and proposes a mechanism for assessing the degree of defect membership in the corresponding groups using elements of fuzzy set theory. The practical testing of the method was performed on the example of grouping software defects after a cyberattack by such features as the level of data integrity damage, system failures, and security system violations. The results of the study confirmed the effectiveness of the proposed approach for the automated formation of groups of critical and minor defects, as well as the possibility of detecting defects with intermediate characteristics. It was established that the use of the "dynamic kernels" method and fuzzy grouping allows to increase the efficiency of the processes of identification, analysis, prediction, and recovery of damaged software in conditions of information uncertainty and changes in the characteristics of cyberattacks.

Keywords: grouping, method, software, defect, mathematical model, software recovery.

Надійшла до редакції (Received): 25.07.2026
Прийнята до друку (Accepted): 08.09.2026
Опубліковано онлайн (Available online): 15.09.2026

УДК 004.056:004.415.53:004.89 DOI: 10.31673/2409-7292.2026.035907

Максимович Максим Віталійович
аспірант кафедри безпеки інформаційних технологій
Національний університет «Львівська політехніка», Львів, Україна
ORCID: 0009-0008-3604-8424
E-mail: maksym.v.maksymovych@lpnu.ua

МЕТОД КОНТЕКСТНО-КЕРОВАНОГО ДИНАМІЧНОГО ТЕСТУВАННЯ БЕЗПЕКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВИКОРИСТАННЯМ АГЕНТІВ НА ОСНОВІ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

Розвиток методів штучного інтелекту, зокрема великих мовних моделей та побудованих на їх основі агентних систем, відкриває нові можливості для вдосконалення методів автоматизованого тестування безпеки програмного забезпечення (ПЗ). Серед методів динамічне тестування безпеки (DAST) є найперспективнішим напрямом для вдосконалення, оскільки воно здійснює наскрізну перевірку реального працюючого застосунку та відображає його фактичну експлуатовану поведінку. Водночас класичні DAST-інструменти формують сценарії атаки шляхом випадкового перебору, що зумовлює неповне покриття та є доволі ресурсозатратним. Метою роботи є розроблення методу контекстно-керованого динамічного тестування безпеки ПЗ на основі агентів, побудованих на великих мовних моделях, що дозволяє підвищити повноту виявлення вразливостей та знизити ресурсозатратність сканування. Запропонований метод використовує специфікацію OpenAPI як джерело семантичного контексту, на основі якого агенти генерують набір наскрізних (E2E) тестів безпеки, що виступають детермінованим джерелом трафіку та скеровують динамічне сканування реальними бізнес-процесами замість випадкового обходу. Метод забезпечує два покращення: розширення покриття на вразливості, недосяжні для сигнатурних сканерів, та скорочення обсягу надлишкових запитів завдяки перевірці лише згенерованого трафіку. Описано модель покриття вразливостей, робочий процес та стратегію формування тестових сценаріїв, а також визначено метрики оцінювання ефективності методу. Отримані результати свідчать про доцільність агентного підходу для розширення покриття та підвищення ресурсоефективності динамічного тестування і окреслюють напрям подальшого експериментального дослідження.

Ключові слова: динамічне тестування безпеки, великі мовні моделі, агентні системи, OpenAPI, BOLA, IDOR, автоматизація тестування.

Вступ

У сучасних умовах стрімкої цифровізації та зростання кількості кібератак питання захисту програмного забезпечення (ПЗ) набуває особливої актуальності [1]. Збільшення складності програмних систем, поширення веб застосунків і програмних інтерфейсів (API), а також скорочення циклів розробки зумовлюють потребу в автоматизованих засобах виявлення вразливостей, здатних інтегруватися безпосередньо у процес неперервної інтеграції та розгортання (CI/CD). Паралельно стрімкий розвиток методів штучного інтелекту, зокрема великих мовних моделей (LLM) та побудованих на їх основі агентних систем, відкриває нові

© Добришин Ю.Є. Метод групування дефектів пошкодженого програмного забезпечення у технологічні подібні групи. Сучасний захист інформації, 3(67), 76–84.
https://doi.org/10.31673/2409-7292.2026.039402

можливості для якісного вдосконалення усталених методів автоматизованого тестування безпеки [2]. Саме поєднання цих методів із наявними підходами до перевірки захищеності ПЗ і визначає актуальність даного дослідження.

Автоматизоване тестування безпеки представлено кількома основними типами аналізу: статичним (SAST), динамічним (DAST) та композитним (SCA). На відміну від статичного та композитного аналізу, які працюють відповідно з вихідним кодом та сторонніми залежностями, динамічне тестування безпеки здійснює перевірку реального працюючого застосунку ззовні, у процесі його виконання. Це означає, що підтверджені під час сканування знахідки відображають фактичну експлуатовану поведінку системи, а сам метод не залежить від мови програмування та природно вбудовується у конвеєр CI/CD. Враховуючи вищесказане, можна зробити висновок, що DAST є найперспективнішим напрямом для вдосконалення засобами штучного інтелекту, оскільки саме він оперує наскрізною (E2E) взаємодією із застосунком, наближеною до реальних сценаріїв використання.

Водночас практичне застосування класичних DAST-інструментів супроводжується низкою суттєвих обмежень. Передусім формування наборів даних для атаки в таких інструментах здійснюється переважно шляхом перебору з використанням заздалегідь визначених словників, що зумовлює неповне та нецілеспрямоване покриття застосунку [3]. Окрім неповноти, такий перебір є ресурсозатратним: він породжує велику кількість запитів, значна частина яких не відповідає наявній функціональності, що збільшує тривалість сканування. Як наслідок, частина функціональності, доступної лише за певної послідовності дій, може залишатися поза межами сканування, що знижує повноту виявлення вразливостей.

Окремою проблемою є нечутливість класичного динамічного тестування до вразливостей бізнес-логіки, пов'язаних із контролем доступу. Сигнатурні сканери орієнтовані передусім на виявлення технічних вразливостей, що мають характерні ознаки у відповідях застосунку, проте вони не оперують семантикою контракту API та не враховують взаємозв’язок між елементами програмного забезпечення [4]. Унаслідок цього вразливості порушення авторизації на рівні об'єктів залишаються поза увагою, попри їхню високу критичність [5].

Таким чином, можна зробити висновок, що ключові обмеження класичного DAST – неповнота та ресурсозатратність нецілеспрямованого перебору, а також відсутність семантичного контексту авторизації - є саме тими задачами, які потенційно здатні вирішити агентні системи на основі великих мовних моделей. Здатність таких систем аналізувати специфікацію застосунку, планувати послідовність дій та генерувати цілеспрямовані тестові сценарії дозволяє припустити доцільність їх застосування для скерування динамічного тестування контекстно-насиченим трафіком, що й формує проблематику даної роботи.

Аналіз літератури

Питання ефективності динамічного тестування безпеки досліджувалося у працях, присвячених порівняльному оцінюванню інструментів. У систематичному огляді [6] показано, що більшість сканерів веб застосунків оцінювалася лише на впровадженні SQL-коду (SQLi) та міжсайтовому скриптингу (XSS), що свідчить про їх орієнтацію на технічні вразливості з характерними сигнатурами. За результатами порівняльного дослідження [1], найкращі результати серед засобів динамічного аналізу демонструє OWASP ZAP, проте й він не виявляє вразливостей окремих категорій. Узагальнюючи обмеження сканерів чорних скриньок, у роботі [4] виокремлено високі рівні хибнопозитивних та хибнонегативних результатів і складність виявлення вразливостей бізнес-логіки, а дослідження [3] підтверджує неповноту стандартного перебору. Для зниження частки хибних спрацьовувань застосовують методи машинного навчання [7].

Окремий напрям становить застосування великих мовних моделей до виявлення вразливостей. У систематичному огляді [2] відзначено перехід від моделей, що виконують поодинокі завдання, до автономних агентів, здатних оркеструвати багатокрокові процеси. Експериментальні дослідження підтверджують потенціал таких моделей: за даними [8], LLM

© Максимович М.В. Метод контекстно-керованого динамічного тестування безпеки програмного забезпечення з використанням агентів на основі великих мовних моделей. Сучасний захист інформації, 3(67), 84–91. https://doi.org/10.31673/2409-7292.2026.035907

досягають середньої точності 62,8% та F1-міри 0,71, перевищуючи окремі засоби статичного аналізу на певних класах вразливостей; подібно, у дослідженні [9] мовні моделі продемонстрували вищу повноту, ніж засоби статичного аналізу, проте з більшою кількістю хибнопозитивних результатів, що обґрунтовує доцільність гібридних конвеєрів.

Поєднання мовних моделей з агентними системами активно досліджується у задачах автоматизованого тестування на проникнення: запропоновано як інтелектуальних помічників із замкненим циклом зворотного зв'язку [10], так і модульні [11] та багатоагентні [12] архітектури, що автоматизують збір інформації, аналіз вразливостей і експлуатацію. Стосовно вразливостей контролю доступу, їх виявлення досліджувалося на основі обробки специфікацій OpenAPI [13], формального моделювання REST API зі специфікації [14] та статичного аналізу коду застосунку [15]. Найближчими до тематики роботи є дослідження [16] та [17], у яких великі мовні моделі застосовуються для генерації тестових сценаріїв безпеки API, орієнтованих на BOLA, із використанням генерації, доповненої пошуком (RAG), та врахуванням контексту ендпоінтів. Водночас ці підходи зосереджені на генерації окремих тестових випадків, тоді як використання згенерованого агентами наскрізного трафіку як детермінованого джерела для динамічного сканування залишається малодослідженим, що й визначає напрям роботи.

Мета та завдання дослідження

Метою роботи є розроблення методу контекстно-керованого динамічного тестування безпеки програмного забезпечення на основі агентів, побудованих на великих мовних моделях, що дозволяє підвищити повноту виявлення вразливостей контролю доступу та знизити ресурсозатратність сканування порівняно з класичним динамічним тестуванням. Для досягнення поставленої мети визначено такі завдання:

- проаналізувати обмеження класичного динамічного тестування безпеки;
- запропонувати концепцію методу контекстно-керованого тестування на основі агентів та специфікації OpenAPI;
- описати робочий процес методу та його переваги;
- визначити метрики оцінювання ефективності методу.

Основна частина

Динамічне тестування як напрям для вдосконалення засобами штучного інтелекту

Вибір динамічного тестування безпеки як об'єкта вдосконалення зумовлений принциповими відмінностями цього методу від інших типів автоматизованого аналізу. Статичний та композитний аналіз працюють відповідно з вихідним кодом та переліком сторонніх залежностей, тобто оперують артефактами розробки, а не поведінкою системи. Натомість динамічне тестування здійснює перевірку реального працюючого застосунку ззовні, у процесі його виконання, що забезпечує дві суттєві переваги.

По-перше, підтверджені під час сканування вразливості відображають фактичну експлуатовану поведінку системи, а отже мають нижчу частку хибнопозитивних результатів порівняно зі статичним аналізом, який, як зазначається у дослідженні [9], характеризується нижчою точністю через відсутність контексту виконання. По-друге, метод не залежить від мови програмування та технологічного стеку, що спрощує його інтеграцію у конвеєр неперервної інтеграції та доставлення.

Саме наскрізний характер динамічного тестування, за якого застосунок перевіряється у режимі, наближеному до реальних сценаріїв використання, робить цей метод найбільш підходящим для вдосконалення засобами штучного інтелекту. Водночас практична ефективність класичних DAST-інструментів обмежується двома чинниками. Перший – формування даних для атаки шляхом стохастичного перебору адрес, що, як показано у роботі [3], забезпечує неповне покриття застосунку та є ресурсозатратним через велику кількість надлишкових запитів. Другий – нездатність виявляти вразливості бізнес-логіки, оскільки сигнатурні сканери не оперують семантикою контракту API [5]. Подолання саме цих обмежень і визначає зміст запропонованого методу: здатність агентів на основі великих

© Максимович М.В. Метод контекстно-керованого динамічного тестування безпеки програмного забезпечення з використанням агентів на основі великих мовних моделей. Сучасний захист інформації, 3(67), 84–91. https://doi.org/10.31673/2409-7292.2026.035907

мовних моделей аналізувати специфікацію застосунку та планувати цілеспрямовані дії дозволяє замінити випадковий обхід контекстно-насиченим трафіком.

Концепція методу контекстно-керованого динамічного тестування

В основу методу покладено використання специфікації OpenAPI [18] як джерела семантичного контексту застосунку. Специфікація містить структурований опис точок доступу, параметрів запитів, схем автентифікації та зв'язків між об'єктами, що дозволяє сформувати цілеспрямовані тестові сценарії замість випадкового перебору. На основі цього контексту агенти, побудовані на великих мовних моделях, генерують набір наскрізних (E2E) тестів безпеки, які відтворюють послідовності дій, характерні для реальних бізнес-процесів. Згенеровані тести виступають детермінованим джерелом трафіку: вони керують динамічним сканером, скеровуючи його до функціональності, що відповідає визначеним бізнес-потокам, а не до випадково виявлених адрес. Принциповим обмеженням методу є те, що агенти аналізують виключно специфікацію та її зміни, не отримуючи доступу до вихідного коду застосунку, що зберігає незалежність динамічного тестування від технологічного стеку.

Порівняно з класичним динамічним тестуванням метод забезпечує два покращення. Перше – розширення покриття на вразливості контролю доступу, насамперед порушення авторизації на рівні об'єктів, недосяжні для сигнатурних сканерів. Друге – підвищення ресурсоефективності: класичний стохастичний перебір адрес за словником породжує велику кількість запитів, значна частина яких не відповідає реальній функціональності застосунку, тоді як скерування сканування лише згенерованим трафіком наскрізних тестів зосереджує перевірку на дійсних бізнес-потоках і суттєво скорочує обсяг надлишкових запитів.

Для формалізації відмінності між класичним та агентно-орієнтованим підходами введено модель покриття вразливостей. Нехай V – множина всіх вразливостей застосунку, V_{DAST} – підмножина вразливостей, які виявляє класичний динамічний сканер, а V_{agent} – підмножина вразливостей, що виявляються запропонованим методом. Сукупне покриття визначається як об'єднання цих підмножин (1):

$$V_{cover} = V_{DAST} \cup V_{agent}, \tag{1}$$

де V_{DAST} охоплює переважно технічні вразливості з характерними сигнатурами, а V_{agent} додатково містить вразливості контролю доступу. Оскільки вразливості порушення авторизації на рівні об'єктів не виявляються сигнатурними сканерами [5], вони належать до V_{agent} і лишаються поза межами V_{DAST}, що формалізує ключову тезу методу: агентний підхід розширює покриття динамічного тестування на клас вразливостей, недосяжний для класичних засобів.

Альтернативою запропонованому методу є пряме сканування на основі специфікації OpenAPI, за якого сканеру передають специфікацію та налаштовану схему авторизації, а він самостійно перебирає описані точки доступу. Такий підхід має суттєві недоліки. По-перше, він не дозволяє обмежити перевірку лише для поточних змін і щоразу опрацьовує специфікацію повністю, що знову призводить до надлишкових витрат. По-друге, точки доступу перевіряються ізольовано, тоді як значна їх частина залежить від стану, попередньо створеного іншими елементами системи: щоб перевірити операцію над об'єктом, його спершу потрібно створити відповідним запитом. По-третє, такий підхід не враховує залежностей та послідовності викликів між елементами, через що запити часто завершуються помилками, не задіюючи реальну логіку застосунку. Унаслідок цього покриття багатокрокових бізнес-потоків залишається низьким, а вразливості контролю доступу, виявлення яких потребує створення об'єкта одним користувачем та спроби доступу до нього іншим, лишаються невиявленими.

На відміну від цього, скерування сканування трафіком наскрізних тестів усуває зазначені обмеження: згенеровані тести відтворюють реальні послідовності дій із дотриманням залежностей між викликами та формують коректний автентифікований трафік, що задіює фактичну бізнес-логіку застосунку. Крім того, запуск процесу за змінами у специфікації

© Максимович М.В. Метод контекстно-керованого динамічного тестування безпеки програмного забезпечення з використанням агентів на основі великих мовних моделей. Сучасний захист інформації, 3(67), 84–91. https://doi.org/10.31673/2409-7292.2026.035907

(openapi-diff) дозволяє генерувати тести лише для оновленої функціональності, забезпечуючи інкрементність перевірки.

Загальний робочий процес методу

Запропонований метод реалізується як послідовність етапів, інтегрована у конвеєр неперервної інтеграції та доставлення (рис. 1). Запуск процесу ініціюється зміною у специфікації API, що фіксується як відмінності між поточною та попередньою версіями специфікації (openapi-diff). Виявлена зміна активує етап генерації, на якому агенти на основі великих мовних моделей, спираючись на семантичний контекст специфікації, формують набір наскрізних тестів безпеки. Сформовані тести виконуються щодо реального працюючого застосунку; у разі невдачі хоча б одного тесту процес припиняється, і подальші етапи не виконуються. Лише після успішного проходження всіх тестів перехоплений під час їх виконання трафік передається динамічному сканеру, розгорнутому у режимі сервісу, який виконує активне сканування застосунку. Результат сканування визначає підсумок тестування, реалізуючи функцію контрольної точки безпеки. Як динамічний сканер використовується OWASP ZAP [19] – відкритий інструмент, що підтримує фоновий режим та інтеграцію з автоматизованими конвеєрами.

Рис. 1. Загальний робочий процес методу контекстно-керованого динамічного тестування

Стратегія формування тестових сценаріїв

Ключовою особливістю методу є цілеспрямованість формування тестових сценаріїв на виявлення вразливостей контролю доступу, насамперед порушень авторизації на рівні об'єктів (BOLA) та небезпечних прямих посилань на об'єкти (IDOR). Згідно з OWASP API Security Top 10, BOLA є найкритичнішим ризиком безпеки програмних інтерфейсів і не виявляється засобами автоматизованого статичного чи динамічного тестування через відсутність розуміння контексту авторизації [5]. Виявлення таких вразливостей потребує перевірки відносин володіння об'єктами між різними користувачами, що неможливо реалізувати засобами сигнатурного аналізу. Запропонована стратегія базується на генерації тестових сценаріїв безпосередньо із семантики специфікації API. На основі опису точок доступу API, що приймають ідентифікатори об'єктів, та визначених схем автентифікації формуються сценарії, у яких від імені одного користувача здійснюється спроба доступу до об'єктів, що належать іншому користувачеві, а також спроби виконання дій, які потребують певних прав.

Метрики оцінювання та апробація

Для оцінювання ефективності методу можуть бути застосовані стандартні метрики виявлення вразливостей: повнота (TPR, true positive rate), частка хибнопозитивних результатів

(FPR, false positive rate), точність (Precision), повнота виявлення (Recall) та узагальнювальна F1-міра, а також час отримання результату (TTR, time-to-result) і кількість запитів на одну виявлену вразливість як показник ресурсоефективності. Базовою метрикою повноти є співвідношення (2):

$$TPR = TP / (TP + FN) , \quad (2)$$

де TP (true positives) – кількість справжніх вразливостей, виявлених методом, FN (false negatives) – кількість наявних вразливостей, що залишилися невиявленими. Зазначені метрики дозволяють кількісно зіставити запропонований метод із класичним динамічним тестуванням за повнотою покриття, рівнем хибних спрацьовувань та ресурсозатратністю.

Попередня апробація методу на навмисно вразливому застосунку підтвердила його здатність виявляти вразливості контролю доступу, зокрема сценарії BOLA, що не виявляються класичним динамічним скануванням. Повноцінне кількісне експериментальне дослідження, спрямоване на порівняння запропонованого методу з класичним динамічним тестуванням, є предметом подальшої роботи.

Висновки

Розвиток методів штучного інтелекту, зокрема великих мовних моделей та агентних систем, відкриває нові можливості для вдосконалення усталених методів автоматизованого тестування безпеки програмного забезпечення. У роботі обґрунтовано, що динамічне тестування безпеки є найперспективнішим напрямом для такого вдосконалення, оскільки воно здійснює наскрізну перевірку реального працюючого застосунку та відображає його фактичну поведінку. Запропоновано метод контекстно-керованого динамічного тестування на основі агентів, побудованих на великих мовних моделях, у якому специфікація OpenAPI слугує джерелом семантичного контексту, а згенеровані наскрізні тести безпеки виступають детермінованим джерелом трафіку, що скеровує динамічне сканування. Сформовано модель покриття вразливостей, яка формалізує розширення можливостей динамічного тестування на клас вразливостей контролю доступу, недосяжний для сигнатурних сканерів.

Запропонований метод забезпечує два взаємодоповнювальні покращення порівняно з класичним динамічним тестуванням: розширення покриття на вразливості контролю доступу (BOLA, IDOR) та підвищення ресурсоефективності за рахунок перевірки лише цілеспрямовано згенерованого трафіку замість стохастичного перебору. Перспективним напрямом подальших досліджень є кількісне експериментальне порівняння запропонованого методу з класичним динамічним тестуванням за повнотою виявлення та ресурсозатратністю, а також розширення стратегії формування тестових сценаріїв на інші категорії вразливостей бізнес-логіки.

Перелік посилань

1. Qadir, S., Waheed, E., Khanum, A., & Jehan, S. (2025). Comparative evaluation of approaches & tools for effective security testing of Web applications. PeerJ Computer Science, 11, e2821. https://doi.org/10.7717/peerj-cs.2821.

2. Xu, H., Wang, S., Li, N., Wang, K., Zhao, Y., Chen, K., Yu, T., Liu, Y., & Wang, H. (2025). Large language models for cyber security: A systematic literature review. ACM Transactions on Software Engineering and Methodology. https://doi.org/10.1145/3769676.

3. Eriksson, B., Pellegrino, G., & Sabelfeld, A. (2021). Black Widow: Blackbox data-driven web scanning. In 2021 IEEE Symposium on Security and Privacy (SP) (pp. 1125–1142). IEEE. https://doi.org/10.1109/ SP40001. 2021.00022.

4. Xue, J., He, Y., Xiao, Y., Chen, Y., Wang, Z., & Lu, H. (2025). From heuristics to intelligence: Evolving black-box scanners for modern web applications. In 2025 IEEE 10th International Conference on Data Science in Cyberspace (DSC) (pp. 247–254). IEEE. https://doi.org/10.1109/DSC67331.2025.00039.

5. OWASP Foundation. (2023). API1:2023 Broken object level authorization. OWASP API Security Top 10 – 2023. https://owasp.org/API-Security/editions/2023/en/0xa1-broken-object-level-authorization/.

6. Alazmi, S., & De Leon, D. C. (2022). A systematic literature review on the characteristics and effectiveness of web application vulnerability scanners. IEEE Access, 10, 33200–33219. https://doi.org/10.1109/ACCESS.2022.3161522.

© Максимович М.В. Метод контекстно-керованого динамічного тестування безпеки програмного забезпечення з використанням агентів на основі великих мовних моделей. Сучасний захист інформації, 3(67), 84–91. https://doi.org/10.31673/2409-7292.2026.035907

7. Millar, S., Podgurskii, D., Kuykendall, D., Martínez del Rincón, J., & Miller, P. (2022). Optimising vulnerability triage in DAST with deep learning. In Proceedings of the 15th ACM Workshop on Artificial Intelligence and Security (pp. 137–147). ACM. https://doi.org/10.1145/3560830.3563724.

8. Khare, A., Dutta, S., Li, Z., Solko-Breslin, A., Alur, R., & Naik, M. (2025). Understanding the effectiveness of large language models in detecting security vulnerabilities. In 2025 IEEE Conference on Software Testing, Verification and Validation (ICST) (pp. 103–114). IEEE. https://doi.org/10.1109/ICST62969.2025.10988968.

9. Gnieciak, D., & Szandała, T. (2025). Large language models versus static code analysis tools: A systematic benchmark for vulnerability detection. IEEE Access, 13, 198410–198422. https://doi.org/10.1109/ ACCESS.2025. 3635168.

10. Happe, A., & Cito, J. (2023). Getting pwn'd by AI: Penetration testing with large language models. In Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 2082–2086). ACM. https://doi.org/10.1145/3611643.3613083.

11. Deng, G., Liu, Y., Mayoral-Vilches, V., Liu, P., Li, Y., Xu, Y., Zhang, T., Liu, Y., Pinzger, M., & Rass, S. (2024). PentestGPT: Evaluating and harnessing large language models for automated penetration testing. In Proceedings of the 33rd USENIX Security Symposium (pp. 847–864). USENIX Association. https://www.usenix.org/conference/ usenixsecurity24/presentation/deng.

12. Shen, X., Wang, L., Li, Z., Chen, Y., Zhao, W., Sun, D., Wang, J., & Ruan, W. (2025). PentestAgent: Incorporating LLM agents to automated penetration testing. In Proceedings of the 20th ACM Asia Conference on Computer and Communications Security (pp. 375–391). ACM. https://doi.org/10.1145/3708821.3733882.

13. Barabanov, A., Dergunov, D., Makrushin, D., & Teplov, A. (2022). Automatic detection of access control vulnerabilities via API specification processing. arXiv. https://doi.org/10.48550/arXiv.2201.10833.

14. Santos Filho, A., Rodríguez, R. J., & Feitosa, E. L. (2025). Automated broken object-level authorization attack detection in REST APIs through OpenAPI to colored Petri nets transformation. International Journal of Information Security, 24(2), 83. https://doi.org/10.1007/s10207-024-00970-5.

15. Huang, Y., Shi, C., Lu, J., Li, H., Meng, H., & Li, L. (2024). Detecting broken object-level authorization vulnerabilities in database-backed applications. In Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (pp. 2934–2948). ACM. https://doi.org/10.1145/3658644.3690227.

16. Pasca, E. M., Delinschi, D., Erdei, R., & Matei, O. (2025). LLM-driven, self-improving framework for security test automation: Leveraging Karate DSL for augmented API resilience. IEEE Access, 13, 56861–56886. https://doi.org/10.1109/ACCESS.2025.3554960.

17. Pasca, E. M., Erdei, R., Delinschi, D., & Matei, O. (2024). Enhancing API security testing against BOLA and authentication vulnerabilities through an LLM-enhanced framework. In 19th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2024) (Lecture Notes in Networks and Systems, pp. 231–240). Springer. https://doi.org/10.1007/978-3-031-75010-6_23.

18. OpenAPI Initiative. (2024). OpenAPI Specification v3.1.1. https://spec.openapis.org/oas/v3.1.1.html.

19. ZAP (Zed Attack Proxy) [Програмне забезпечення]. (2025). Checkmarx. https://www.zaproxy.org/.

Maksym Maksymovych
Postgraduate student of the Department of Information Technology Security
Lviv Polytechnic National University, Lviv, Ukraine
ORCID: 0009-0008-3604-8424
E-mail: maksym.v.maksymovych@lpnu.ua

METHOD OF CONTEXT-DRIVEN DYNAMIC SOFTWARE SECURITY TESTING USING LARGE LANGUAGE MODEL-BASED AGENTS

The advancement of artificial intelligence methods, in particular large language models and the agent-based systems built upon them, opens new opportunities for improving automated software security testing. Among the existing methods, dynamic application security testing (DAST) is the most promising direction for such improvement, since it performs end-to-end verification of a real running application and reflects its actual exploitable behaviour. At the same time, classical DAST tools construct attack scenarios through stochastic enumeration based on predefined wordlists, which results in incomplete coverage and considerable resource consumption. The aim of this work is to develop a method of context-driven dynamic software security testing based on agents built upon large language models, which increases the completeness of vulnerability detection and reduces the resource cost of scanning. The proposed method uses the OpenAPI specification as a source of semantic context, on the basis of which agents generate a set of end-to-end (E2E) security tests that act as a deterministic source of traffic and steer the dynamic scan through real business processes instead of random crawling. The method provides two improvements: extended coverage of vulnerabilities that are unreachable for signature-based scanners, and a reduced volume of redundant requests, since only the generated traffic is scanned. A vulnerability coverage model, the workflow, and the test scenario generation strategy are described, and metrics for

© Максимович М.В. Метод контекстно-керованого динамічного тестування безпеки програмного забезпечення з використанням агентів на основі великих мовних моделей. Сучасний захист інформації, 3(67), 84–91. https://doi.org/10.31673/2409-7292.2026.035907

evaluating the effectiveness of the method are defined. The results indicate the feasibility of the agent-based approach for extending coverage and improving the resource efficiency of dynamic testing, and outline the direction of further experimental research.

Keywords: dynamic application security testing, large language models, agent-based systems, OpenAPI, BOLA, IDOR, test automation.

Надійшла до редакції (Received): 27.07.2026
Прийнята до друку (Accepted): 08.09.2026
Опубліковано онлайн (Available online): 15.09.2026

УДК 004.75:004.056 DOI: 10.31673/2409-7292.2026.033419

Прокопович-Ткаченко Дмитро Ігорович
кандидат технічних наук, доцент, завідувач кафедри кібербезпеки та інформаційних технологій
Університет митної справи та фінансів, Дніпро, Україна
ORCID: 0000-0002-6590-3898
E-mail: prokopovich-tkachenko@umsf.dp.ua

Пономарьов Ігор Володимирович
кандидат технічних наук, доцент кафедри електронних обчислювальних машин
Дніпровський національний університет імені Олеся Гончара, Дніпро, Україна
ORCID: 0009-0009-7139-2885
E-mail: ponomarov_i@365.dnu.edu.ua

Бабенко Костянтин Євгенович
аспірант кафедри комп'ютерних наук та інформаційних технологій, факультету фізики, електроніки та комп'ютерних систем
Дніпровський національний університет імені Олеся Гончара, Дніпро, Україна
ORCID: 0009-0009-0945-7078
E-mail: babenko-k@365.dnu.edu.ua

ОПЕРАЦІЙНЕ ВИЗНАЧЕННЯ ТА МЕТРИКИ НАСКРІЗНОЇ ЗАТРИМКИ БЕЗПЕЧНОГО ДОСТУПУ ДО СТАНУ БЛОКЧЕЙН-СИСТЕМ ПЕРШОГО РІВНЯ

У статті досліджено проблему операційного визначення затримки, яку спостерігає кінцевий користувач програмованої блокчейн-системи першого рівня між поданням підписаної транзакції та першим підтвердженим читанням оновленого стану через RPC-інтерфейс. Актуальність зумовлена тим, що метрики консенсусної фінальності не охоплюють індексацію, публікацію стану та клієнтське виявлення, а тому можуть занижувати оцінку доступності сервісу і тривалості використання застарілого стану. Мета роботи – формалізувати користувацько-видиму затримку T_visible як окремий об'єкт вимірювання та запропонувати відтворювану систему супровідних показників. Методика охоплює причинну декомпозицію конвеєра на етапи приймання, пакетування, консенсусу, виконання, фіксації, індексації, публікації та виявлення; формальний предикат видимості сертифікованого стану; розмежування правосторонньо цензурованих спостережень і конкуруючих кінцевих подій; оцінювання квантилів і відокремлення дискретизаційної складової опитування. У чисельному експерименті, параметри якого відкалібровано за опублікованими вимірюваннями конвеєрних блокчейнів, медіанний спостережуваний розрив між фінальністю та першим читанням становив 168–241 мс, або до 27 % повної затримки, а консервативний індикатор ймовірності застарілої відповіді для затримки 200 мс досягав 0,69. Паралельний конвеєр з раннім оприлюдненням стану скоротив медіану T_visible з 889 до 686 мс і частку порушень порогу 1 с – з 17,7 % до 0,2 %. Наукова новизна полягає у поєднанні предиката видимості, метрик розриву видимості, застарілої відповіді та порушення порогу в єдиному протоколі вимірювання. Практичне значення – можливість коректно порівнювати сценарії, обґрунтовувати SLO та оцінювати доступність і цілісність даних, що надаються користувачеві.

Ключові слова: наскрізна затримка, видимість стану, блокчейн-система, RPC-інтерфейс, кіберстійкість, застаріла відповідь, якість обслуговування.