УДК 004.056:004.415.53

МЕТОДИ АДАПТИВНОЇ ПРІОРИТЕЗАЦІЇ РЕГРЕСІЙНОГО ТЕСТУВАННЯ В СЕРЕДОВИЩАХ DEVSECOPS З УРАХУВАННЯМ РИЗИКІВ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ОНЛАЙН-СЕРВІСІВ

Владислав Чіжов, Андрій Фесенко

Сучасні онлайн-сервіси, зокрема державні цифрові платформи, функціонують в умовах високої динамічності розробки та підвищених кіберзагроз, що обумовлює необхідність інтеграції безпекових аспектів у процес тестування програмного забезпечення. У роботі розглянуто проблему пріоритезації регресійного тестування в середовищах DevSecOps, де обмеження часу та ресурсів у CI/CD процесах унеможливлюють повне виконання тестових наборів.

Проведено аналіз сучасних методів пріоритезації регресійного тестування, зокрема підходів, заснованих на покритті коду, історії дефектів, результатах попередніх виконань тестів, оцінці ризиків, евристичних алгоритмах та методах машинного навчання.

Визначено їх сильні та слабкі сторони з точки зору ефективності виявлення дефектів, адаптивності, складності реалізації та придатності до використання в умовах CI/CD.

Особливу увагу приділено оцінці рівня врахування ризиків інформаційної безпеки в кожному з підходів. Встановлено, що більшість існуючих методів орієнтована на виявлення функціональних дефектів і не враховує безпекові фактори як самостійний критерій пріоритезації, що призводить до потенційного ігнорування критичних з точки зору безпеки компонентів системи.

На основі проведеного порівняльного аналізу виявлено ключові обмеження сучасних підходів, зокрема відсутність інтеграції з інструментами безпекового аналізу, орієнтацію на непрямі індикатори якості, недостатню адаптивність до змін профілю загроз та ігнорування контексту використання системи. Обґрунтовано необхідність розвитку адаптивних підходів до пріоритезації регресійного тестування, які забезпечують інтеграцію безпекових метрик у процес прийняття рішень.

Ключові слова: *регресійне тестування, пріоритезація тестів, DevSecOps, інформаційна безпека, CI/CD, Test Case Prioritization, ризик-орієнтоване тестування, SAST, DAST, вразливості, онлайн-сервіси.*

ВСТУП

Сучасний етап цифровізації в Україні характеризується активною розробкою програмного забезпечення для державних установ та онлайн-сервісів, які забезпечують взаємодію громадян з державою. Такі системи виконують функції критичної цифрової інфраструктури, оскільки обробляють персональні дані, фінансову інформацію та інші чутливі ресурси. Прикладом подібних рішень є платформа «Дія», яка об'єднує широкий спектр державних сервісів, включаючи механізми автентифікації користувачів, доступ до цифрових документів та реалізацію онлайн-послуг [17].

Характерною особливістю сучасних програмних систем є їхня висока динамічність, що обумовлена широким впровадженням практик безперервної інтеграції та доставки (CI/CD) [16]. У таких умовах розробка відбувається шляхом частих ітеративних змін замість рідкісних масштабних оновлень. Це створює необхідність швидкого виявлення дефектів і підтвердження стабільності системи після кожної зміни. Особливо критичним це є для онлайн-сервісів, які функціонують у відкритому середовищі та піддаються підвищеним кіберзагрозам [12].

За таких умов тестування програмного забезпечення набуває особливого значення як засіб своєчасного виявлення дефектів і підтвердження коректності роботи системи після внесення змін. На перший погляд, перевірка незначних модифікацій може здаватися відносно простою задачею, оскільки обсяг зміненого коду або функціональності є обмеженим. Однак у складних програмних системах окремі компоненти тісно взаємодіють між собою, тому зміни в одній частині системи можуть спричинити небажані наслідки в інших, на перший погляд не пов'язаних, модулях.

Саме тому важливим інструментом забезпечення стабільності програмного забезпечення є регресійне тестування, спрямоване на перевірку того, що вже реалізована функціональність продовжує працювати коректно після внесення змін, а нові модифікації не призвели до появи дефектів у раніше протестованих компонентах системи.

Водночас зі зростанням складності програмних систем і обсягів тестових наборів виникає проблема масштабованості регресійного тестування. Повне виконання всіх тестових сценаріїв у межах одного релізного циклу стає ресурсно витратним і часто неможливим у часових обмеженнях CI/CD. У

результаті виникає необхідність визначення порядку виконання тестів таким чином, щоб максимально швидко виявляти критичні дефекти.

Додатковим ускладненням є інтеграція вимог інформаційної безпеки у процес розробки програмного забезпечення, що реалізується в межах концепції DevSecOps. Для України це питання набуває особливого значення, оскільки державні цифрові сервіси функціонують в умовах підвищеного рівня кіберзагроз, пов'язаних як із загальними ризиками відкритих онлайн-систем, так і з цілеспрямованими атаками на критичну інфраструктуру. У таких умовах безпека перестає бути окремим етапом перевірки та інтегрується безпосередньо в процес розробки. Це означає, що тестування повинно забезпечувати не лише функціональну коректність системи, але й системне виявлення потенційних вразливостей, які можуть бути використані в умовах реальних атак.

Існуючі методи пріоритезації регресійного тестування здебільшого базуються на технічних характеристиках, таких як покриття коду, історія дефектів або ймовірність їх виявлення. Однак ці підходи не враховують рівень ризику інформаційної безпеки як окремий фактор, що призводить до ситуацій, коли критичні з точки зору безпеки компоненти системи можуть отримувати низький пріоритет у процесі тестування.

У таких умовах недостатня увага до безпекових аспектів регресійного тестування може призводити до невиявлення вразливостей, що створює ризики витоку даних, порушення роботи сервісів і втрати довіри користувачів. Таким чином, виникає необхідність у дослідженні підходів до пріоритезації регресійного тестування, які здатні враховувати як функціональні, так і безпекові аспекти якості програмного забезпечення в середовищах DevSecOps.

ПОСТАНОВКА ПРОБЛЕМИ

У середовищах безперервної інтеграції та доставки повне виконання регресійного тестування після кожної зміни є обмеженим у часі та ресурсах. Це зумовлює необхідність застосування методів пріоритезації тестових випадків, спрямованих на визначення порядку їх виконання з метою якнайшвидшого виявлення критичних дефектів.

Більшість існуючих підходів до пріоритезації базується на технічних характеристиках тестових випадків, зокрема покритті коду, історії дефектів або результатах попередніх запусків. Такі методи орієнтовані переважно на підвищення ефективності виявлення функціональних помилок і не враховують ризики інформаційної безпеки як самостійний критерій визначення пріоритету.

Унаслідок цього тестові випадки, пов'язані з компонентами, що обробляють чутливі дані або реалізують механізми автентифікації та авторизації, можуть виконуватися із запізненням. Це підвищує ризик несвоєчасного виявлення вразливостей, особливо в умовах DevSecOps, де безпекові перевірки мають бути інтегровані безпосередньо в цикл розробки.

Отже, проблема полягає у відсутності достатньо формалізованих підходів до пріоритезації регресійного тестування, які враховують не лише функціональні та технічні характеристики тестових випадків, але й рівень ризику інформаційної безпеки пов'язаних з ними компонентів системи. Це зумовлює необхідність дослідження підходу до пріоритезації регресійних тестів із урахуванням безпекових факторів у середовищах DevSecOps [13].

МЕТА СТАТТІ

Метою статті є систематизація та порівняльний аналіз підходів до пріоритезації регресійного тестування з погляду їх застосовності в середовищах DevSecOps, зокрема щодо можливості врахування ризиків інформаційної безпеки при визначенні порядку виконання тестових випадків та виявлення обмежень існуючих підходів у контексті сучасних онлайн-сервісів.

У межах дослідження передбачається здійснити класифікацію основних підходів до пріоритезації регресійного тестування, проаналізувати їх за ключовими характеристиками. Окрему увагу передбачається приділити оцінці рівня врахування ризиків інформаційної безпеки в кожному підході, а також провести їх порівняльний аналіз у контексті застосування в середовищах DevSecOps. На основі отриманих результатів планується визначити сильні сторони та обмеження існуючих методів, зокрема щодо їх застосування в онлайн-сервісах, що обробляють чутливі дані.

АНАЛІЗ ОСТАННІХ ДОСЛІДЖЕНЬ І ПУБЛІКАЦІЙ

Сучасний вектор досліджень у сфері регресійного тестування спрямований на вирішення проблеми масштабованості тестових наборів у високошвидкісних циклах розробки.

Ключова увага приділяється методам Test Case Prioritization (TCP), які дозволяють оптимізувати використання обмежених обчислювальних ресурсів. Класичні підходи, що базуються на максимальному покритті коду та аналізі історії дефектів, традиційно оцінюються за метрикою APFD (Average Percentage of Faults Detected). Проте в умовах DevSecOps ці методи стають недостатніми, оскільки не враховують специфіку цілеспрямованих атак на онлайн-сервіси [1-5].

Окремий масив наукових робіт присвячений risk-based підходам. У них пріоритезація базується на складності архітектури або критичності бізнес-логіки [7]. Втім, більшість існуючих моделей розглядають ризик крізь призму стабільності коду (ймовірність багу), ігноруючи ризики інформаційної безпеки (ймовірність експлуатації вразливості). Для онлайн-сервісів, що функціонують у відкритих мережевих середовищах, цей розрив є критичним, оскільки звичайна програмна помилка та безпекова вразливість мають різний рівень потенційних збитків.

Зі зростанням популярності парадигми Infrastructure as Code (IaC) та мікросервісної архітектури, з'являються дослідження адаптивних моделей на основі машинного навчання (ML) та підкріплювального навчання (Reinforcement Learning). Такі моделі здатні динамічно змінювати порядок тестів, реагуючи на зміни в коді. Однак, як зазначають дослідники, головною перешкодою для їх впровадження залишається «проблема холодного старту» та дефіцит структурованих наборів даних, що поєднують результати тестування з реальними векторами атак.

Інтеграція безпекових перевірок у середовища DevSecOps (через інструменти SAST, DAST та IAST) наразі здебільшого відбувається паралельно з основним регресійним тестуванням, що призводить до дублювання перевірок та затримок у CI/CD конвеєрах. Останні публікації акцентують увагу на необхідності створення risk-aware підходів, які б об'єднували дані про вразливості (CVE), бізнес-критичність сервісів та динамічний ландшафт загроз у єдину адаптивну модель [13,15].

Таким чином, проведений аналіз підтверджує наявність наукової прогалини: відсутність цілісних методів адаптивної пріоритезації, які б інтегрували ризики інформаційної безпеки безпосередньо в процес регресійного тестування. Це обумовлює необхідність розробки методів, здатних динамічно реагувати на появу нових загроз для онлайн-сервісів, мінімізуючи час до виявлення критичних вразливостей (MTTD).

ОСНОВНА ЧАСТИНА

Сучасні методи пріоритезації регресійного тестування формуються на основі різних джерел даних і підходів до оцінки ефективності тестів, що безпосередньо впливає на їх здатність забезпечувати швидке виявлення дефектів у середовищах із частими змінами. Вибір конкретного підходу визначає компроміс між швидкістю виконання тестів, точністю виявлення дефектів та можливістю врахування додаткових факторів, зокрема ризиків інформаційної безпеки.

Методи, засновані на покритті коду (coverage-based), орієнтовані на максимізацію покриття виконуваних елементів програми (statement, branch, path coverage). Їх ключова перевага полягає у простоті реалізації та високій швидкості виконання, що робить їх придатними для використання у CI/CD конвеєрах. Наприклад, у системах з великою кількістю модульних тестів ці підходи дозволяють швидко визначити тести, що покривають змінений код. Водночас їх головним недоліком є відсутність зв'язку між покриттям і критичністю дефектів: тест може покривати значну частину коду, але не перевіряти вразливі або бізнес-критичні сценарії.

Підходи, засновані на історії дефектів (fault-based), використовують інформацію про попередні помилки для визначення зон підвищеного ризику. Вони демонструють вищу ефективність виявлення дефектів порівняно з coverage-based методами, оскільки фокусуються на реально проблемних компонентах. Наприклад, якщо певний модуль регулярно містить дефекти, відповідні тести отримують вищий пріоритет. Проте ці методи мають обмеження: вони не враховують нові зміни або нові типи дефектів, що ще не проявлялися, а також повністю ігнорують безпекові ризики, якщо ті не були зафіксовані раніше [6].

History-based підходи розширюють fault-based логіку, використовуючи результати попередніх запусків тестів (pass/fail history). Їх перевага — адаптивність: тести, що часто падають, отримують вищий пріоритет. Це ефективно у великих системах з нестабільними компонентами. Однак ці методи схильні до локальної оптимізації: вони фокусуються на історичних проблемах і не здатні виявляти нові ризики, зокрема пов'язані з безпекою або новим функціоналом.

На відміну від coverage-based підходів, які орієнтовані на структуру коду, fault- та history-

based методи використовують емпіричні дані про поведінку системи, що підвищує їх ефективність у виявленні дефектів. Водночас, на відміну від AI/ML-підходів, вони не здатні узагальнювати нові залежності та залишаються обмеженими рамками історичних даних [11].

Risk-based підходи теоретично є найбільш релевантними для сучасних умов, оскільки враховують як ймовірність виникнення дефекту, так і його вплив. У практиці це може означати пріоритезацію тестів для критичних бізнес-функцій (наприклад, автентифікація або платіжні операції). Сильна сторона цих методів — орієнтація на бізнес-критичність [8]. Проте їх слабкість полягає у визначенні самого ризику: у більшості реалізацій він базується на суб'єктивних оцінках або функціональних характеристиках і не включає дані про вразливості, такі як результати SAST/DAST або CVSS-оцінки. Це суттєво обмежує їх застосування в DevSecOps [7,8].

Search-based та евристичні методи використовують алгоритми оптимізації (наприклад, генетичні алгоритми) для максимізації метрик ефективності тестування, таких як APFD. Вони здатні знаходити глобально оптимальні або близькі до оптимальних порядки виконання тестів. Наприклад, у складних системах з великою кількістю залежностей такі методи можуть значно підвищити швидкість виявлення дефектів. Однак їх використання обмежується високою обчислювальною складністю, що робить їх малопридатними для швидких CI/CD пайплайнів.

Методи, засновані на машинному навчанні (AI/ML-based), є найбільш гнучкими, оскільки дозволяють враховувати різнорідні дані: історію дефектів, зміни в коді, результати тестів тощо. Наприклад, модель може прогнозувати ймовірність падіння тесту на основі попередніх запусків і змін у коді [10]. Сильна сторона цих підходів — висока адаптивність і потенціал до врахування складних залежностей. Проте вони мають суттєві обмеження: потребують великого обсягу якісних даних, складні у впровадженні та інтерпретації, а також не гарантують врахування безпекових факторів без явної інтеграції відповідних даних.

Порівняльний аналіз методів пріоритезації регресійного тестування виконано на основі узагальнення результатів сучасних досліджень у сфері Test Case Prioritization [1-5], а також експертної оцінки їх характеристик за визначеними критеріями. Оцінювання має якісний характер і не базується на єдиному експериментальному дослідженні, а відображає типові властивості кожного класу методів, описані в науковій літературі.

Вибір критеріїв для порівняльного аналізу методів пріоритезації регресійного тестування обумовлений специфікою сучасних середовищ розробки, зокрема DevSecOps [14]. У таких умовах ефективність тестування визначається не лише здатністю швидко виявляти дефекти, але й необхідністю врахування ризиків інформаційної безпеки, адаптації до частих змін у системі, а також обмежень, пов'язаних із ресурсами та часом виконання тестів у CI/CD процесах [14]. Відповідно, у якості критеріїв порівняння обрано ефективність виявлення дефектів, рівень врахування безпеки, адаптивність, складність реалізації та придатність до CI/CD. Оцінка ефективності виявлення дефектів ґрунтується на узагальнених результатах досліджень, у яких застосовується метрика APFD (Average Percentage of Faults Detected).

Врахування безпекових аспектів оцінюється з точки зору використання методами відповідних джерел даних і сигналів безпеки, зокрема результатів статичного (SAST) і динамічного (DAST) аналізу, аналізу залежностей (SCA), оцінок вразливостей (наприклад, CVSS), а також моделей загроз. Низький рівень відповідає відсутності таких даних у процесі пріоритезації, середній — їх частковому або неформалізованому використанню, тоді як високий — системній інтеграції безпекових метрик у механізм прийняття рішень.

Адаптивність визначається здатністю методу змінювати порядок виконання тестів у відповідь на нові дані або зміни в системі. Складність реалізації оцінюється з урахуванням обчислювальних витрат, необхідності збору та обробки даних, а також рівня інтеграції в існуючі процеси розробки. Низька складність відповідає методам, що використовують вже доступні метрики (наприклад, покриття коду), середня — підходам, що потребують додаткових даних або налаштування, а висока — методам, які вимагають складної інфраструктури, значних обчислювальних ресурсів або застосування моделей машинного навчання.

Придатність для CI/CD визначається впливом методу на час виконання тестів і можливістю його застосування у високочастотних релізних циклах.

Такий підхід дозволяє здійснити систематизоване порівняння методів і виявити їх сильні та слабкі сторони з урахуванням вимог середовищ DevSecOps.

Порівняльний аналіз зазначених підходів наведено в Таблиці № 1.

МЕТОД	**КРИТЕРІЇ**				
	Ефективність виявлення дефектів	Врахування безпеки	Адаптивність	Складність реалізації	Придатність для CI/CD
Coverage-based	середня	низька	низька	низька	висока
Fault-based	висока	низька	середня	середня	висока
History-based	середня	низька	середня	низька	висока
Risk-based	висока	середня	середня	середня	середня
Search-based	висока	низька	низька	висока	низька
AI/ML-based	висока	потенційно висока	висока	висока	середня

Як видно з таблиці, методи, що базуються на структурних характеристиках (coverage-based), демонструють високу придатність до CI/CD за рахунок низької складності, проте поступаються іншим підходам у здатності враховувати критичність дефектів. Водночас підходи, що використовують історичні дані або моделі навчання, забезпечують вищу ефективність виявлення дефектів, але мають обмеження з точки зору інтеграції та ресурсних витрат.

Порівняльний аналіз показує, що кожен із розглянутих підходів оптимізує окремий аспект процесу тестування, що обумовлено природою використовуваних даних і метрик. Зокрема, coverage-based методи мінімізують час досягнення покриття зміненого коду, однак їх ефективність обмежується відсутністю кореляції між покриттям і ймовірністю виявлення критичних дефектів або вразливостей. Наприклад, тест, що покриває значну частину коду, може не включати перевірки сценаріїв, пов'язаних з автентифікацією або обробкою чутливих даних.

Fault-based та history-based підходи підвищують ймовірність виявлення дефектів за рахунок використання історичних даних, проте їх дія є ретроспективною. Вони ефективні лише в умовах стабільної архітектури та повторюваних типів помилок. У випадку нових змін або появи нових класів вразливостей (наприклад, пов'язаних із некоректною валідацією вхідних даних) такі методи не забезпечують належного покриття.

Risk-based підходи частково вирішують цю проблему, оскільки враховують вплив дефектів на систему або бізнес-процеси. Однак на практиці оцінка ризику часто базується на експертних припущеннях або функціональній критичності компонентів і не включає формалізовані метрики безпеки, такі як CVSS або результати аналізу вразливостей. Це призводить до ситуації, коли високоризикові з точки зору безпеки компоненти не отримують відповідного пріоритету.

Search-based методи демонструють високу ефективність з точки зору оптимізації глобальних метрик (наприклад, APFD), проте вони не враховують семантику ризиків, оскільки працюють на рівні формальних критеріїв оптимізації. У результаті оптимальний порядок виконання тестів не обов'язково відповідає реальному профілю загроз системи.

AI/ML-based підходи мають потенціал для подолання зазначених обмежень завдяки здатності обробляти різнорідні дані та виявляти складні залежності. Проте їх ефективність безпосередньо залежить від складу навчальних даних. У випадку відсутності безпекових сигналів (наприклад, результатів SAST/DAST або даних про вразливості) такі моделі фактично відтворюють поведінку history-based або fault-based підходів і не забезпечують врахування інформаційної безпеки.

Таким чином, жоден із розглянутих підходів у стандартному вигляді не забезпечує комплексного врахування факторів, необхідних для ефективного тестування в умовах DevSecOps. Основні обмеження сучасних методів можна деталізувати наступним чином:

• Відсутність інтеграції з безпековими джерелами даних. Більшість підходів не використовує результати статичного (SAST), динамічного (DAST) аналізу або аналізу залежностей (SCA). Унаслідок цього процес пріоритезації не враховує наявність відомих вразливостей, навіть якщо вони вже ідентифіковані інструментами безпеки.

• Орієнтація на історичні або структурні характеристики.

Методи, що базуються на покритті або історії дефектів, працюють із непрямими індикаторами якості, які не відображають реальний рівень ризику. Це призводить до систематичного недооцінювання нових або латентних вразливостей.

• Недостатня адаптивність до нових загроз.

Більшість моделей не враховує зміну профілю загроз у часі. Наприклад, поява нових типів атак або змін у залежностях системи не

впливає на пріоритезацію тестів, якщо ці фактори не відображені в історичних даних.

• Ігнорування контексту використання системи.

Пріоритезація не враховує, які саме компоненти системи є критичними з точки зору користувача або бізнесу (наприклад, обробка персональних даних або фінансові операції), що особливо важливо для державних цифрових сервісів, таких як «Дія».

У сукупності ці обмеження свідчать про те, що існуючі методи є ефективними лише в межах вузьких задач оптимізації регресійного тестування, орієнтованих переважно на функціональну якість. Вони не відповідають вимогам середовищ DevSecOps, де пріоритетним є врахування актуальних ризиків інформаційної безпеки. Це обумовлює необхідність подальших досліджень, спрямованих на інтеграцію безпекових метрик у процес пріоритезації тестування.

ВИСНОВКИ

Отже, в рамках дослідження проведено аналіз сучасних методів пріоритезації регресійного тестування в контексті їх застосування у середовищах DevSecOps. Встановлено, що існуючі підходи, зокрема coverage-based, fault-based, history-based, risk-based, search-based та AI/ML-based, орієнтовані переважно на оптимізацію виявлення функціональних дефектів і не забезпечують належного врахування ризиків інформаційної безпеки.

Порівняльний аналіз показав, що кожен із методів має сильні сторони у вирішенні окремих задач: підходи на основі покриття забезпечують швидкість і простоту інтеграції, fault- та history-based — підвищують ймовірність виявлення дефектів, risk-based — враховують бізнес-критичність, а AI/ML-підходи демонструють високий рівень адаптивності. Водночас жоден із них у класичному вигляді не інтегрує безпекові фактори як системоутворюючий елемент процесу пріоритезації.

Виявлено ключові обмеження сучасних підходів, зокрема відсутність інтеграції з інструментами безпекового аналізу (SAST, DAST, SCA), орієнтацію на історичні або структурні характеристики системи, недостатню адаптивність до змін профілю загроз та ігнорування контексту використання системи. Ці обмеження знижують ефективність регресійного тестування в умовах DevSecOps, де безпека є невід'ємною складовою процесу розробки.

Особливої актуальності зазначена проблема набуває для державних цифрових сервісів, таких як «Дія», які функціонують в умовах підвищених кіберзагроз та обробляють чутливі дані. У таких системах недостатнє врахування безпекових ризиків у процесі тестування може призводити до критичних наслідків.

Отже, результати дослідження підтверджують необхідність розвитку адаптивних методів пріоритезації регресійного тестування, які інтегрують оцінку ризиків інформаційної безпеки. Перспективним напрямом подальших досліджень є створення інтегрованих рішень для адаптивної пріоритезації регресійного тестування, що базуються на комбінованому використанні даних про вразливості, зміни в коді, історію дефектів та бізнес-критичність функціональності з метою підвищення ефективності тестування в умовах DevSecOps.

ЛІТЕРАТУРА

[1]. Rothermel G., Untch R. H., Chu C., Harrold M. J. Prioritizing test cases for regression testing. // IEEE Transactions on Software Engineering. — 2001. — Vol. 27, No. 10. — P. 929–948.
URL: https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1017&context=csearticles

[2]. Yoo S., Harman M. Regression testing minimization, selection and prioritization: A survey. // Software Testing, Verification and Reliability. — 2012. — Vol. 22, No. 2. — P. 67–120.
URL: http://www0.cs.ucl.ac.uk/staff/m.harman/stvr-shin-survey.pdf

[3]. Elbaum S., Malishevsky A. G., Rothermel G. Test case prioritization: A family of empirical studies. // IEEE Transactions on Software Engineering. — 2002. — Vol. 28, No. 2. — P. 159–182.
URL: https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1018&context=csearticles

[4]. Li Z., Harman M., Hierons R. Search algorithms for regression test case prioritization. // IEEE Transactions on Software Engineering. — 2007. — Vol. 33, No. 4. — P. 225–237.
URL:https://www.researchgate.net/publication/3189727_Search_Algorithms_for_Regression_Test_Case_Prioritization

[5]. Kim J.-M., Porter A. A history-based test prioritization technique for regression testing in resource constrained environments. // Proceedings of ICSE. — 2002. — P. 119–129.

URL: https://www.cs.umd.edu/users/aporter/Docs/icse2002.pdf

[6]. Srikanth H., Williams L. On the economics of requirements-based test case prioritization. // ACM SIGSOFT Software Engineering Notes. — 2005. — Vol. 30, No. 4. URL: https://dl.acm.org/doi/10.1145/1082983.1083100

[7]. Stallbaum H., Metzger A., Pohl K. An automated technique for risk-based test case prioritization. // Proceedings of ICSE. — 2008. — P. 105–114.

[8]. Felderer M., Herrmann A. Risk-based testing: A systematic literature review. // Software Quality Journal. — 2015. — Vol. 23. — P. 291–332. URL: https://www.researchgate.net/publication/282182677_Risk-Based_Testing

[9]. Scandariato R., Walden J., Hovsepyan A., Joosen W. Predicting vulnerable software components via text mining. // IEEE Transactions on Software Engineering. — 2014. — Vol. 40, No. 10. — P. 993–1006.

[10]. Sharma S., Kumar B. Artificial Intelligence in Software Testing: A Comprehensive Review of Machine Learning Approaches // Procedia Computer Science. — 2025. — Vol. 254. — P. 432–439. URL: https://www.sciencedirect.com/science/article/pii/S1877050925017624

[11]. Sagar S., Saha D. A systematic review of machine learning applications in software testing. // Journal of Systems and Software. — 2017.

[12]. OWASP Foundation. OWASP Top 10 – The Ten Most Critical Web Application Security Risks. — 2021.

[13]. Behl A., Behl K. DevSecOps: Integrating security into DevOps. — Springer, 2020.

[14]. Humble J., Farley D. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. — Addison-Wesley, 2010.

[15]. Myrbakken H., Colomo-Palacios R. DevSecOps: A multivocal literature review. // Future Generation Computer Systems. — 2017.

[16]. Shahin, M., Babar, M. A., & Zhu, L. (2017). Continuous integration, delivery and deployment: A systematic review on approaches, tools, challenges and practices. IEEE Access, 5, 3909–3943. URL: https://doi.org/10.1109/ACCESS.2017.2685629

[17]. Міністерство цифрової трансформації України. Дія — державні онлайн-сервіси. URL: https://diia.gov.ua

METHODS FOR ADAPTIVE PRIORITIZATION OF REGRESSION TESTING IN DEVSECOPS ENVIRONMENTS CONSIDERING INFORMATION SECURITY RISKS OF ONLINE SERVICES

Modern online services, particularly government digital platforms, operate under conditions of high development dynamics and increased cyber threats, which necessitates the integration of security considerations into the software testing process. This paper addresses the problem of regression test prioritization in DevSecOps environments, where time and resource constraints within CI/CD pipelines make it infeasible to execute the entire test suite.

An analysis of contemporary regression test prioritization methods is conducted, including approaches based on code coverage, defect history, previous test execution results, risk assessment, heuristic algorithms, and machine learning techniques. Their strengths and limitations are identified in terms of fault detection effectiveness, adaptability, implementation complexity, and suitability for CI/CD environments.

Particular attention is given to evaluating the extent to which information security risks are considered within each approach. It is established that most existing methods primarily focus on the detection of functional defects and do not incorporate security factors as an independent prioritization criterion, which may result in overlooking components critical from a security perspective.

Based on the comparative analysis, key limitations of current approaches are identified, including the lack of integration with security analysis tools, reliance on indirect quality indicators, insufficient adaptability to evolving threat landscapes, and neglect of system usage context. The necessity of developing adaptive regression test prioritization approaches that integrate security metrics into the decision-making process is substantiated.

Keywords: regression testing, test prioritization, DevSecOps, information security, CI/CD, Test Case Prioritization, risk-based testing, SAST, DAST, vulnerabilities, online services.

Чіжов Владислав Вадимович
ТОВ «Дрімскейп Нетворкс»
Інженер із забезпечення якості
E-mail: vladyslav.chizhov@gmail.com
Orcid ID: 0009-0002-6646-904X.
Chizhov Vladyslav
DREAMSCAPE NETWORKS LLC
General QA Engineer

Фесенко Андрій Олексійович,
Кандидат технічних наук, доцент,
декан факультету
комп'ютерних наук та технологій,
Державний університет «Київський авіаційний інститут»,
м. Київ, Україна.
Andriy Fesenko,
PhD in Technical Sciences,
Associate Professor, Dean of the Faculty of
Computer Science and Technologies
State university «Kyiv
aviation institute», Kyiv, Ukraine.
E-mail: aafesenko88@gmail.com.
ORCID ID: 0000-0001-5154-5324.

Андрій Фесенко є відповідальним секретарем фахового видання «Захист інформації», тому не брав участі у рецензуванні та прийнятті рішення щодо публікації цієї статті.

Автори заявляють про відсутність конфлікту інтересів. Спонсори не брали участі в розробленні дослідження; у зборі, аналізі чи інтерпретації даних; у написанні рукопису; у рішенні про публікацію результатів.

Andriy Fesenko is the Executive Secretary of the professional publication «Ukrainian Information Security Research Journal», and therefore she did not participate in the review and decision-making process regarding the publication of this article.

The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in writing of the manuscript; in the decision to publish the results.