cdot 4 \cdot 3 = 864. \quad (2)$$

Зазначене число показує максимальну розрізнювальну здатність таксономії та слугує верхньою оцінкою кількості функціонально різних класів обфускаційних технік, які можуть бути виокремлені у запропонованій рамці. У реальному корпусі обфускаційних технік не всі 864 комбінації мають функціональну реалізацію, оскільки частина з них передбачає поєднання значень, що є взаємовиключними з огляду на технічну природу обфускаційних механізмів.

Класифікація конкретної обфускаційної техніки за запропонованою таксономією здійснюється шляхом послідовного визначення значень кожного з п'яти вимірів. Вхідними даними алгоритму є опис техніки, що включає опис принципу її роботи, посилання на джерела з описом техніки, опис типових артефактів роботи техніки у виконуваному файлі та у середовищі виконання. Вихідними даними алгоритму є кортеж $T(\tau) = (s, c, m, p, i)$, що формалізує класифікацію техніки відповідно до функції (1). Алгоритм класифікації наведено на рисунку 2, він складається з наступних кроків:

Рис. 2. Схематичне представлення алгоритму класифікації обфускаційної техніки за запропонованою таксономією

1. Аналіз опису техніки з виокремленням ключових індикаторів кожного з п'яти вимірів.
2. Визначення значення виміру S через ідентифікацію артефакту програми, на якому переважно проявляється ефект обфускації.

3. Визначення значення виміру C через ідентифікацію частки програми, що охоплена трансформацією.

4. Визначення значення виміру M через ідентифікацію типу операції, що виконує техніка над програмою.

5. Визначення значення виміру P через ідентифікацію моменту активації ефекту обфускації у життєвому циклі програми.

6. Визначення значення виміру I через ідентифікацію характеру залежності поведінки програми від середовища виконання.

7. Формування кортежу таксномії $T(\tau) = (s, c, m, p, i)$ та перевірка його внутрішньої узгодженості.

Для обфускаційних технік, що проявляються одночасно на кількох рівнях програми, застосовується процедура багатоміткового маркування (multi-label labelling), що передбачає виокремлення первинного рівня прояву (primary scope), на якому зосереджений головний обфускаційний ефект, та вторинних рівнів прояву (secondary scopes), які фіксуються у розширеному профілі техніки [10]. Прикладом такої техніки є упакування виконуваного файлу, що одночасно модифікує метадані файлу, трансформує структуру виконуваного коду та взаємодіє із середовищем виконання у фазі завантаження. Первинним рівнем прояву упакування у запропонованій таксономії визначено метадані, оскільки структурна зміна заголовків PE/ELF та таблиці імпорту є тією властивістю, за якою упакування виявляється у статичному аналізі з найбільшою надійністю. Вторинними рівнями прояву упакування зафіксовано код та середовище.

Приклади класифікації обфускаційних технік. Для демонстрації застосування алгоритму класифікації, розглянуто дві обфускаційні техніки, що належать до різних функціональних класів та структурно відрізняються за усіма п'ятьма вимірами таксономії.

Першим прикладом є техніка трансформації коду зі збереженням функціональної семантики, другим прикладом є техніка протидії аналізу. Класифікація обох технік здійснена за послідовністю кроків алгоритму класифікації з визначенням значень кожного з п'яти вимірів на основі опису принципу роботи технік у джерелах [10, 13]. Прикладом техніки трансформації коду є розплощення потоку керування (Control Flow Flattening, CFF). Зазначена техніка замінює природну структуру розгалужень функції на єдиний центральний цикл-диспетчер із таблицею станів, у якій кожному оригінальному базовому блоку функції відповідає окремий стан. Ефект техніки проявляється у структурі графа потоку керування (control flow graph), який повною мірою присутній у статичному вигляді виконуваного файлу та не змінюється протягом виконання програми. На основі зазначеного опису класифікація техніки за п'ятьма вимірами здійснена у наступній формі. Значення виміру S визначено як «Код», оскільки ефект техніки проявляється на рівні структури коду функції. Значення виміру C визначено як «Середній», оскільки трансформація охоплює окрему функцію без поширення на міжфункційні зв'язки. Значення виміру M визначено як «Реструктуризація», оскільки трансформація змінює структуру взаємозв'язку базових блоків функції без додавання нового коду. Значення виміру P визначено як «Статичний», оскільки ефект техніки повністю присутній у статичному вигляді виконуваного файлу. Значення виміру I визначено як «Повна», оскільки трансформація зберігає функціональну семантику програми та забезпечує її поведінкову ідентичність у будь-якому середовищі виконання. Сформована класифікація техніки CFF має вигляд: T(CFF) = (Код, Середній, Реструктуризація, Статичний, Повна).

Прикладом техніки трансформації коду є протидія відлагодженню (Anti-Debugging). Зазначена техніка реалізує перевірку ознак присутності аналітичного середовища через звернення до системних інтерфейсів операційної системи. До типових механізмів перевірки належать виклики системних функцій IsDebuggerPresent, CheckRemoteDebuggerPresent та NtQueryInformationProcess, що повертають інформацію про наявність приєднаного до програми відлагоджувача, перевірка прапорця стану відлагодження у структурі параметрів процесу, перевірка стану апаратних точок зупину у регістрах відлагодження процесора, а також перевірки часових інтервалів виконання критичних ділянок коду, що дозволяють виявити уповільнення виконання у середовищі інструментованого аналізу [13]. Зазначені перевірки активуються лише у момент звернення до них під час виконання програми, після чого програма змінює поведінку через припинення виконання або перехід до стану призупинення активності. На основі зазначеного опису класифікація техніки за п'ятьма вимірами здійснена у наступній формі. Значення виміру S визначено як «Середовище», оскільки ефект техніки проявляється у взаємодії програми з операційною системою та аналітичним інструментарієм. Значення виміру C визначено як «Широкий», оскільки техніка охоплює усю програму через впровадження механізмів виявлення аналітичного середовища у її початковій частині та подальшу реакцію на результати виявлення. Значення виміру M визначено як «Втручання», оскільки техніка реалізує активний вплив на аналітичний інструмент через його виявлення та зміну поведінки програми у відповідь на результати виявлення. Значення виміру P визначено як «Умовний», оскільки ефект техніки активується лише за умови

виявлення аналітичного середовища. Значення виміру I визначено як «Часткова», оскільки техніка передбачає свідому зміну поведінки програми при виявленні аналітичного середовища через явні механізми його виявлення. Сформована класифікація техніки протидії відлагодженню має вигляд: T(Anti-Debug) = (Середовище, Широкий, Втручання, Умовний, Часткова).

Графічне подання розміщення зазначених двох технік у класифікаційному просторі таксономії наведено на рисунку 3. Рисунок демонструє розташування технік CFF та Anti-Debugging у багатовимірному просторі вимірів та унаочнює структурну відмінність між класифікаційними кортежами зазначених технік за усіма п'ятьма вимірами таксономії.

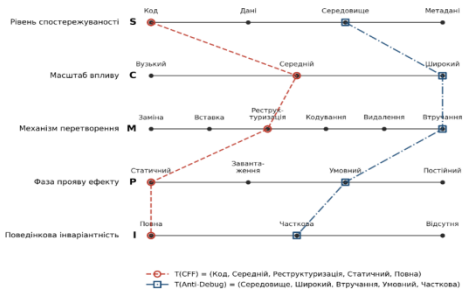

Рис. 3. Відображення розміщення двох технік у класифікації

Зведена таблиця класифікації 16 технік. Для перевірки розрізнювальної здатності й покриття, таксономію застосовано до представницького набору з 16 обфускаційних технік, відібраних на основі огляду літератури [3, 6, 8, 10, 12, 20] як таких, що зустрічаються у сучасному ШПЗ найчастіше і покривають широкий спектр механізмів. Перелік технік та їхні класифікації наведено у таблиці 2.

Таблиця 2

Класифікація 16 обфускаційних технік за фасетною таксономією T = (S, C, M, P, I)

№	Техніка	S	C	M	P	I
1	Control Flow Flattening	Код	Середній	Реструктуризація	Статичний	Повна
2	Anti-Debugging	Середовище	Широкий	Втручання	Умовний	Часткова
3	Anti-Virtualization	Середовище	Широкий	Втручання	Умовний	Часткова
4	Packing (UPX-like)	Метадані	Широкий	Кодування	Завантаження	Повна
5	String Encryption	Дані	Вузький	Кодування	Завантаження	Повна
6	Junk Code Insertion	Код	Вузький	Вставка	Статичний	Повна
7	Instruction Substitution	Код	Вузький	Заміна	Статичний	Повна
8	Environmental Keying	Середовище	Широкий	Кодування	Умовний	Відсутня
9	Subroutine Reordering	Код	Широкий	Реструктуризація	Статичний	Повна
10	Code Transposition	Код	Середній	Реструктуризація	Завантаження	Повна
11	API Hashing	Дані	Вузький	Кодування	Умовний	Повна
12	Import Table Obfuscation	Метадані	Середній	Видалення	Завантаження	Повна
13	Polymorphic Encryption	Код	Широкий	Кодування	Завантаження	Повна
14	Self-Modifying Code	Код	Середній	Заміна	Постійний	Повна
15	Code Virtualization	Код	Середній	Кодування	Постійний	Повна
16	Continuous Environment Monitoring	Середовище	Широкий	Втручання	Постійний	Часткова

Обмеження запропонованої таксономії. Запропонована таксономія має низку істотних обмежень. Перше обмеження пов'язане з тим, що визначення значень окремих вимірів таксономії для конкретної техніки може потребувати експертної оцінки. Це стосується переважно виміру механізму перетворення, оскільки його домен містить значення, які є концептуально близькими та можуть застосовуватися до однієї і тієї ж техніки. Наприклад, для технік, що замінюють оригінальні елементи коду еквівалентними, доводиться обирати між значеннями «Заміна» та «Кодування». У таких випадках людина яка проводить класифікацію спирається на власне розуміння сутності техніки, що вносить у процедуру класифікації елемент суб'єктивної оцінки. Для усунення цього елемента застосовується процедура валідації через

залучення кількох незалежних експертів. Експерти класифікують однаковий набір технік, після чого результати порівнюються для встановлення ступеня їхнього збігу [7]. У межах поточного дослідження така процедура не виконувалася і становить напрям подальших робіт з валідації запропонованої таксономії.

Друге обмеження пов'язане з тим, що запропонована таксономія не пройшла емпіричну валідацію на повномасштабному корпусі сучасного ШПЗ. Розглянутий у дослідженні набір з 16 обфускаційних технік є репрезентативним, проте кількісно обмеженим. Він охоплює основні класи обфускаційних механізмів, відомих з літератури, але не претендує на вичерпну реконструкцію їх сучасного арсеналу. Для встановлення повноти охоплення таксономії у реальних умовах, необхідно провести дослідження її застосування до ширшого корпусу зразків ШПЗ.

Третє обмеження полягає у тому, що запропонована таксономія сама по собі не може бути застосована до невідомого зразка ШПЗ без попереднього виявлення обфускації у ньому. Таксономія класифікує вже ідентифіковані обфускаційні техніки, але не виявляє їх у досліджуваному зразку. Тому для практичного застосування таксономії до конкретного зразка необхідне попереднє використання інструментів виявлення обфускації, які належать до окремого класу аналітичних засобів. Без таких інструментів запропонована таксономія не може використовуватися як самостійний компонент автоматизованого аналізу ШПЗ. Інтеграція таксономії з засобами виявлення обфускації становить необхідну умову її повноцінного практичного впровадження.

ВИСНОВКИ І ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

У роботі розв'язано наукову задачу формалізації багатовимірної фасетної таксономії технік обфускації шкідливого програмного забезпечення. Запропонований підхід дозволяє перейти від ієрархічних класифікаційних схем до фасетної моделі, у якій техніка обфускації описується незалежним кортежем п'яти вимірів.

У ході аналізу наявних таксономій було виявлено три спільні системні обмеження. По-перше, наявні схеми не дозволяють описати техніку одночасно за кількома її властивостями, через що багатоаспектні техніки (наприклад, упакування виконуваного файлу) отримують неповний опис. По-друге, у літературі не запропоновано інтегрованої класифікаційної схеми, що охоплювала б класичні обфускаційні техніки та техніки протидії аналізу зі структурним розрізненням між ними. По-третє, наявні класифікаційні схеми у домені обфускації побудовані без застосування формального методу побудови таксономій. Для подолання зазначених обмежень як методологічну основу побудови таксономії застосовано метод Nickerson, Varshney та Muntermann, який систематично використовується у суміжних доменах кібербезпеки, проте у домені обфускації ШПЗ застосований уперше. Виміри таксономії отримано за три ітерації методу. Сформована фасетна таксономія містить п'ять незалежних вимірів, а саме рівень спостережуваності, масштаб впливу, механізм перетворення, фазу прояву ефекту та поведінкову інваріантність. Сформовано алгоритм класифікації техніки за п'ятьма вимірами з диференціальними критеріями для випадків неоднозначності. Таксономію застосовано до 16 представницьких обфускаційних технік сучасного ШПЗ. Окремим результатом роботи є формалізація виміру поведінкової інваріантності, що операційно відрізняє класичні обфускаційні техніки від технік протидії аналізу та забезпечує одночасне об'єднання двох напрямів технік приховування у спільну класифікаційну рамку зі збереженням структурного розрізнення між ними.

Практична значущість запропонованого підходу полягає у можливості безпосереднього співвідношенння класу техніки з придатним методом її виявлення. Значення виміру рівня спостережуваності вказує на тип потрібного аналізу (статичний, динамічний або комбінований), а значення виміру поведінкової інваріантності вказує на необхідність приховування аналітичних інструментів від досліджуваної програми. Перспективи подальших досліджень охоплюють два напрями. Перший напрям полягає в емпіричній валідації таксономії на повномасштабному корпусі ШПЗ та усуненні наведених обмежень. Другий напрям полягає у використанні запропонованої таксономії як формальної основи для проектування систем виявлення обфускованого ШПЗ на основі машинного навчання, де класифікаційний кортеж техніки потенційно визначає тип необхідного аналізу, релевантний простір ознак та конфігурацію середовища спостереження.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. AV-TEST Institute. (2025). *Malware statistics & trends report (2024-2025)*. https://www.av-test.org/en/statistics/malware/
2. MITRE. (n.d.). *Obfuscated files or information (Technique T1027)*. MITRE ATT&CK® Framework. https://attack.mitre.org/techniques/T1027/

3. Brezinski, K., & Ferens, K. (2023). Metamorphic malware and obfuscation: A survey of techniques, variants, and generation kits. *Security and Communication Networks, 2023*, Article 8227751, 1-41. https://doi.org/10.1155/2023/8227751
4. Galloro, N., Polino, M., Carminati, M., Continella, A., & Zanero, S. (2022). A systematical and longitudinal study of evasive behaviors in Windows malware. *Computers & Security, 113*, 102550. https://doi.org/10.1016/j.cose.2021.102550
5. Collberg, C., Thomborson, C., & Low, D. (1997). *A taxonomy of obfuscating transformations* (Technical Report No. 148). Department of Computer Science, The University of Auckland. https://researchspace.auckland.ac.nz/handle/2292/3491
6. Afianian, A., Niksefat, S., Sadeghiyan, B., & Baptiste, D. (2019). Malware dynamic analysis evasion techniques: A survey. *ACM Computing Surveys, 52*(6), Article 126, 1-28. https://doi.org/10.1145/3365001
7. Nickerson, R. C., Varshney, U., & Muntermann, J. (2013). A method for taxonomy development and its application in information systems. *European Journal of Information Systems, 22*(3), 336-359. https://doi.org/10.1057/ejis.2012.26
8. You, I., & Yim, K. (2010). Malware obfuscation techniques: A brief survey. In *Proceedings of the 2010 International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA 2010)* (pp. 297-300). IEEE. https://doi.org/10.1109/BWCCA.2010.85
9. Opirskyi, I., Dzioban, T., & Vasylyshyn, S. (2025). Bypassing EDR in combination with SIEM: Analysis of methods for hiding attacks in logs – A study of tactics used by attackers to avoid detection. *Cybersecurity: Education, Science, Technique, 1*(29), 8-26. https://doi.org/10.28925/2663-4023.2025.29.865
10. Xu, H., Zhou, Y., Ming, J., & Lyu, M. (2020). Layered obfuscation: A taxonomy of software obfuscation techniques for layered security. *Cybersecurity, 3*, Article 9. https://doi.org/10.1186/s42400-020-00049-3
11. Aboaoja, F. A., Zainal, A., Ghaleb, F. A., Al-Rimy, B. A. S., Eisa, T. A. E., & Elnour, A. A. H. (2022). Malware detection issues, challenges, and future directions: A survey. *Applied Sciences, 12*(17), 8482. https://doi.org/10.3390/app12178482
12. Asghar, H. J., Zhao, B. Z. H., Ikram, M., Nguyen, G., Kaafar, D., Lamont, S., & Coscia, D. (2024). Use of cryptography in malware obfuscation. *Journal of Computer Virology and Hacking Techniques, 20*(1), 135-152. https://doi.org/10.1007/s11416-023-00504-y
13. Park, J., Jang, Y.-H., Hong, S., & Park, Y. (2019). Automatic detection and bypassing of anti-debugging techniques for Microsoft Windows environments. *Advances in Electrical and Computer Engineering, 19*(2), 23-28. https://doi.org/10.4316/AECE.2019.02003
14. Carrier, T., Victor, P., Tekeoglu, A., & Lashkari, A. H. (2022). Detecting obfuscated malware using memory feature engineering. In P. Mori, G. Lenzini, & S. Furnell (Eds.), *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP 2022)* (Vol. 1, pp. 177-188). SCITEPRESS. https://doi.org/10.5220/0010908200003120
15. Geng, J., Wang, J., Fang, Z., Zhou, Y., Wu, D., & Ge, W. (2024). A survey of strategy-driven evasion methods for PE malware: Transformation, concealment, and attack. *Computers & Security, 137*, 103595. https://doi.org/10.1016/j.cose.2023.103595
16. Alkhateeb, E., Ghorbani, A., & Habibi Lashkari, A. (2024). A survey on run-time packers and mitigation techniques. *International Journal of Information Security, 23*(2), 887-913. https://doi.org/10.1007/s10207-023-00759-y
17. Alkhateeb, E., Ghorbani, A., & Habibi Lashkari, A. (2024). Identifying malware packers through multilayer feature engineering in static analysis. *Information, 15*(2), 102. https://doi.org/10.3390/info15020102
18. Kundisch, D., Muntermann, J., Oberländer, A. M., Rau, D., Röglinger, M., Schoormann, T., & Szopinski, D. (2022). An update for taxonomy designers: Methodological guidance from information systems research. *Business & Information Systems Engineering, 64*(4), 421-439. https://doi.org/10.1007/s12599-021-00723-x
19. De Sutter, B., Schrittwieser, S., Coppens, B., & Kochberger, P. (2025). Evaluation methodologies in software protection research. *ACM Computing Surveys, 57*(4), Article 86, 1-41. https://doi.org/10.1145/3702314
20. Muralidharan, T., Cohen, A., Gerson, N., & Nissim, N. (2022). File packing from the malware perspective: Techniques, analysis approaches, and directions for enhancements. *ACM Computing Surveys, 55*(5), Article 108, 1-45. https://doi.org/10.1145/3530810

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 1 (33), 2026
ISSN 2663 – 4023

Nazarii Chetvertukha
PhD Student, Department of Information Security
Lviv Polytechnic National University, Lviv, Ukraine
ORCID: 0009-0001-0944-2599
nazariy.chetvertukha@gmail.com

Viktor Otenko
PhD, Associate Professor, Department of Information Security
Lviv Polytechnic National University, Lviv, Ukraine
ORCID: 0000-0003-4781-7766
viktor.i.otenko@lpnu.ua

DEVELOPMENT OF A MULTIDIMENSIONAL FACETED TAXONOMY OF MALWARE OBFUSCATION TECHNIQUES

Abstract. The article addresses the problem of systematisation of malware obfuscation techniques, which is complicated by the structural limitations of existing classification schemes in the field and the absence of an integrated approach to describing classical obfuscation techniques and anti-analysis techniques. In this work, obfuscation is understood in a broad sense, encompassing both classical obfuscation techniques and anti-analysis techniques, united by the common purpose of concealing the malicious behaviour of a program from analytical tools. Classical and modern obfuscation taxonomies are analysed, among which common systemic limitations are identified, including the inability to describe multi-aspect techniques by several independent characteristics simultaneously, the absence of an integrated approach to classical obfuscation techniques and anti-analysis techniques, and the construction of existing schemes without the application of a formal taxonomy development method. The goal of the work is to develop a multidimensional faceted taxonomy of malware obfuscation techniques that eliminates the identified limitations. To achieve this goal, the taxonomy development method of Nickerson, Varshney and Muntermann is applied with alternation of empirical-to-conceptual and conceptual-to-empirical iterations. As a result, a faceted taxonomy with five independent dimensions has been formed, namely the level of observability, the scope of impact, the mechanism of transformation, the phase of effect manifestation, and the behavioural invariance. The proposed taxonomy is applied to 16 representative obfuscation techniques of contemporary malware. The behavioural invariance dimension is proposed separately; it operationally distinguishes classical obfuscation techniques from anti-analysis techniques and ensures the integration of these two directions of concealment techniques into a common classification framework while preserving the structural distinction between them. The proposed taxonomy provides a methodological basis for the systematisation of obfuscation techniques and the design of detection systems for obfuscated malware.

Keywords: malware obfuscation; faceted taxonomy; anti-analysis techniques; classification scheme; behavioural invariance; malware detection; Nickerson methodology; multidimensional classification; malware.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1. AV-TEST Institute. (2025). *Malware statistics & trends report (2024-2025)*. https://www.av-test.org/en/statistics/malware/
2. MITRE. (n.d.). *Obfuscated files or information (Technique T1027)*. MITRE ATT&CK® Framework. https://attack.mitre.org/techniques/T1027/
3. Brezinski, K., & Ferens, K. (2023). Metamorphic malware and obfuscation: A survey of techniques, variants, and generation kits. *Security and Communication Networks, 2023*, Article 8227751, 1-41. https://doi.org/10.1155/2023/8227751
4. Galloro, N., Polino, M., Carminati, M., Continella, A., & Zanero, S. (2022). A systematical and longitudinal study of evasive behaviors in Windows malware. *Computers & Security, 113*, 102550. https://doi.org/10.1016/j.cose.2021.102550
5. Collberg, C., Thomborson, C., & Low, D. (1997). *A taxonomy of obfuscating transformations* (Technical Report No. 148). Department of Computer Science, The University of Auckland. https://researchspace.auckland.ac.nz/handle/2292/3491

6. Afianian, A., Niksefat, S., Sadeghiyan, B., & Baptiste, D. (2019). Malware dynamic analysis evasion techniques: A survey. *ACM Computing Surveys, 52*(6), Article 126, 1-28. https://doi.org/10.1145/3365001
7. Nickerson, R. C., Varshney, U., & Muntermann, J. (2013). A method for taxonomy development and its application in information systems. *European Journal of Information Systems, 22*(3), 336-359. https://doi.org/10.1057/ejis.2012.26
8. You, I., & Yim, K. (2010). Malware obfuscation techniques: A brief survey. In *Proceedings of the 2010 International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA 2010)* (pp. 297-300). IEEE. https://doi.org/10.1109/BWCCA.2010.85
9. Opirskyi, I., Dzioban, T., & Vasylyshyn, S. (2025). Bypassing EDR in combination with SIEM: Analysis of methods for hiding attacks in logs – A study of tactics used by attackers to avoid detection. *Cybersecurity: Education, Science, Technique, 1*(29), 8-26. https://doi.org/10.28925/2663-4023.2025.29.865
10. Xu, H., Zhou, Y., Ming, J., & Lyu, M. (2020). Layered obfuscation: A taxonomy of software obfuscation techniques for layered security. *Cybersecurity, 3*, Article 9. https://doi.org/10.1186/s42400-020-00049-3
11. Aboaoja, F. A., Zainal, A., Ghaleb, F. A., Al-Rimy, B. A. S., Eisa, T. A. E., & Elnour, A. A. H. (2022). Malware detection issues, challenges, and future directions: A survey. *Applied Sciences, 12*(17), 8482. https://doi.org/10.3390/app12178482
12. Asghar, H. J., Zhao, B. Z. H., Ikram, M., Nguyen, G., Kaafar, D., Lamont, S., & Coscia, D. (2024). Use of cryptography in malware obfuscation. *Journal of Computer Virology and Hacking Techniques, 20*(1), 135-152. https://doi.org/10.1007/s11416-023-00504-y
13. Park, J., Jang, Y.-H., Hong, S., & Park, Y. (2019). Automatic detection and bypassing of anti-debugging techniques for Microsoft Windows environments. *Advances in Electrical and Computer Engineering, 19*(2), 23-28. https://doi.org/10.4316/AECE.2019.02003
14. Carrier, T., Victor, P., Tekeoglu, A., & Lashkari, A. H. (2022). Detecting obfuscated malware using memory feature engineering. In P. Mori, G. Lenzini, & S. Furnell (Eds.), *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP 2022)* (Vol. 1, pp. 177-188). SCITEPRESS. https://doi.org/10.5220/0010908200003120
15. Geng, J., Wang, J., Fang, Z., Zhou, Y., Wu, D., & Ge, W. (2024). A survey of strategy-driven evasion methods for PE malware: Transformation, concealment, and attack. *Computers & Security, 137*, 103595. https://doi.org/10.1016/j.cose.2023.103595
16. Alkhateeb, E., Ghorbani, A., & Habibi Lashkari, A. (2024). A survey on run-time packers and mitigation techniques. *International Journal of Information Security, 23*(2), 887-913. https://doi.org/10.1007/s10207-023-00759-y
17. Alkhateeb, E., Ghorbani, A., & Habibi Lashkari, A. (2024). Identifying malware packers through multilayer feature engineering in static analysis. *Information, 15*(2), 102. https://doi.org/10.3390/info15020102
18. Kundisch, D., Muntermann, J., Oberländer, A. M., Rau, D., Röglinger, M., Schoormann, T., & Szopinski, D. (2022). An update for taxonomy designers: Methodological guidance from information systems research. *Business & Information Systems Engineering, 64*(4), 421-439. https://doi.org/10.1007/s12599-021-00723-x
19. De Sutter, B., Schrittwieser, S., Coppens, B., & Kochberger, P. (2025). Evaluation methodologies in software protection research. *ACM Computing Surveys, 57*(4), Article 86, 1-41. https://doi.org/10.1145/3702314
20. Muralidharan, T., Cohen, A., Gerson, N., & Nissim, N. (2022). File packing from the malware perspective: Techniques, analysis approaches, and directions for enhancements. *ACM Computing Surveys, 55*(5), Article 108, 1-45. https://doi.org/10.1145/3530810

Отримано редакцією журналу / Received: 27.02.26
Прорецензовано / Revised: 10.03.26
Схвалено до друку / Accepted: 25.06.26