DOI 10.15276/imms.v11.no3.227
УДК 004.056.5

Informatics and Mathematical Methods in Simulation
Vol. 11 (2021), No. 3, pp. 227-238

РОЗРОБКА ЗАХИЩЕНОЇ АВТОМАТИЗОВАНОЇ ВЕБ-СИСТЕМИ ТЕСТУВАННЯ ЗНАНЬ

С.О. Перерва, К.О. Трифонова

Національний університет «Одеська політехніка»,
просп. Шевченка, 1, Одеса, 65044, Україна; e-mail: katikkatik@gmail.com

Сучасна освіта націлена на надання можливості отримання якісних знань та умінь. Багато чисельні дослідження останніх років демонструють значний вклад інформаційних технологій в підвищення якості отримуваних знань та умінь в будь-яких сферах освіти. Для оцінки якості отримуваних знань на протязі учбового процесу необхідно виконувати проведення контролю у різних формах. Значні педагогічні дослідження доводять ефективність проведення тестового контролю для отримання оцінки якості знань та умінь. Інформаційні технології можуть допомогти підвищити незалежність перевірки та встановлення оцінки від особи викладача, тобто підвищити об'єктивність, завдяки застосуванню форми проведення тестування у автоматизованому форматі. У відповідності до сучасних реалій, розробка, реалізація та подальше використання автоматизованого програмного забезпечення для тестування знань, відбувається у мережі Інтернет. У зв'язку з цим, фактором, який повинен бути визначений найважливішим, є інформаційна безпека. Отже, метою даної роботи є розробка та реалізація автоматизованої веб-системи тестування знань з підвищеним рівнем інформаційної безпеки. Один з найпопулярніших відкритих проектів з дослідження безпеки веб-додатків Open Web Application Security Project, щорічно публікує топ десять найпоширеніших загроз. В роботі представлено розробка механізму захисту програмного забезпечення, рівень інформаційної безпеки якої, забезпечує захист інформації від найбільш критичних ризиків, встановлених у відповідності до OWASP Топ-10 2021. Розроблений програмний продукт може бути рекомендований для застосування в вищих та загальноосвітніх навчальних закладах, для підвищення якості знань та об'єктивності, завдяки застосуванню форми проведення тестування у автоматизованому форматі, з підвищеним рівнем інформаційної безпеки.

Ключові слова: захист інформації, веб-застосування, дистанційне навчання, якість освіти, тестування знань

Вступ

Сучасна освіта націлена на надання можливості отримання якісних знань та умінь [1-7].

Використання інформаційних технологій при оцінюванні почалося в 1920-х роках, коли Сідні Л. Пресс розробив машини для автоматичного тестування. Більше того, водночас у школах почали використовувати стандартизоване оцінювання та технологію автоматичного підрахунку балів, що допомогло зробити масштабне тестування зручним та економічно ефективним. Величезні зміни в багатьох секторах, особливо в освіті, відбулися, коли в 1990-х роках було введено всесвітню мережу. З того часу багато компаній запровадили власну систему електронного оцінювання. В Англії, Уельсі та Північній Ірландії принципи та вказівки щодо електронного оцінювання були запроваджені JISC (Об'єднаний комітет інформаційної системи) для роз'яснення різних кваліфікаційних регуляторів у Сполученому Королівстві. У 2009 році IMS Global Learning Consortium розробив специфікацію інтероперабельності IMS Question and Test. У 2009 році Cisco, Intel та Microsoft випустили Transforming Education: Assessing and Teaching 21st Century Skills [3].

На даний час, електронне оцінювання надає нові привабливі можливості та методики оцінювання різних аспектів і рівнів набутих знань, умінь і компетенцій. Це допомагає подолати деякі недоліки, які притаманні традиційним підходам до оцінювання [4].

Електронне оцінювання має ряд переваг [4]:

- використання різних методик оцінювання, які відповідають заздалегідь поставленим цілям навчання та використовуються педагогічні підходи;
- оцінка великої кількості тестів виконується швидко і легко завдяки високому ступеню процесу автоматизації;
- негайний персоналізований зворотній зв'язок, для подальшого корегування навчання;
- об'єктивність за рахунок автоматизації оціночної діяльності або оцінювання на основі заздалегідь встановлених критеріїв і шаблонів;
- впровадження в будь-який час і в будь-якому місці, що гарантує більшу гнучкість;
- можливість повторного використання діяльності або їх компонентів у різних комбінаціях і контекстах;
- широкий вибір типів запитань в електронних тестах;
- можливість оцінити групу у цілому, враховуючи індивідуальний внесок.

Розрізнюють наступні види електронного оцінювання [4]:

- за поставленими навчальними цілями та періодом, в який воно проводиться, оцінювання може бути: діагностичним (допомагає поставити чіткі, точні та адекватні навчальні цілі, що відповідають рівню готовності до роботи з новим змістом); формуюча (допомагає контролювати успішність та вносити зміни в навчальний процес, щоб адаптуватися до потреб та покращити навчання); підсумковий (підтримує оцінку знань і навичок наприкінці навчання за встановленими критеріями або стандартами, щоб визначити, чи були досягнуті поставлені цілі навчання [5]);
- відбувається оцінювання одночасно чи ні: синхронно (одночасно оцінюються); асинхронно (оцінювання в зручний час);
- за кількістю оцінюваних та суб’єктів-оцінювачів: групові (оцінюються вироби, які є результатом спільної роботи групи та їх уміння працювати в команді); індивідуальний (оцінюються індивідуальні знання та вміння); рівноправний (допомагає розвивати критичне мислення, мотивувати та відстоювати власну думку на основі фактів і доказів) [6].

Отже, багато чисельні дослідження останніх років демонструють значний вклад інформаційних та комунікаційних технологій в підвищення якості отримуваних знань та умінь в будь-яких сферах освіти, та доводять необхідно виконувати проведення тестового контролю для отримання оцінки якості знань та умінь на протязі учбового процесу. У відповідності до сучасних реалій, розробка, реалізація та подальше використання автоматизованого програмного забезпечення для тестування знань, відбувається у мережі Інтернет. У зв’язку з цим, фактором, який повинен бути визначений найважливішим, є інформаційна безпека. Компанії з розробки програмного забезпечення найчастіше не дотримуються вимог інформаційної безпеки, концентруючись на інших цілях. Компанія Positive Technologies постійно виконує дослідження з інформаційної безпеки веб-додатків, які свідчать, що значна частина програмного веб-забезпечення містить вразливості різного ступеня ризиків. OWASP – відкритий проект з дослідження безпеки веб-додатків, щорічно публікує топ десять найпоширеніших загроз. Все це стимулює проведення подальших досліджень в поданій галузі та робить тему даної роботи надзвичайно актуальною.

Мета і задачі дослідження

Метою роботи є підвищення рівня інформаційної безпеки розробленої та

реалізованої автоматизованої веб-системи тестування знань.

Для досягнення поставленої мети необхідно розв'язати наступні *задачі*:

а) виконати аналіз педагогічного напрямку дослідження форми проведення та оцінки якості знань у форматі автоматизованого тестування;
б) виконати аналіз найбільш критичних ризиків та існуючих механізмів захисту веб-додатків;
в) розробити механізм захисту програмного забезпечення, рівень інформаційної безпеки якої, забезпечує захист інформації від найбільш критичних ризиків, встановлених у відповідності до OWASP Топ-10 2021;
г) реалізувати програмний продукт, автоматизовану веб-систему тестування знань, з підвищеним рівнем інформаційної безпеки, у відповідності до запропонованого механізму захисту програмного забезпечення.

Основна частина

На сьогодні інформаційні технології стають все більш поширеними в суспільстві. Вони використовуються в найрізноманітніших галузях: у соціальних мережах, електронній пошті, новинних та урядових порталах, електронній комерції, форумах, блогах та інших веб-сайтах. Разом з розвитком ІТ все більшого значення набуває інформаційна безпека (ІБ). Якщо розглядати історичну перспективу розвитку інформаційних технологій, то виявляється, що з 1996 року, коли була прийнята Конституція нашої держави, і до моменту написання цієї роботи кількість людей, які відчувають себе захищеними в інформаційному полі, постійно падає. Справа не лише в лавиноподібному зростанні використання ІТ, а, швидше за все, у порушеннях, які держава робить в інформаційній сфері. Зростання вимог до забезпечення інформаційних технологій визначається, насамперед, розвитком ІТ. Потенційні порушники мають нові можливості для деструктивних дій у всіх сферах нашого життя. Найефективніші з цих дій відбуваються у кіберпросторі. Виходячи з вищесказаного, можна зробити висновок про необхідність захисту веб-додатків.

Залежно від того, де використовується веб-додаток, в ньому може оброблятися інформація різних рівнів доступу та значень. Наприклад, програми можуть використовувати інформацію про банківські картки, персональні дані користувачів, паролі та інші ідентифікаційні дані. Жоден користувач не застрахований від того, що його дані можуть бути скопійовані або піддані ряду випадкових і зловмисних впливів у процесі їх обробки, передачі та зберігання [8].

Веб-додатки реалізують архітектуру «клієнт-сервер». У цій концепції клієнт – це браузер користувача, а сервер – веб-сервер. В основі концепції «клієнт-сервер» відбувається обмін інформацією через мережу, клієнт починає взаємодіяти, а дані зберігаються переважно на сервері.

В основному веб-додатки мають розподілену структуру. Основні переваги цієї конструкції:

- хороша масштабованість – можливість збільшення функціональності без зміни структури;
- наявність можливості керувати навантаженням програми – направлення потоків запитів користувача на менш завантажені сервери.

Типова архітектура цих програм може бути представлена у трьох рівнях: клієнтська частина (веб-браузер), веб-сервер та база даних.

Кожен день будь-який веб-сайт може піддаватися кібератакам. Як правило, більшість атак є цільовими. Це означає, що зловмисник не обмежується однією спробою отримати необхідні дані несанкціонованим способом, оскільки він не знає, які саме уразливості є в коді. Усі спроби зламати сайт зводяться до серії подій, які відбуваються протягом певного періоду часу. За статистикою за 2018 рік, найбільша

кількість атак на одне веб-додаток припадає на сайти фінансових організацій, транспортних компаній та обслуговуючих компаній. Вибір методу атаки залежить від особливостей веб-додатка. Припустимо, якщо в додатку не передбачена можливість введення даних користувача, то зловмисник не буде проводити атаки, спрямовані на зміну логіки програми за допомогою введення будь-яких даних користувачем. Найпопулярніші атаки включають: введення коду SQL, вихід за межі каталогу та між-сайтові сценарії.

Щоб виявити вразливості в коді та зрозуміти, як система реагуватиме на атаку – програму потрібно протестувати. Процес тестування максимально схожий на процес злому, який проводить зловмисник. Мета таких дій – визначити, наскільки вразливим є веб-додаток.

Для критичної інфраструктури спеціалізовані фішери використовують передові методи, які поєднують соціальну інженерію, орієнтуючись як на відсутність спеціалізованих активних заходів безпеки системи, так і на недостатню обізнаність або пильність співробітників [9].

Найбільш популярний методології тестування [8]:
- посібник з методології тестування безпеки з відкритим кодом; - Спеціальна публікація Національного інституту стандартів і технологій (NIST) 800-115;
- керівництво з тестування OWASP;
- стандарт виконання тестування на проникнення;
- структура оцінки безпеки інформаційних систем.

Для перевірки безпеки веб-додатків доцільніше використовувати методологію OWASP. Ця методологія заснована на методі чорного ящика – інформація про тестований додаток обмежена або відсутня взагалі. Безпека програмного забезпечення охоплює дуже широку область тем. Щоб мати безпечне програмне забезпечення, потрібно враховувати багато речей.

Найважливіші вразливості OWASP

Розуміння поширених вразливостей у веб-додатках допомагає краще підготуватися до захисту даних від таких атак. Завдяки знанням, отриманим від досліджень, користувачі та розробники можуть бути краще підготовлені для боротьби з найпоширенішими атаками та формувати рішення для запобігання майбутнім атакам на їхні веб-додатки. Вразливості існують у багатьох формах у сучасних веб-додатках, які можна легко пом’якшити, вкладаючи час та дослідження [10].

Нижче перелічено найважливіші вразливості 2021 року.

Атаки SQL Injection відбуваються, коли введені дані користувача не очищені належним чином і мають найбільший вплив на Інтернет. Коли зловмисники знаходять ін’єкцію SQL, у більшості випадків вони можуть отримати дані з бази даних, а в деяких випадках призводять до віддаленого виконання коду в цільовій системі. Атаки ін'єкцій SQL бувають у багатьох формах, таких як атаки на основі помилок, атаки на основі об'єднань, сліпі атаки та атаки поза діапазоном [10].

Порушена аутентифікація – це термін, який використовується для позначення кількох уразливостей, які дозволяють зловмисникам використовувати та видавати себе за користувачів веб-додатків. Існують різні методи, за допомогою яких зловмисники можуть отримати облікові дані користувача або зловити сеанси користувача, щоб мати можливість видавати себе за цих користувачів, наприклад, слабкі облікові дані користувача, які можна вгадати, неправильно збережені облікові дані, такі як паролі, які не були хешовані, які можуть бути вилучені з інших типів атак. такі як ін'єкції SQL, ідентифікатори сеансів, які відкриваються в URL-адресах, атаки фіксації сеансів, фіксовані ідентифікатори сеансів і особливо паролі, ідентифікатори сеансів та інші облікові дані, які надсилаються через незашифровані з'єднання, такі як HTTP [10].

Конфіденційні дані – це веб-програми, які не захищають таку інформацію, як паролі, фінансову інформацію або дані про стан здоров'я, що може призвести до того, що кіберзлочинці зловживають цією інформацією для отримання несанкціонованого доступу до облікових записів користувачів, вчиняють шахрайські дії, такі як онлайн-покупки за допомогою вкраденого платежу. інформації або здійснювати шантаж отриманими конфіденційними даними. Розкриття конфіденційних даних може призвести до фінансових втрат, завдати шкоди репутації корпорацій, які розкрили свою інформацію чи активи, і спонукає підприємства оплачувати витрати на розслідування порушень даних. Захист від таких атак залежить від законодавчої бази країни та галузі, оскільки їх ігнорування може призвести до фінансово руйнівних результатів.

Атаки введення зовнішніх об'єктів XML, також відомі як ін'єкції XXE, відбуваються, коли зловмисники зловживають аналізаторами розширеної мови розмітки (XML) на веб-серверах, надсилаючи на веб-сервери спеціально створені шкідливі XML-документи, які обробляються і можуть призвести до відмови в обслуговуванні, віддаленого виконання коду або підробка запитів на стороні сервера.

Порушений контроль доступу складається з кількох можливих векторів атак, таких як обхід перевірок контролю доступу, редагування облікових записів інших користувачів, підвищення привілеїв, неправильні налаштування CORS, які дозволяють несанкціонований доступ до обмежених API, маніпулювання метаданими за допомогою маркерів контролю доступу, таких як веб-токени JSON (JWT) або доступ неавторизовані веб-сторінки як непривілейований користувач, що може призвести до контролю зловмисників бізнес-функцій або можливості отримання зловмисниками всіх даних. Рекомендується використовувати списки контролю доступу та забороняти доступ до функцій, використовуючи код на стороні сервера, де зловмисники не можуть отримати доступ до метаданих або контролювати їх.

Неправильна конфігурація безпеки відноситься до веб-програм, які були неправильно налаштовані таким чином, що залишають їх підданими загрозам безпеки. Вони можуть включати неправильні конфігурації брандмауера, відкриті порти адміністрування, які піддають програму віддаленим атакам, або застарілі програми, які намагаються спілкуватися з програмами, які більше не існують. Забезпечення того, що конфігурації проходять належний процес забезпечення якості та ретельно перевіряються, тестуються та перевіряються, зменшує поверхню атаки від такого типу вразливості.

Міжсайтові сценарії, також відомі як XSS, мають багато форм, які призводять до різних результатів залежно від типу виконуваного XSS, але зазвичай відбувається, коли зловмисники вводять у веб-додаток шкідливі сценарії, які потім розкривають конфіденційну інформацію, внутрішні служби або розкривають файли cookie привілейованих користувачів.

Небезпечні атаки десеріалізації відбуваються, коли програми намагаються перетворити шкідливі дані, які контролює зловмисник, у внутрішні структури даних, які контролюються програмою. Впроваджуючи спеціально створені корисні навантаження, зловмисники можуть взяти під контроль змінні, функції та внутрішні стани програми. Це часто призводить до вразливостей віддаленого виконання коду, а також до впливу операційної системи веб-додатка.

Використання компонентів з відомими вразливими місцями означає використання певного програмного або апаратного забезпечення з відомими вразливими місцями, незалежно від того, чи вони були припинені або закінчилися терміном служби. Зловмисники, швидше за все, використовують поширені або відомі експлойти для отримання доступу до систем, а не виявляють нові вразливості. Захист від цієї вразливості вимагає відстеження залежностей програми, належної документації, видалення невикористаних залежностей, видалення мертвого коду та

включення залежностей у політику оновлення програми, процедури та життєвий цикл обслуговування.

Недостатнє ведення журналів і моніторинг означає відсутність належних механізмів ведення журналів, які допомагають у моніторингу та виявленні інцидентів безпеки. Це дозволяє зловмисникам вести свою діяльність непомітно, що значно ускладнює завдання виявлення інцидентів та реагування на атаки. Журнали використовуються не лише для відстеження діяльності зловмисників або виявлення помилок та інших аномальних дій, які можуть мати місце в програмі. Крім того, багато нормативних вимог залежать від належних механізмів ведення моніторингу.

Методи самозахисту веб-додатку під час виконання

Додатки соціальних мереж стали важливим джерелом Інтернет-послуг і трафіку. Як правило, він заснований на динамічних сторінкових програмах, таких як обмін інформацією, інтерактивний зворотний зв'язок і запити на послуги, що надаються веб-програмами. Більшість хмарних платформ та інтерфейсів керування мережевими пристроями також використовують веб-інтерфейси. Упорядковуючи додаткові оператори сценаріїв у введених користувачами, за відсутності надійних методів фільтрації, міжсайтові сценарії можуть викликати ефект вставки зайвого коду на сторінки веб-програм і порушення логіки вихідної сторінки, спричиняючи зловмисних дій клієнта.

Зазвичай уразливості міжсайтових сценаріїв можна уникнути за допомогою безпечних методів програмування, таких як фільтрація даних користувача. Оскільки веб-додатки побудовані на мові сценаріїв високого рівня, розробники більше стурбовані дизайном інтерфейсу з Web 2.0, а технології інтерфейсу стають більш складними. В результаті ускладняються перетікання змінних. Якщо це відсутність загального дизайну та розробки безпеки, а також немає ефективної фільтрації та обмежень, оскільки вразливості, засновані на архітектурі такого типу додатків, неминуче існуватимуть міжсайтові вразливості, які стануть актуальною проблемою безпеки мережі в умовах широкого застосування. Уразливості міжсайтових сценаріїв широко існують у веб-додатках у звіті про загрози безпеки OWASP десяти найбільших (OWASP top 10), ін'єкція команд і міжсайтове скриптування залишаються важливими загрозами [8]. Його можна виявити як на стороні сервера, так і на стороні клієнта. Метод виявлення на стороні сервера в основному полягає в тому, щоб переглянути введений користувачем вміст, відфільтрувати ненадійний вміст і зробити так, щоб шкідливий код сценарію не міг досягти браузера користувача. З точки зору виявлення на стороні сервера, репрезентативними методами є вбудовані політики з підтримкою браузера (BEEP). Щоб запобігти введенню зловмисного коду сценарію у веб-програми, BEEP реалізує білий список довірених сценаріїв. Таким чином, створюється лише код надійного сценарію веб-розробником може виконуватися на клієнті, а решта відфільтровується. Однак BEEP має серйозні недоліки при роботі з динамічно генерованими сценаріями. Особливо коли поточна структура веб-моделі дозволяє процедурний дизайн, проблема стає більш очевидною, і були складніші атаки XSS, спеціально спрямовані на BEEP.

Метод визначення політики безпеки вмісту (Content Security Policy - CSP) довіряє виконанню браузера. Вважається, що причиною проблеми XSS є недоліки у веб-додатку. З цієї причини CSP додає багато атрибутів до основної частини кожної сторінки, створеної на стороні сервера. Під час створення сторінки надійність сценаріїв, зображень або іншого вмісту на сторінці визначається цими атрибутами. Таким чином, він забезпечує функцію автоматичного захисту від несправності. Оскільки ця політика поширюється на всю сторінку, ненадійний вміст обмежується цією політикою. Однак, оскільки CSP накладає суворі обмеження на створені сторінки, це спричинить серйозні проблеми з продуктивністю перед великомасштабними веб-

додатками, а також не може добре адаптуватися до веб-архітектури дизайну процедурної моделі [11].

WAF (брандмауер веб-додатків) проникає у веб-протокол для фільтрації та використовує технологію брандмауера для обмеження з'єднань. WAF зазвичай розгортається на передньому кінці кластера веб-серверів для захисту веб-сайтів і використовує режим зворотного проксі для виконання двонаправленої фільтрації пакетів запитів HTTP. Якщо брандмауер розуміє протокол і семантику параметрів програми, це призведе до більш точного виявлення вразливостей. Однак із збільшенням складності додатків і швидкою появою нових типів програм і технологій (таких як JSON, REST тощо), здатність WAF забезпечувати точне виявлення вразливостей без штучної настройки ще не реалізована. Сьогодні мало хто вважає, що WAF розгорнуто в режимі безумовного блокування для будь-якої програми, що доводить притаманну неточність технології WAF. Деякі вдосконалені методи використовують машинне навчання для вивчення правил підпису, а також стикаються з труднощами отримання набору даних і робочого навантаження на тегування вручну [11]. Комбінація методів виявлення на стороні сервера і клієнта в основному полягає в тому, що клієнтський браузер повинен проаналізувати повернутий HTML-документ способом, визначеним сервером, щоб шкідливий код не міг бути виконаний, навіть якщо він досягне клієнта. Створення наскрізного довіреного шляху між користувачем веб-програми та веб-сервером. Типовими прикладами є цілісність структури документів (DSI) [11].

Технологія RASP (Runtime application self-protection) є вдосконаленням WAF. Вона вводить код захисту в прикладну програму, інтегрується з прикладною програмою, а також відстежує та блокує атаки в режимі реального часу, так що програма має власні можливості захисту. Захищеному додатку не потрібно вносити жодних змін у кодування, лише за допомогою простої конфігурації. Цей метод підходить для рішень динамічного захисту від виправлень для певних вразливостей. Вона поки не може самостійно та комплексно вирішити певний тип загрози безпеці [11].

Техніка виявлення атак з використанням мови структурованих запитів

Більшість методів виявлення атак – це методи, які використовуються веб-розробниками для виявлення атак, як правило, під час виконання коду JavaScript на стороні клієнта. Сьогодні існує багато методів, і більшість з них відрізняються один від одного у виявленні та запобіганні SQLIA за допомогою спеціальних інструментів і механізмів кодування для виявлення або запобігання зловмисних атак SQL на цільову базу даних на сервері. Перш ніж представити техніку виявлення, важливо показати дуже поширені типи вразливостей безпеки [12].

Тип I - перевірка введених даних – це спроба перевірити або відстежити будь-які підозрілі вхідні дані на предмет можливої зловмисної поведінки. Неправильна перевірка поля даних може призвести до багаторазового виконання шкідливого коду без належної та точної перевірки початкового наміру. Таким чином, зловмисник, використовуючи SQL, може захотіти скористатися неправильною перевіркою, щоб він міг виконати шкідливий код, а потім здійснити атаку на цільову базу даних.

Тип II - відсутність поділу між типами даних, які приймаються як вхідні.

Тип III - будь-яка затримка процесів до етапу виконання, оскільки наявні змінні вимірюються, незважаючи на те, що вихідний код використовує вираз для здійснення атаки.

Деякі статті в літературі навіть посилаються на збережені процедури як на засіб проти SQLIA. Оскільки збережені процедури знаходяться на передньому плані бази даних, запропоновані ними методи не можна застосовувати для захисту самих збережених процедур. Наприклад, деякі дослідження запропонували нову техніку захисту від атак, спрямованих на збережені процедури. Ця техніка поєднує статичний

аналіз коду програми з перевіркою під час виконання, щоб виключити такі атаки. Інші дослідники розробили метод, який виявляє та запобігає атакам SQL-ін’єкції, перевіряючи, чи введені користувачем зміни в результатах запиту. Вони запропонували метод виявлення атак ін'єкції SQL за допомогою токенізації запитів, який реалізується методом Query Parser, який використовує метод Black Box Testing, для тестування веб-додатків на наявність вразливостей ін'єкції SQL. Ця техніка використовує веб-сканер для визначення всіх точок у веб-додатку, які можна використовувати для введення SQLIA. Інші дослідники використовували техніку під назвою динамічний аналіз, де це дуже корисно для проведення аналізу часу виконання або динамічного SQL-запиту, він генерується за допомогою даних, що вводяться користувачем, спочатку шляхом реалізації веб-додатка. Техніка виявлення також у постгенерованій категорії може виконувати запит перед відправкою запиту до цільової бази даних на стороні сервера. Існує ще одна запропонована методика під назвою SQL-IDS, це підхід на основі специфікації для виявлення шкідливих вторгнень. Деякі дослідники SQLIA пропонують використовувати нову методологію, засновану на специфікації, а також на характеристиках експлуатації та виявлення шкідливих ін’єкцій SQL, щоб уникнути вразливостей. Виявлено, що виявлення, засноване на певному запиті, дозволяє системі виконувати аналіз, зосереджений на незначних обчисленнях без будь-якої необхідності створення хибно-негативних або хибно-позитивних результатів. У центрі уваги дослідження методів виявлення I типу, запропонована методика виявлення, яка називається Combined Detect, заснована на JavaScript, і два рішення, де перевірка в залежності від введених даних як спроба довести або відфільтрувати будь-який підозрілий вхід містить специфікацію зловмисної поведінки. Більшість методів показали, що прийнята методика виявлення недостатньої перевірки введення дозволить виконувати шкідливий код без належної перевірки його наміру. Тому будь-яка методика виявлення повинна перешкоджати зловмисникові скористатися перевагами недостатньої перевірки введення, які загрожують цільовій базі даних і допомагають зловмиснику використовувати шкідливий код SQL для легкого проведення атак.

В даному розділі виконано огляд теоретичних основ захисту веб-додатків.

Безпеці веб-додатків необхідно приділяти дуже багато уваги. Безпека веб-додатків залежить від якості програмного коду, від кваліфікації системного адміністратора та від компетенцій усіх користувачів, які мають доступ до чутливої інформації.

Вразливості OWASP є дуже шкідливими і в цьому списку зосереджені найнебезпечніші вразливості, які можуть коштувати купу грошей, або підрив ділової репутації, чи довести аж до втрати бізнесу.

Методи самозахисту веб-додатку під час виконання мають деякі складнощі з виявленням вразливостей без штучного налаштування, але з використанням машинного навчання можуть бути корисними у нагоді.

Техніки виявлення атак з використанням мови структурованих запитів є необхідними, однак фільтрація даних здається дуже ефективною та найпростішою технікою. Незважаючи на те, що виявлено, що методи виявлення SQLIA, які використовують перевірку введення, схильні до великої кількості хибно-позитивних результатів, і все ж немає 100% гарантії, що немає помилкових негативів.

Практична реалізація захищеної автоматизованої веб-системи тестування знань

Для розробки захищеної автоматизованої веб-системи тестування знань було використано мови програмування JavaScript та PHP, фреймворки Nuxt та Laravel. Для розробки використовувалось інтегроване середовище PhpStorm.

Архітектура програмного продукту

Структура програмного продукту є модульною. Функціональна структура додатку включає в себе наступні модулі:

- модуль автентифікації та реєстрації;
- модуль відновлення паролю;
- модуль публікацій;
- модуль тестувань знань;
- модуль груп;
- модуль користувачів;
- модуль управління особистою інформацією.

Модуль автентифікації та реєстрації дозволяє ідентифікувати користувачів по їх електронній адресі. Без цього модулю користувач не може бути ідентифікований, а це значить, що він не може використовувати функціонал додатку. Цей модуль, використовуючи API віддаленого серверу, відправляє запит на перевірку або створення нового облікового запису.

В успішному випадку сервер віддає токен, тобто деякий унікальний ключ. Унікальний ключ складається з 512 бітів інформації, тобто 64 символи по одному байту, що на практиці унеможливлює підбір цього ключа зловмисником. За допомогою унікального ключа користувач може здійснювати керування обліковим записом та отримувати інформацію з сервера, де необхідна автентифікація за токеном.

Модуль відновлення паролю дозволяє відновлювати пароль, якщо виникає така ситуація, коли користувач не пам’ятає пароль від облікового запису. На вказану електрону адресу посилається лист з токеном для відновлення паролю. Використовуючи цей токен, користувач може змінити пароль, але на протязі деякого часу, так як токен обмежений за часом.

Модуль публікацій дозволяю створювати публікації, переглядати та видаляти їх. Модуль є повністю незалежним від інших, тому з легкістю може переноситись на інші веб-додатки.

Модуль тестувань знань тестувань знань дозволяє створювати тестування з різними питаннями та відповідями, які можуть підтримувати математичні формули. Модуль підтримує перемішування питань та відповідей в питаннях. Є можливість завдання обмеження за часом, щоб проходження тестування було доступно лише в деяких проміжках часу. Також він дозволяє проходити тестування, переглядати вже пройдені тестування та бачити результати тестувань.

Модуль груп дозволяє створювати та видаляти групи, можливість додавати користувачів до груп. При використанні модуля груп надається можливість чітко визначати область видимості для тестувань знань, тобто надавати доступ до тестування лише для вказаних груп.

Модуль користувачів надає можливість перегляди усіх користувачів в системі, обмежувати або надавати доступ до системи користувачу, переглядати дії користувача. Є можливість змінювати профіль користувача, а саме змінювати ім’я, групу, роль, пароль. Використовуючи фільтр, надається можливість знаходити лише певних користувачів, які задовольняють умовам фільтру.

Модуль управління власною інформацією надає можливість лише змінювати пароль та ім’я в системі для усіх ролей в системі.

Технологія реалізації програмного продукту

Під час реалізації веб-додатку було виявлено чимало проблем як з клієнтської частиною, так із серверною.

Першою проблемою, що було виявлено в ході розробки, стало те, що додаток використовує фреймворк Nuxt, який в свою чергу використовує NodeJS збоку серверу. Майже всі додатки, що спершу роблять відображення контенту зі сторони серверу,

можуть мати проблеми з витіком пам’яті. Тобто це процес, при якому відбувається постійне зменшення доступної програмі оперативної пам'яті, причому програма не має інформації про більшу частину зайнятої пам'яті. При витіку пам’яті додаток рано чи пізно переходить до аварійного завершення, що означає, що користувачі можуть бачити помилку 502 з боку серверу та не можуть користуватися веб-додаток. Проблема була вирішена використанням PM2, що є менеджером виробничих процесів для додатків Node.js з вбудованим балансувальником навантаження. Це дозволяє підтримувати додатки назавжди, перезавантажувати їх без простоїв і полегшувати звичайні задачі системного адміністратора. Менеджер слідкує за станом веб-додатку, тому при або аварійному завершенні, або при витіку пам’яті за деяку межу, або при раптовому перезавантаженню серверу, веб-додаток буде перезавантажений.

Другою проблемою виявилося те, що коли користувач намагається відновити пароль та просить сервер, щоб він відправив листа до заданої електронної пошти, то ця операція відбувається синхронно, що є проблемою з точки зору великого відгуку від серверу. Вирішенням цієї проблеми було використано менеджер фонових завдань. Створювати завдання у чергу, які можуть опрацьовуватися у фоновому режимі. Переміщаючи трудомісткі завдання в чергу і виконуючи їх у фоні, програма може швидше обробляти веб-запити та швидше відповідати клієнту. Але в даному випадку, фонові завдання повинні оброблятися зі сторони автоматичного програмного забезпечення без відвідання будь-якої сторінки для їх запуску.

Автоматична обробка фонових завдань передбачає, що програма, яка оброблює завдання, також може потерпіти невдачу та аварійно завершитись. Отож для вирішення цієї проблеми було використано supervisor - клієнт-серверну систему, яка дозволяє контролювати низку процесів у операційних системах, подібних до UNIX.

Третьою проблемою є дуже повільна віддача статичного контенту веб-додатком, адже Nuxt призначен в першу чергу для візуалізацію з JavaScript коду на HTML представлення. Отож Nuxt дуже погано підходить в якості веб-серверу, тому було вирішено використовувати в якості прошарки NGINX, що є HTTP-сервером і може виконувати завдання зворотного проксі-серверу. Таким чином, коли користувач вперше потрапляє на сторінку, nginx оброблює запит та делегує завдання візуалізації через проксі-сервер до Nuxt. Натомість вся статика, яка буде проходити на сервер, буде оброблятися саме веб-сервером NGINX, що підвищує продуктивність серверу та економить ресурси серверу.

У відповідності до перелічених проблем узагальнена схема запитів до серверу зображена на рисунку 1.

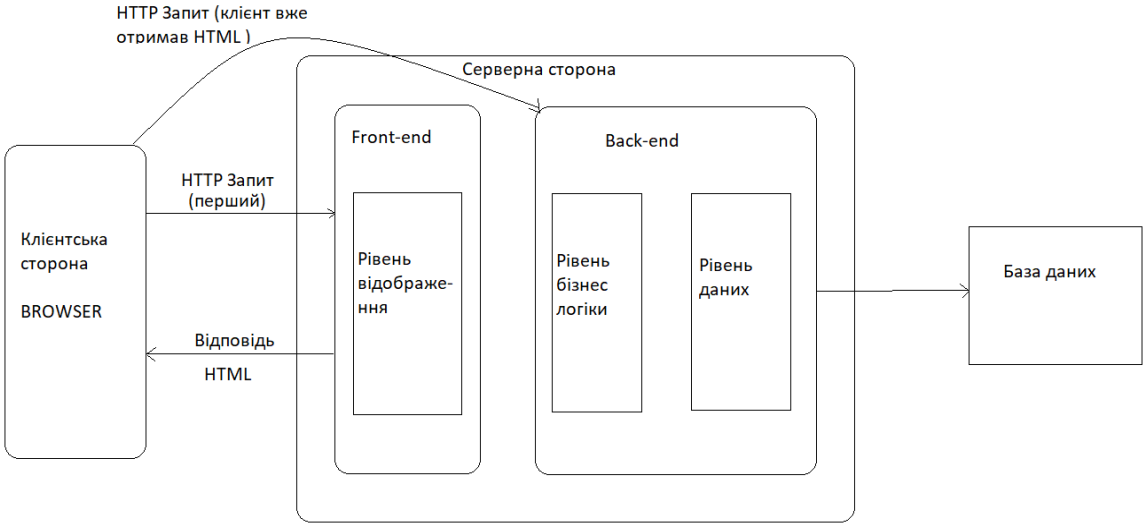

Рис. 1. Схема запитів до серверу с використанням візуалізації на стороні серверу

Висновки

В роботі виконана розробка та реалізація автоматизованої веб-системи тестування знань, рівень інформаційної безпеки якої, забезпечує захист інформації від найбільш критичних ризиків, встановлених у відповідності до OWASP Топ-10 2021. Розроблений програмний продукт може бути рекомендований для застосування в вищих та загальноосвітніх навчальних закладах, для підвищення якості знань та об'єктивності, завдяки застосуванню форми проведення тестування у автоматизованому форматі, з підвищеним рівнем інформаційної безпеки.

Список літератури

1. Mimirinis M. Qualitative differences in academics' conceptions of e-assessment. *Sci-hub*. URL:https://www.tandfonline.com/doi/abs/10.1080/02602938.2018.1493087
2. Li X. An examination of a gamified E-quiz system in fostering students' reading habit, interest and ability. URL: https://doi/10.1002/pra2.2018.14505501032
3. Alruwais N., Wills G., Wald M. Advantages and Challenges of Using e-Assessment. *International Journal of Information and Education Technology*. 2018. Vol. 8, No. 1. P. 34-37.
4. Kiryakova G. E-assessment-beyond the traditional assessment in digital environment. *International Conference on Technics, Technologies and Education 2020 (ICTTE 2020)*, 4-6 November 2020 Yambol, Bulgaria. P. 1-8.
5. Yin H. Taking e-Assessment Quizzes - A Case Study with an SVD Based Recommender System. *Sci-hub*. URL: https://doi/10.1007/978-3-030-03493-1
6. Mouri K. An automatic quiz generation system utilizing digital textbook logs. *Sci-hub*. URL:https://www.tandfonline.com/doi/abs/10.1080/10494820.2019.1620291
7. Petrova T., Ivanova M., Naydenova I. Evaluation of e-assessment: the students' perspective. *The 16th International Scientific Conference eLearning and Software for Education Bucharest*, 23-24 April 2020, Bucharest. P. 199-206.
8. Pevnev V., Popovichenko O., Tsokota Ya. Web application protection technologies. *Сучасні інформаційні системи*. 2020. Т.4, № 1. С. 119-123.
9. Demertzis K. Cognitive Web Application Firewall to Critical Infrastructures Protection from Phishing Attacks. *Journal of Computations & Modelling*. 2012. V.9. P. 1-26.
10. Bach-Nutman M. Understanding The Top 10 OWASP Vulnerabilities. *Ar Xiv*. URL: https://arxiv.org/ftp/arxiv/papers/2012/2012.09960.pdf
11. Zhongxu Y., Zhufeng L., Yan C. A Web Application Runtime Application Self-protection Scheme against Script Injection Attacks. *4th International Conference, ICCCS 2018*, 8-10 June 2018, Haikou, China. P. 566-577.
12. Rua M.T., Musab A.M., Farooq B.A. The impact of sql injection attacks on the security of databases. *6th International Conference on Computing and Informatics, ICOCI 2017*, 25-27 April, 2017, Korea. P. 323-331.

DEVELOPMENT OF A SECURE AUTOMATED WEB-BASED KNOWLEDGE TESTING SYSTEM

S.O. Pererva, K.O. Tryfonova

National Odessa Polytechnic University
1, Shevchenko, Odessa, 65044, Ukraine; e-mail: katikkatik@gmail.com

Modern education is aimed at providing an opportunity to acquire quality knowledge and skills. Many studies in recent years demonstrate the significant contribution of information technology to improving the quality of acquired knowledge and skills in all areas of education. To assess the quality of the knowledge gained during the educational process, it is necessary to carry out control in various forms. Significant pedagogical research proves the effectiveness of test control to obtain an assessment of knowledge and skills. Information technology can help increase the independence of testing and assessment from the personality of the teacher, that is, increase objectivity through the use of an automated test form. In accordance with modern realities, the development, implementation and subsequent use of automated software for testing knowledge takes place on the Internet. In this regard, information security is a factor that must be identified as the most important. Therefore, the purpose of this work is to develop and implement an automated web-based knowledge testing system with an increased level of information security. One of the most popular open source web application security research projects, the Open Web Application Security Project, publishes the top ten most common threats annually. The paper presents the development of a software protection mechanism, the level of information security of which ensures the protection of information from the most critical risks established in accordance with the OWASP Top 10 2021. The developed software product can be recommended for use in higher and general educational institutions, to improve the quality of knowledge and objectivity through the use of a test form in an automated format with an increased level of information security.

Keywords: information protection, web applications, distance learning, the quality of education, knowledge testing