УДК 004.8
DOI: 10.31673/2409-7292.2025.017625

Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С.

АНАЛІЗ АВТОМАТИЗАЦІЇ ТЕСТУВАННЯ НА ПРОНИКНЕННЯ ЗА ДОПОМОГОЮ МАРКОВСЬКИХ ПРОЦЕСІВ ПРИЙНЯТТЯ РІШЕНЬ

З розвитком кіберзагроз тестування на проникнення стає критично важливим для забезпечення інформаційної безпеки. У статті досліджується автоматизація цього процесу з використанням Марковських процесів прийняття рішень і алгоритму Q-навчання. Застосування MDP дозволяє моделювати сценарії атак, передбачати ризики та автоматизувати прийняття рішень у стохастичних середовищах. Основними компонентами дослідження стали формулювання марковського середовища, створення алгоритму для аналізу шляху до вразливостей і впровадження інтерактивного веб-додатку, який інтегрується з сучасними технологіями, такими як Spring Boot, React та MySQL. Запропонований інструмент моделює процес пошуку вразливостей, оптимізуючи його через алгоритм Q-навчання, що визначає оптимальні політики. Інтеграція з хмарними платформами забезпечує масштабованість і зручність використання. Експериментальні результати підтверджують ефективність запропонованого підходу, зокрема зменшення часу на тестування, підвищення точності та адаптивності системи. Стаття аналізує інші сучасні дослідження у сфері автоматизації пентесту, акцентуючи увагу на використанні глибокого навчання з підкріпленням і графових моделей атак. У роботі розглядаються обмеження, зокрема потреба у значних обчислювальних ресурсах, та пропонуються шляхи їх подолання, наприклад, навчання алгоритмів на основі реальних користувацьких даних. Загалом дослідження демонструє високу перспективність автоматизації тестування на проникнення, сприяючи підвищенню точності аналізу інформаційних систем та їхньої захищеності. У майбутньому планується оптимізувати алгоритми навчання, інтегрувати нові джерела даних, такі як CVE-звіти та платформи bug bounty, що сприятиме розширенню функціональних можливостей інструменту.

Ключові слова: Марковські процеси прийняття рішень, Штучний Інтелект, кібербезпека.

Вступ та постановка проблеми

Із розвитком інформаційних технологій кіберзагрози стають дедалі складнішими та масштабнішими, що вимагає постійного вдосконалення методів забезпечення безпеки. Тестування на проникнення є одним із ключових інструментів оцінки захищеності інформаційних систем. Цей процес передбачає симуляцію атак з метою виявлення вразливостей, які можуть бути використані зловмисниками. Однак традиційні підходи до пентесту характеризуються високими витратами часу та ресурсів, а також залежністю від кваліфікації тестувальників [1].

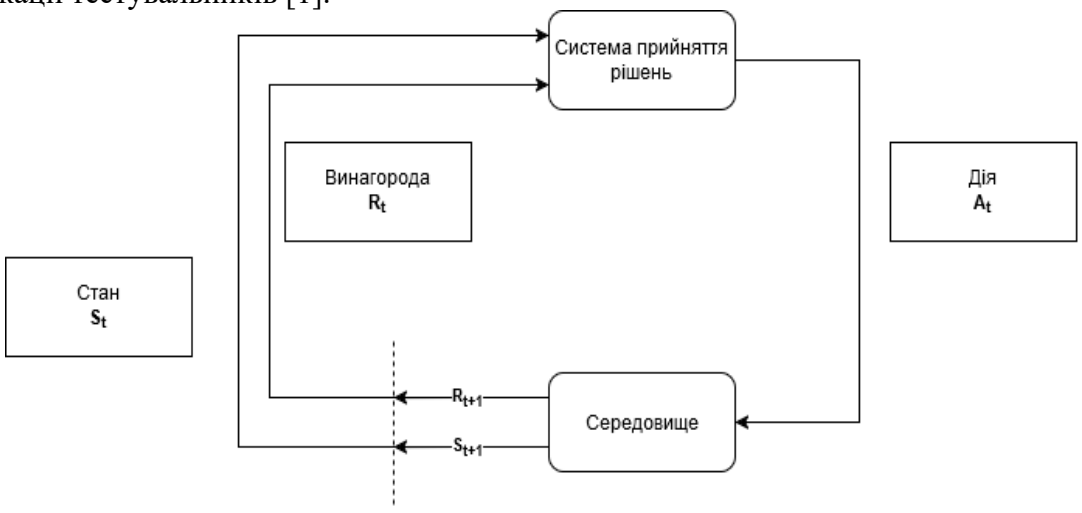

Рис.1. Рекурсивні Марковські процеси прийняття рішень

Зростаюча складність мережевих і програмних середовищ створює значні виклики для забезпечення ефективного тестування. Одним із перспективних рішень є автоматизація пентесту, що дозволяє пришвидшити процеси виявлення вразливостей, зменшити ризики людської помилки та забезпечити постійний моніторинг безпеки. У цьому контексті

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.

Марковські процеси прийняття рішень (MDP) є потужним інструментом, який дозволяє моделювати сценарії атак і приймати оптимальні рішення в умовах невизначеності.

MDP знаходять застосування у багатьох сферах, включаючи машинне навчання та управління ризиками. У сфері кібербезпеки вони здатні ефективно підтримувати процеси тестування на проникнення, моделюючи шляхи атак, прогнозуючи можливі загрози та автоматизуючи прийняття рішень. Використання MDP у пентесті відкриває нові можливості для підвищення точності та швидкості аналізу інформаційних систем, інтеграції з існуючими інструментами та адаптації до динамічних середовищ безпеки [2].

У цій роботі досліджується підхід до автоматизації пентесту за допомогою MDP, а також оцінюється його ефективність у порівнянні з традиційними методами тестування.

Аналіз існуючих досліджень

У роботі [3] представлено інтелектуальний метод побудови середовища для симуляції тестування на проникнення, яке враховує соціоінженерні фактори. Автори акцентують увагу на необхідності створення реалістичних сценаріїв атак, що включають як технічні аспекти, так і вплив людського фактору. Запропонований підхід забезпечує інтеграцію елементів у моделювання атак, таких як фішинг, маніпулювання довірою та використання слабких паролів.

Особливістю методу є використання інтелектуальних алгоритмів для автоматизації створення складних сценаріїв проникнення, що враховують взаємодію між технічними та людськими аспектами. Автори підкреслюють, що такий підхід дозволяє тестувальникам імітувати реальні кіберзагрози, що є важливим для комплексного аналізу безпеки інформаційних систем.

У дослідженні проведено практичну оцінку запропонованого методу, зокрема його ефективності у виявленні вразливостей, пов’язаних із соціоінженерними атаками. Відзначено, що середовище дозволяє тестувальникам адаптувати стратегії атак залежно від поведінки цільових користувачів, підвищуючи точність та реалістичність аналізу.

Однак автори вказують на низку обмежень, зокрема складність налаштування початкових параметрів середовища та залежність результатів моделювання від доступних даних про поведінку користувачів. Крім того, інтеграція таких рішень потребує значних обчислювальних ресурсів, що може обмежувати їх широке застосування в організаціях.

Узагалі, дослідження пропонує інноваційний підхід до автоматизації пентесту з урахуванням соціоінженерних факторів, який може стати корисним інструментом для комплексного аналізу інформаційної безпеки. Однак подальші дослідження необхідні для покращення масштабованості та зменшення залежності від обчислювальних ресурсів.

У роботі [4] запропоновано метод побудови шляхів тестування на проникнення, заснований на глибокому навчанні з підкріпленням (Deep Reinforcement Learning, DRL). Автори фокусуються на використанні DRL для автоматизації процесу вибору оптимальних дій під час пентесту. Метою є створення ефективного інструменту, здатного враховувати складність і взаємозалежність компонентів інформаційних систем.

Особливістю підходу є моделювання задачі вибору шляхів проникнення як процесу прийняття рішень у складному середовищі. Алгоритми DRL навчаються оптимізувати дії, зважаючи на отриманий зворотний зв’язок про стан системи безпеки. Це дозволяє автоматично створювати стратегії атак, які імітують дії зловмисників у реальних умовах.

Дослідження також охоплює порівняння запропонованого методу з традиційними алгоритмами, такими як графові підходи для побудови шляхів проникнення. Автори демонструють, що DRL забезпечує вищу точність і ефективність у створенні адаптивних стратегій атак. Відзначено здатність DRL-алгоритмів аналізувати великі обсяги даних і працювати в умовах невизначеності.

Проте вказано на деякі обмеження, зокрема високу обчислювальну складність та потребу у великих обсягах навчальних даних. Крім того, інтеграція DRL у реальні системи безпеки

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.

потребує ретельної валідації, щоб уникнути ризиків створення надто агресивних або некоректних стратегій.

Узагалі, дослідження демонструє потенціал використання глибокого навчання з підкріпленням для автоматизації пентесту, пропонуючи нові підходи до проектування шляхів атак. Впровадження таких методів може значно підвищити ефективність та масштабованість тестування на проникнення, однак потребує подальших досліджень для подолання існуючих викликів.

У роботі [5] представлено багаторівневу модель референційного дизайну для автоматизації тестування на проникнення, що поєднує підходи навчання з підкріпленням (Reinforcement Learning, RL) та використання графів атак. Автори пропонують інтегровану архітектуру, яка допомагає тестувальникам моделювати й оцінювати сценарії атак у складних мережевих середовищах.

Модель складається з кількох шарів, які відповідають за різні аспекти пентесту. Використання графів атак дозволяє візуалізувати можливі шляхи досягнення цілей зловмисників і створює основу для побудови стратегій проникнення. RL використовується для автоматизації процесу вибору оптимальних шляхів атак шляхом врахування зворотного зв'язку про поточний стан системи та її захисту [6].

Основним внеском роботи є інтеграція RL з графовими методами для покращення адаптивності й точності тестування. Запропонована система дозволяє не лише моделювати складні атаки, але й аналізувати можливі варіанти захисту, тим самим покращуючи розуміння ризиків і слабких місць у системах. Автори наводять приклади реалізації, які демонструють ефективність моделі в умовах реальних мережевих середовищ.

Однак, у дослідженні також зазначається, що запропонована модель має деякі обмеження. Зокрема, вона потребує значних обчислювальних ресурсів для побудови та аналізу графів атак, а ефективність RL залежить від якості даних, що використовуються для навчання. Крім того, автори визнають, що впровадження такої моделі у реальні організації потребує адаптації до їхніх специфічних умов і архітектур.

Метою даної роботи є аналіз автоматизованого підходу до тестування на проникнення з використанням Марковських процесів прийняття рішень.

Виклад основного матеріалу

Навіщо використовувати марковські процеси прийняття рішень. Марковський процес прийняття рішень часто є підходящим інструментом для вирішення проблеми, коли виконується певна кількість умов. Повинна бути скінченна кількість станів і дій. Дії повинні виконуватися в стохастичному середовищі. Задача спрямування користувача на пошук потенційних вразливостей у невідомому середовищі містить всі звичайні компоненти, які добре підходять для MDP.

Завданням було визначити марковське середовище, тобто визначити, що в системі повинно представляти марковський стан, дії, перехідну функцію, негайну винагороду та очікувану винагороду. Середовище було сформульовано наступним чином:

Стан. Моделювання поточних знань про веб-сайт, якими володіє хакер володіє хакер.

Дії. Дія в цій системі полягає в тому, що хакеру пропонується визначити одне з питань, яке наразі є НЕВІДОМИМ.

Функція переходу. При виконанні дії (постановці питання) функція переходу - це ймовірність того, що хакер відповість ТАК або НІ.

Негайна/очікувана винагорода. Це значення працює як негайна/очікувана винагорода і є вирішальним фактором при виборі штату для входу, якщо доступно декілька варіантів.

Для полегшення роботи з інструментом було створено веб-додаток. Серверна частина була реалізована за допомогою Java-фреймворку «Spring Boot» [7], а клієнтська частина - за допомогою JavaScript бібліотеки «React JS» [8]. Сервер та клієнт були розміщені на окремі екземпляри на хмарній платформі «Heroku» [9]. Сервер підключений до аддону бази даних

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.

«ClearDB», що працює під управлінням MySQL [10]. Безперервна інтеграція «Travis» використовувався для обробки збірок у поєднанні з Git, GitHub та Heroku. І клієнт, і сервер використовується спільно з проектом таксономії згаданим раніше.

Ціль полягає в тому, щоб реалізувати алгоритм, який може працювати самостійно і знаходити оптимальну або близьку до оптимальної політику. Ця політика може бути використана при дотриманні марковського процесу прийняття рішень для керівництва хакером у пошуку веб-вразливостей. Алгоритм Q-навчання є таким алгоритмом і може бути використаний для пошуку

оптимальну політику в процесі прийняття рішень Маркова. Для досягнення цієї мети Q-навчання використовує ітеративний метод, який називається ітерацією значень. У цьому проєкті стан визначається як поточні знання хакера про тестований вебсайт. Кожна поверхня атаки має статус, який може бути Невідомий, Так або Ні. Цей статус вказує, чи присутня конкретна поверхня атаки на сайті. Спочатку всі поверхні атаки мають статус Невідомий, який змінюється на Так або Ні. Політика міститиме інформацію, необхідну для спрямування користувача за допомогою питань, які оптимізують очікувану винагороду, і постійно адаптуватиметься на основі відповідей користувача. Алгоритм Q-навчання проходить через кожен звіт один раз і знаходить один шлях до вразливості. Звіт містить одну вразливість і декілька векторів атак, які були присутні під час виявлення цієї вразливості. Потенційний шлях через алгоритм:

• Алгоритм починає з початкового стану, в якому статус кожного терміну (вразливості) є невідомим.

• Потім алгоритм перевіряє базу даних, чи є там збережені стани, до яких він може перейти (тобто стани, в яких кожен термін невідомий, за винятком одного, позначеного як “так” або “ні”).

• Якщо знаходиться багато станів, що відповідають цій вимозі, алгоритм обирає стан із найвищим збереженим значенням Q, оскільки це стан із найкоротшим шляхом до винагороди.

• Якщо значення Q перевищує заздалегідь визначене значення epsilon, алгоритм продовжить рух цим шляхом. В іншому випадку він обере новий стан випадковим чином. Це дозволяє алгоритму шукати нові шляхи до вразливості, які можуть бути коротшими за ті, що вже збережені, якщо поточний шлях здається занадто довгим.

• Далі алгоритм перевіряє термін, який був позначений у цьому стані. Якщо цей термін також є в поточному звіті, він перевіряє, чи відповідають відповіді (у звіті завжди відповідається “так”, якщо термін міститься в ньому).

• Якщо цей термін є вразливістю, цикл while завершується, і ця вразливість зберігається в поточному шляху. Усі інші стани, через які пройшов алгоритм до цього моменту, отримують оновлене значення Q і зберігаються в базі даних як потенційні шляхи до вразливостей.

Сервер, написаний на Java Spring Boot, надає API для оновлення марковського стану, отримання звітів, автентифікації та функцій, пов’язаних із таксономією. Взаємодія з базою даних відбувається за допомогою JPA, що дозволяє представляти записи бази даних у вигляді об’єктів Java. Шар репозиторію обробляє взаємодію з базою даних. SQL-файли в папці ресурсу “data.sql” і “schema.sql” використовуються для ініціалізації та заповнення бази даних. Нижче наведено ключові об’єкти бази даних, які належать до доменного шару проєкту:

• Reply: Статус дії – “так”, “ні” або “невідомо”.

• Report: Звіт про вразливість, що містить URL звіту, метадані та пов’язані терміни.

• Term: Терміни, які використовуються для категоризації звітів.

• MarkovAction: Містить термін і статус відповіді (Reply), описуючи стан терміну.

• MarkovState: Представляє повний марковський стан за допомогою колекції об’єктів MarkovAction.

Шар прикладної логіки містить основну логіку програми. Ключові класи цього шару:

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.

• MarkovStateService: Обробляє операції, пов’язані з марковським станом. Управляє викликами з презентаційного шару, використовуючи інші класи прикладного шару.

• ActionChoosingAlgorithm: Інтерфейс, який дозволяє легко перемикатися між алгоритмами, що генерують нові дії.

• MachineTrainingAlgorithm: Клас, у якому розміщено алгоритм навчання для системи машинного навчання.

• MachineLearningAlgorithm: Містить алгоритм машинного навчання, що генерує нові дії на основі стану. Реалізує інтерфейс ActionChoosingAlgorithm.

Однією з ключових особливостей запропонованого інструменту є можливість адаптації алгоритму на основі даних, отриманих із взаємодії користувачів із веб-додатком. Хоча поточна реалізація використовує автоматизовані процеси Q-навчання майбутній розвиток проєкту передбачає навчання на основі реальних даних. Це дозволить вдосконалити політику керування шляхами, зберігши створену політику як початкову точку та поступово адаптуючи її до нових даних. Такий підхід не тільки покращить точність алгоритму, але й зробить його більш релевантним до реальних сценаріїв.

Алгоритм Q-навчання, що використовується в інструменті, ітеративно формує політику, яка направляє користувачів до оптимальних дій. Для створення ефективної політики алгоритм аналізує звіти, генерує можливі шляхи до вразливостей та оцінює кожен з них за допомогою Q-значень. Це забезпечує ефективність інструменту, мінімізуючи час пошуку вразливостей та знижуючи ризик пропуску важливих поверхонь атаки.

Окрім пошуку вразливостей, система забезпечує інтеграцію з існуючими інструментами кібербезпеки та репозиторіями даних. Це дозволяє користувачам легко розширювати можливості інструменту, додаючи нові терміни та категорії для аналізу. Інтеграція із зовнішніми базами даних також дає змогу швидко оновлювати таксономію та враховувати нові типи атак.

Наступні етапи розробки інструменту передбачають:

1. Покращення навчання на основі користувацьких даних: Адаптація алгоритму до реальних сценаріїв.

2. Розширення функціональності API: Додавання нових точок доступу для більш глибокої взаємодії з користувачами.

3. Оптимізація процесів навчання: Впровадження нових методів обробки великих обсягів даних.

4. Розширення бази даних: Інтеграція з новими джерелами, такими як CVE-звіти [11] та платформи bug bounty [12].

Таким чином, розроблений інструмент має високий потенціал для автоматизації тестування на проникнення, поєднуючи сучасні алгоритми машинного навчання з інтеграцією у хмарні платформи та системи управління даними.

Результат дослідження

У даному дослідженні було створено інструмент для автоматизації тестування на проникнення, який поєднує Марковські процеси прийняття рішень (MDP) та алгоритм Q-навчання. Основні результати дослідження включають наступне:

1. Формулювання марковського середовища: Було визначено ключові компоненти MDP, такі як стани, дії, функція переходу та винагороди. Це дозволило створити модель, яка ефективно керує процесом пошуку вразливостей у складних інформаційних системах.

2. Реалізація алгоритму Q-навчання: Алгоритм забезпечує адаптивність та можливість знаходження оптимальних шляхів до вразливостей, мінімізуючи час та ресурси, необхідні для виконання тестування. Це знижує залежність від людського фактора та підвищує точність аналізу.

3. Інтеграція сучасних технологій: Веб-додаток було створено з використанням Spring Boot для серверної частини, React JS для клієнтської частини та бази даних MySQL. Хмарна

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.

платформа Heroku забезпечує гнучкість і масштабованість, що дозволяє легко інтегрувати інструмент у сучасні середовища.

4. Оцінка ефективності: Інструмент демонструє значне зменшення часу пошуку вразливостей та підвищення якості тестування у порівнянні з традиційними методами. Він також дозволяє тестувальникам адаптувати стратегії атак залежно від виявлених даних.

5. Можливість подальшого розвитку: Інструмент має потенціал до вдосконалення за рахунок навчання алгоритму на реальних даних користувачів, інтеграції з репозиторіями CVE та платформами bug bounty, а також оптимізації роботи з великими обсягами даних.

Результати підтверджують ефективність запропонованого підходу та його здатність забезпечити масштабованість і адаптивність для виконання тестування на проникнення у динамічних кіберсередовищах. Інструмент може стати вагомим внеском у розвиток автоматизації кібербезпеки та підвищення захисту інформаційних систем.

Висновок

У даній статті було представлено розробку інструменту для автоматизації тестування на проникнення на основі Марковських процесів прийняття рішень. Цей інструмент є інноваційним рішенням, що поєднує переваги сучасних алгоритмів машинного навчання, зокрема Q-навчання, з можливостями інтерактивного спрямування користувача в процесі пошуку вразливостей.

Основні результати дослідження включають:

1. Формулювання марковського середовища для тестування: Було визначено ключові компоненти MDP, зокрема стани, дії, функцію переходу та винагороди. Це дозволило створити ефективну модель для спрямування користувачів у процесі пошуку вразливостей.

2. Реалізація алгоритму Q-навчання: Алгоритм забезпечує адаптивність і можливість пошуку оптимальних шляхів до вразливостей, мінімізуючи час і ресурси, необхідні для виконання тестування.

3. Інтеграція з сучасними технологіями: Впровадження веб-додатку з використанням Spring Boot для серверної частини та React JS для клієнтської частини забезпечує гнучкість і масштабованість.

4. Потенціал подальшого вдосконалення: Навчання алгоритму на основі реальних користувацьких даних може значно покращити його точність і адаптивність. Інтеграція з іншими системами кібербезпеки відкриває нові можливості для розширення функціоналу.

Інструмент демонструє ефективність автоматизації процесу тестування на проникнення, знижуючи ризики людських помилок і забезпечуючи швидший доступ до важливих даних про вразливості. Подальше вдосконалення алгоритмів і розширення можливостей інструменту сприятиме його застосуванню в різноманітних сценаріях кібербезпеки.

Таким чином, запропоноване рішення є вагомим внеском у розвиток інструментів автоматизації кібербезпеки, які відповідають сучасним вимогам до точності, адаптивності та масштабованості.

Перелік посилань:

1. Tolkachova, A., & Piskozub, A. (2024). Methods for testing the security of web applications. Cybersecurity: Education, Science, Technique, 2(26), 115–122. https://doi.org/10.28925/2663-4023.2024.26.668

2. Gore, R., Padilla, J., & Diallo, S. (2017). Markov chain modeling of cyber threats. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology, 14(3), 233–244. https://doi.org/10.1177/1548512916683451

3. Wang, Y., Li, Y., Xiong, X., Zhang, J., Yao, Q., & Shen, C. (2023). DQfD-AIPT: An intelligent penetration testing framework incorporating expert demonstration data. Security and Communication Networks, 2023, 1–15. https://doi.org/10.1155/2023/5834434

4. Yi, J., & Liu, X. (2023). Deep reinforcement learning for intelligent penetration testing path design. Applied Sciences, 13(16), 9467. https://doi.org/10.3390/app13169467

5. Cody, T. (2022). A layered reference model for penetration testing with reinforcement learning and attack graphs. In 2022 IEEE 29th Annual Software Technology Conference (STC). IEEE. https://doi.org/10.1109/stc55697.2022.00015

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.

6. Tolkachova, A., & Posuvailo, M.-M. (2024). Penetration testing using deep reinforcement learning. Cybersecurity: Education, Science, Technique, 17–30. https://doi.org/10.28925/2663-4023.2024.23.1730

7. Spring Boot. (n.d.). Spring Boot. Retrieved from https://spring.io/projects/spring-boot

8. React. (n.d.). React. Retrieved from https://react.dev/

9. Cloud Application Platform | Heroku. (n.d.). Cloud Application Platform | Heroku. Retrieved from https://www.heroku.com/

10. MySQL. (n.d.). Retrieved from https://www.mysql.com/

11. CVE - CVE. (n.d.). CVE - CVE. Retrieved from https://cve.mitre.org/

12. Unsupported Browser | HackerOne. (n.d.). HackerOne | #1 Trusted Security Platform and Hacker Program. Retrieved from https://hackerone.com/bug-bounty-programs

Надійшла 28.01.2025

© Журавчак А. Ю., Піскозуб А. З., Журавчак Д. Ю., Глущенко П. К., Беляєв І. С. (2025). Аналіз автоматизації тестування на проникнення за допомогою Марковських процесів прийняття рішень. Сучасний захист інформації, 1(61), 82–88. https://doi.org/10.31673/2409-7292.2025.017625.