УДК 004.422.81+004.774
DOI: https://doi.org/10.17721/ISTS.2024.8.17-25

Сергій БУЧИК, д-р техн. наук, проф.
ORCID ID: 0000-0003-0892-3494
e-mail: buchyk@knu.ua
Київський національний університет імені Тараса Шевченка, Київ, Україна

Андрій КУРОЄДОВ, студ.
ORCID ID: 0009-0007-5811-4798
e-mail: askuroyedov@gmail.com
Київський національний університет імені Тараса Шевченка, Київ, Україна

МОДЕЛЬ АНАЛІЗУ ВЕБЕКСПЛОЙТІВ НА ОСНОВІ JAVASCRIPT

Вступ. *Задача забезпечення безпеки вебдодатків і серверів залишається актуальною в умовах постійного зростання кількості атак у кіберпросторі. Використання різних систем керування вмістом із відкритим кодом (наприклад WordPress, Joomla, Open Journal Systems, Drupal), які через свою простоту в установленні та використанні, є доволі популярними для створення вебсайтів, на жаль вимагають постійного оновлення не тільки для покращення за змістом, але і для забезпечення безпеки системи. Автори статті зосередили увагу саме на системі WordPress, хоча цей підхід може бути використаний і для інших систем. Матеріал статті підкреслює значення раннього виявлення вразливостей для запобігання потенційним кіберзагрозам та їх негативним наслідкам. Запропоновано модель і скрипт, які призначені для прискорення виявлення вразливостей у додатках WordPress. Автоматизація процесу сканування за допомогою власного скрипту дозволяє швидко виявляти вразливості, забезпечуючи оперативне виправлення й оновлення. Такий підхід не лише зміцнює безпеку, але і сприяє збереженню репутації вебсайтів та брендів, що є критично важливим у сучасному цифровому середовищі.*

Методи. *Використано методи аналізу вебексплойтів на основі JavaScript з урахуванням загальних принципів їх аналізу та з огляду на методології аналізу вебдодатків на вразливості.*

Результати. *Представлено вдосконалену модель аналізу вебдодатка на CMS WordPress, в основі якої є скрипт, який забезпечує автоматизоване сканування вебдодатка за допомогою запуску таких утиліт: NMAP, Dirb, Nikto, SQLMap, WPScan і PwnXSS. Усі результати записують в окремий файл для подальшого вивчення всіх знайдених проблем безпеки вебдодатка.*

Висновки. *Розроблена модель і скрипт повинні допомогти розробникам і тестувальникам прискорити процес виявлення вразливостей у WordPress, оскільки вони можуть запустити один скрипт і через короткий термін часу отримати об'ємний і змістовний звіт із виявленими вразливостями. За рахунок цього оптимізується виявлення вразливостей за допомогою автоматизованого запуску сканерів.*

Ключові слова: *вразливість, вебсайт, вебдодаток, вебексплуатація, аналіз вебдодатків, пошук вразливостей. SQL, XSS, CSRF.*

Вступ

В сучасному цифровому світі вебексплойти стали однією з найсерйозніших загроз для безпеки вебдодатків та вебінфраструктури. Ця проблема виникає з того, що вебексплойти дають можливість зловмисникам використовувати вразливості вебдодатків для виконання шкідливого коду або отримання несанкціонованого доступу до системи. Методи аналізу вебексплойтів на основі JavaScript є надзвичайно важливими, оскільки JavaScript є однією з основних мов програмування для веброзробки, і відповідно, використовується практично на кожній вебсторінці.

JavaScript, як основна мова програмування для веброзробки, проникає практично в кожен аспект інтернет-простору. Від створення динамічного змісту до взаємодії з користувачем, JavaScript є невід'ємною складовою будь-якого сучасного вебдодатка. Ця широка поширеність робить JavaScript дуже привабливою мішенню для зловмисників, які шукають можливості використовувати вразливості цієї мови для здійснення атак.

Атаки на основі JavaScript можуть мати різноманітні форми і наслідки. Вони можуть включати впровадження шкідливого коду на сторінках вебсайтів для отримання конфіденційної інформації користувачів, викрадення сесійних файлів, а також підміну вмісту сторінок для поширення фішингових атак або розповсюдження шкідливого програмного забезпечення.

Захист вебдодатків і серверів – це багатогранна задача, що поєднує в собі збереження конфіденційності даних, безперебійну роботу ресурсів і запобігання фінансовим втратам. Нехтування безпекою може призвести до серйозних наслідків, таких як втрата репутації, юридичні проблеми та збитки. Дослідження методів аналізу вебексплойтів JavaScript є ключовим фактором у забезпеченні безпеки вебдодатків та інфраструктури в цифрову епоху. Ця тема потребує постійного розвитку та вдосконалення для гарантування безпеки в онлайн-середовищі (Common JavaScript Vulnerabilities and How They Manipulate Data (2022), https://www.preemptive.com/blog/a-review-/on-javascript-security-in-2022/).

Мета дослідження – аналіз існуючих методів і моделей аналізу вебдодатків, побудова власної, удосконаленої моделі аналізу вебексплойтів на основі JavaScript.

Об'єктом дослідження є процес аналізу вебексплойтів на основі JavaScript.

Предметом дослідження є методи та моделі аналізу вебексплойтів на основі JavaScript.

Отже, основне завдання полягає в удосконаленні моделі аналізу вебексплойтів на основі JavaScript, яка має пришвидшити процес виявлення вразливостей у вебдодатках, які побудовані на CMS Wordpress.

Огляд літератури. Атаки на вебдодатки – це зловмисна діяльність, спрямована на використання вразливостей у конструкції або реалізації програм, що працюють через веб. Ці атаки можуть призвести до незаконного доступу, крадіжки даних або інших шкідливих наслідків.

Загальні принципи веббезпеки та поширені вразливості, зокрема і пов'язані з JavaScript розглянуто в (Liang, 2014) та (Stuttard, & Marcus Pinto, 2008).

ISSN 2707-1758

Особливо варто відмітити книгу (Hoffman A., 2024) в якій пояснено наступальні та захисні прийоми кібербезпеки. Автор вважає, що "освоївши концепції, а не прийоми використання платних інструментів, будь-який користувач зможе переходити від одного інструменту до іншого або створювати свої інструменти, виявляти вразливості та вживати заходів щодо пом'якшення наслідків".

З огляду на вказане автори, зосередили увагу як на використанні інструментів, так і на поєднанні певної концепції, шляхом автоматизації самого процесу виявлення вразливостей.

В ході викладення опису результатів, щоб дотримативсь логічної послідовності подання матеріалу, автори додатково здійснюють огляд літератури.

Методи

Використано методи аналізу вебексплойтів на основі JavaScript з урахуванням загальних принципів їх аналізу та з огляду на методології аналізу вебдодатків на вразливості.

Результати

Найпоширеніші методи атак JavaScript включають виконання шкідливих сценаріїв, викрадення даних сесії користувача або даних із локального сховища вебпереглядача, спонукання користувачів до виконання непередбачуваних дій та експлуатацію вразливостей у вихідному коді вебпрограм – рис. 1 (Web Exploitation, 2024: https://devopedia.org/web-exploitation).

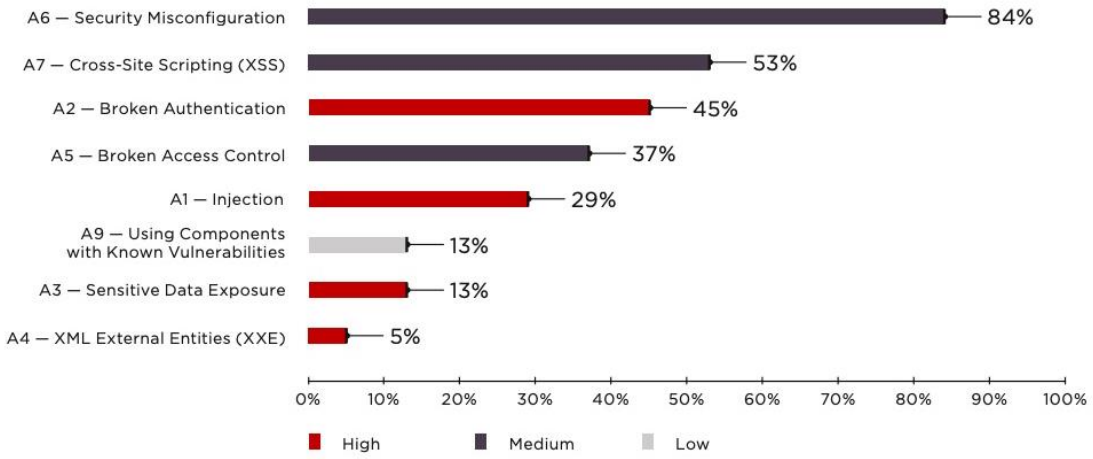

Рис. 1. Найпоширеніші вебатаки

Нижче наведено найпоширеніші атаки на вебдодатки (The 7 Most Common Web Application Attacks And How A WAF Can Prevent Them, 2024, https://www.nexusguard.com/blog/the-7-most-common-web-application-attacks-and-how/-a-waf-can-prevent-them); 8 Types of Web Application Attacks and Protecting Your Organization, 2023, https://brightsec.com/blog/8-types-of-web-application-attacks/-and-protecting-your-organization/).

Однією з найпоширеніших вебатак є *SQL-ін'єкція*. Зловмисники можуть вставляти шкідливий SQL-код у поля введення вебдодатків, що може призвести до несанкціонованого доступу до бази даних. Заходи захисту, такі як вебзахист від вторгнень (WAF), можуть блокувати підозрілі SQL-запити та використовувати методи відповідності шаблонів для виявлення та блокування спроб SQL-ін'єкцій. Ще однією вразливістю є *неправильна конфігурація*, яка виникає, коли налаштування належно не зберігають під час ручного виконання процесів. Це може створювати ризики безпеки та вразливості в системі.

Міжсайтовий сценарій (XSS) є ще однією серйозною проблемою. Він виникає, коли сервер приймає зловмисний код JavaScript через введення користувача та повертає його у відповідь, що призводить до виконання цього коду в браузері, що може мати негативні наслідки для безпеки вебдодатка.

Міжсайтова підробка запитів (CSRF) – це тип атаки на вебпрограму, яка шахрайським шляхом змушує користувача виконати небажану дію з вебпрограмою, в якій він уже автентифікований. Цього часто досягають, надсилаючи спеціально створене посилання або код користувачеві, який потім виконує небажану дію після виконання. Наприклад, атака CSRF може бути використана для проведення несанкціонованих операцій, таких як купівля або зміни налаштувань облікового запису. Щоб запобігти атакам CSRF, можна використовувати анти-CSRF маркери, що є унікальними ідентифікаторами, які генеруються вебпрограмою для кожного сеанса користувача і мають включатися в кожний запит до програми.

Щодо застарілого програмного забезпечення, з огляду на зростання використання пакетів програмного забезпечення з відкритим кодом і сторонніх програм, важливо постійно оновлювати їх. Використання застарілого програмного забезпечення може стати загрозою, особливо якщо вразливості стають загальнодоступними.

HTTP request smuggling. Транспортування HTTP-запитів використовує невідповідність у аналізі невідповідних HTTP-запитів, яка виникає у разі обміну їх між двома HTTP-пристроями, зазвичай між сервером і брандмауером, що підтримує HTTP, або зовнішнім проксісервером. Процес транспортування HTTP-запитів відбувається за допомогою створення кількох налаштованих HTTP-запитів, які дозволяють двом цільовим сутностям бачити дві різні серії запитів.

Заголовок HTTP пропонує два різні способи вказати, де закінчується запит: заголовок Transfer-Encoding і заголовок Content-Length. Уразливість транспортування HTTP-запитів виникає, коли зловмисник надсилає обидва заголовки в одному запиті. Це може призвести до того, що зовнішній або внутрішній сервер неправильно інтерпретує запит через зловмисний HTTP-запит. Уразливості транспортування запитів

ISSN 2707-1758

використовують кіберзлочинці, щоб обійти заходи безпеки, отримати доступ до конфіденційної інформації та для прямого компрометування користувачів різних програм. Можна також використовувати цю вразливість для вторинних експлойтів, наприклад, обхід брандмауерів, часткове отруєння кешу та міжсайтовий сценарій.

Атака відмови в обслуговуванні (DDoS) – це спосіб атаки на вебпрограму, що полягає в перевантаженні програми великою кількістю трафіка з різних джерел, таких як ботнети або скомпрометовані пристрої. Це може призвести до того, що вебпрограма стане недоступною для легітимних користувачів. DDoS-атаки можна уникнути за допомогою пристроїв мережної безпеки – брандмауерів і систем запобігання вторгненням, які можуть виявляти та блокувати шкідливий трафік. До того ж розробники вебдодатків можуть скористатися мережами доставки контенту (CDN) та балансувальниками навантаження для розподілу трафіка між декількома серверами з метою пом'якшення наслідків DDoS-атак.

XML External Entity (XXE) – це форма атаки на вебдодатки, що використовує вразливості у парсерах XML, що застосовані у додатку. Це може дозволити зловмиснику отримати доступ до конфіденційних даних або виконати несанкціоновані дії на сервері вебдодатка. Часто атаки XXE включають упровадження спеціально створених корисних даних XML, які використовують можливість синтаксичного аналізатора XML читати зовнішні сутності. Цим атакам можна запобігти, вимкнувши синтаксичний аналіз зовнішніх об'єктів або використовуючи захищені аналізатори XML, які належно очищують вхідні дані.

Автентифікація та авторизація. Ідентифікатор сесії може бути розкритим через URL-адресу. Пароль може бути незашифрованим. Якщо тайм-аут реалізовано неправильно, можливе викрадення сесії. Доступ до неавторизованих ресурсів можливий, навіть якщо інтерфейс користувача не розкриває їх.

Прямі посилання на об'єкти. Через поганий дизайн або помилку кодування прямі посилання можуть бути доступні для клієнтів. Наприклад, запит GET на download.php?file=secret.txt може обійти авторизацію та дозволити пряме завантаження захищеного файлу. Іншим прикладом є пряме скидання пароля адміністратора.

Brute Force. Атака грубою силою – це автоматичний метод вгадування комбінації імені користувача та пароля для отримання несанкціонованого доступу до вебпрограми. Зловмисники використовують програмні інструменти, щоб спробувати різні комбінації імен користувачів і паролів, поки не отримають доступ. Щоб убезпечити себе від атак грубої сили, вебдодатки можуть упроваджувати стратегії контролю швидкості та блокування облікових записів. Контроль швидкості обмежує кількість спроб входу з однієї IP-адреси, тоді як блокування облікового запису тимчасово призупиняє доступ до облікового запису після певної кількості невдалих спроб входу.

Розкриття даних. Одним із ризиків є витікання конфіденційних даних, які можуть зберігатися в незашифрованому вигляді або бути відкритими у файлах cookie чи URL-адресах під час взаємодії між клієнтом і сервером без використання HTTPS.

В роботі (Noman, Iqbal, & Manzoor, 2020), а також у (Rokia, & El Habib, 2022) описано, що всі вразливості можна поділили на три групи: неправильна перевірка введених даних, неправильне керування сеансами, неналежна автентифікація та авторизація.

До першої групи належать такі вразливості: маніпуляція / додавання запитів, упровадження коду на стороні клієнта, упровадження файлів у вебдодаток. Загалом для експлуатації вразливостей цієї групи, можна здійснити такі атаки, як SQL Injection, NoSQL Injection, Xpath and LDAP Injection, Cross-site scripting, Remote document local record consideration, Path / Directory Injection and Remote Code infusion.

До другої групи відносять вразливість керування сеансами. Для експлуатації цієї вразливості можна впровадити атаку Cross-site request forgery (CSRF).

До третьої групи належить помилка в логіці коду. Відповідно, можуть бути впроваджені такі атаки: Unreliable Direct Object Reference, missing Functional access Control, Invalidated Redirects and Forwards or application rationale susceptibilities.

Існують спеціальні інструменти для аналізу вразливостей до поширених атак на вебдодатки.

Acunetix Vulnerability Scanner – це комплексний і автоматизований інструмент для виявлення вразливостей у програмному забезпеченні. Він використовує методи аналізу Black-Box і Gray-Box для виявлення різноманітних проблем безпеки та може бути розгорнутий як у хмарі, так і на стороні клієнта. Acunetix здатний ідентифікувати і повідомляти про різноманітні вразливості в програмах, що побудовані на різних платформах, таких як WordPress, PHP, ASP.NET, Java Framework, Ruby on Rails тощо. Інструмент має широкий набір можливостей і для автоматизованого, і для ручного тестування, що дозволяє оцінити й усунути виявлені проблеми безпеки. Система Acunetix розрахована на роботу з багатьма користувачами і забезпечує доступ лише до необхідних ресурсів, що сприяє командній гнучкості та підвищує продуктивність (Attack Simulation and Vulnerability Management, 2024, https://itbiz.ua/proizvoditeli/acunetix/acunetix-web-vulnerability/-scanner-standard-premium-360/).

AppSpider – це рішення динамічного тестування безпеки додатків, яке здатне сканувати веб- та мобільні додатки щодо наявності вразливостей. Основною технологією, що використовується в AppSpider, є універсальний перекладач, який адаптується до новітніх технологій, таких як AJAX, HTML5 і JSON, що застосовуються у сучасних веб- та мобільних додатках, і проводить сканування і традиційних, і новітніх програм (Welcome to AppSpider, 2024, https://docs.rapid7.com/appspider/).

OWASP ZAP – це інструмент, який дуже простий у використанні для проведення тестів на проникнення програмою, а також для виявлення вразливостей у вебдодатках. OWASP ZAP проводить тестування на проникнення вебдодатка та дозволяє виявляти такі атаки, як SQL injection, XSS, clickjacking тощо (Introduction to OWASP ZAP, 2024, https://medium.com/@lavanya.agre.cyb/introduction-to-owasp-zap-bdc58293005f).

Nmap чи Network Mapper – це відкритий інструмент, призначений для вивчення мережі та перевірки її безпеки. Початково він був розроблений для швидкого сканування великих мереж, але також ефективно застосовується для аналізу окремих цілей. Nmap використовує IP-пакети у специфічний спосіб для визначення доступних хостів у мережі, надаючи інформацію про надані ними послуги (назву та версію програм), операційні системи та їхні версії, типи використаних пакетних фільтрів / брандмауерів та інші характеристики. Nmap часто застосовують для оцінювання безпеки, а також для контролю структури мережі та керування розкладами запуску служб та обліку часу

роботи хостів або служб (Nmap (Man Page), 2024, https://nmap.org/book/man.html).

Metasploit Framework – це дуже потужний інструмент, який використовують як кіберзлочинці, так і етичні хакери для виявлення системних уразливостей у мережах та на серверах. Оскільки це фреймворк з відкритим кодом, його можна легко налаштувати та використовувати на більшості операційних систем (What is Metasploit? The Beginner's Guide, 2024, https://www.varonis.com/blog/what-is-metasploit).

SQLMap – це інструмент для тестування на проникнення. Інструмент має вбудований механізм виявлення із спеціалізованими можливостями для досвідчених тестерів на проникнення та широким спектром параметрів для отримання детального звіту з тестування на проникнення. SQLMap може використовуватися зловмисниками для проведення SQL-ін'єкцій на потрібні додатки за допомогою різних методів, таких як логічні значення, часові проміжки, помилки, UNION-запити, стековані запити та позасмугове впровадження (SQlMap: automatic SQL injection and database takeover tool, 2024, https://sqlmap.org/).

Дослідження вебдодатків на вразливості нині – це важливе завдання, яке дозволить вебпрограмам належно функціонувати. Це тестування виконують розробники та тестувальники ПЗ, які зазвичай використовують уже існуючі та перевірені часом методики. До таких методологій можна віднести (5 Most Popular Web App Security Testing, 2024, https://www.apriorit.com/qa-blog/524-web-application-security-testing):

- Open Source Security Testing Methodology Manual (OSSTMM);
- Open Web Application Security Project (OWASP);
- Web Application Security Consortium Threat Classification (WASC-TC);
- Penetration Testing Execution Standard (PTES);
- Information Systems Security Assessment Framework (ISSAF);
- PCI Penetration Testing Guide;
- NIST Special Publication 800-115;
- MITRE ATT&CK.

Усі вказані методології – потужні інструменти та техніки, які допомагають у різних сферах упровадження безпеки. Проаналізуємо наскільки ефективними будуть ці методики для вебдодатків, побудованих на CMS WordPress.

OSSTMM може бути корисною для глибокого тестування безпеки.

Методологія OWASP надає широкий перелік рекомендацій і кращих практик із покращення безпеки вебдодатків. З огляду на широке використання платформ WordPress та OpenCart, рекомендації OWASP можуть бути дуже корисними для ідентифікації та уникнення загроз безпеці.

Методологія WASC TC допомагає виявити специфічні загрози, що можуть виникнути в контексті застосування WordPress та OpenCart.

Використання PTES дозволяє ідентифікувати й експлуатувати вразливості, що допоможе покращити безпеку вебдодатків на платформах WordPress та OpenCart.

Розглянемо сценарій, де власник онлайн-магазину на платформі OpenCart прагне підвищити рівень безпеки свого вебдодатка. Використовуючи ISSAF, він може провести аналіз потенційних загроз і ризиків, що можуть виникнути внаслідок недоліків у безпеці мережі, конфігурації сервера або програмного забезпечення. За допомогою цього аналізу він може ідентифікувати критичні вразливості і розробити стратегії для їхнього усунення.

Використовуючи NIST Special Publication 800-115, власник магазину може впровадити стандартизовані методики тестування безпеки для проведення тестування на проникнення й аудиту безпеки. Це дозволить виявити можливі вразливості в додатку й інфраструктурі магазину, такі як SQL-ін'єкції, XSS-атаки, недоліки в конфігурації сервера тощо.

Для створення моделі аналізу вебдодатків обрано середовище для тестування – CMS Wordpress. Через велику кількість користувачів WordPress стає привабливою мішенню для зловмисників, оскільки потенційної шкоди можна завдати багатьом вебсайтам одночасно. Зауважимо, що CMS WordPress містить безліч плагінів і тем, що розширюють функціонал системи.

Розробники плагінів і тем не завжди дотримуються найвищих стандартів безпеки, що може призводити до появи вразливостей у вебдодатку. До того ж активний розвиток WordPress і його постійні оновлення можуть приховувати недоліки безпеки, які тестувальники можуть виявити перед релізом нових версій. Нарешті, широкий функціонал платформи, такий як можливість створювати власні додаткові поля або налаштовувати права доступу, створює додаткові можливості для потенційних вразливостей.

Для проведення дослідження був установлений Wordpress на Ubuntu 22.04. Для тестування встановлено вразливу тему для Wordpress (рис. 2) з github (https://github.com/vavkamil/dvwp).

До плагінів належать:

- Infinite WP client – версія 1.9.4.4;
- Social Warfare – версія 3.5.2;
- Wordpress File Upload – версія 4.12.2;
- WP-Advanced-Search – версія 3.3.3;
- Askimet Anti-Spam – версія 4.1.3;
- Backup and Staging by WP Time Capsule – версія 1.21.15;
- Hello Dolly – версія 1.7.2.

Wordpress містить безліч плагінів для аналізу вразливостей у вебдодатку, проте ці плагіни можуть і самі бути вразливими.

Коли йдеться про аналіз вразливостей у вебдодатках, створення власного скрипту може бути кращим рішенням, ніж використання плагінів у WordPress, тому що власний скрипт може бути розроблений під ваш вебдодаток, що дозволить врахувати особливості цього вебдодатка. Зазначимо, що при розробленні і використанні власного рішення, ви можете мати повний контроль над кодом, що дозволяє гарантувати безпеку вашого рішення, оскільки ви не залежите від сторонніх плагінів та бібліотек і використовуєте свої функції. Застосовуючи власний скрипт, можна оперативно виправляти вразливості, які можуть бути знайдені.

Отже, власний скрипт для аналізу вразливостей може бути надійнішим та ефективнішим варіантом, ніж використання плагінів у WordPress.

Модель аналізу вебдодатка на CMS Wordpress зображено на рис. 3. Нижче опишемо її.

1. Початкова розвідка вебдодатка та сканування всіх існуючих сервісів, портів і служб. Використовуючи інструменти, такі як nmap, скрипт зможе сканувати порти та визначати відкриті служби.

2. Аналіз вебсайту на вразливості. Скрипт зможе використовувати інструменти, такі як Nikto або OWASP ZAP, для сканування вебсайту на наявність відомих вразливостей.

3. Генерація звіту. Скрипт зможе створювати звіт із виявленими вразливостями та рекомендаціями щодо їхнього виправлення.

ISSN 2707-1758

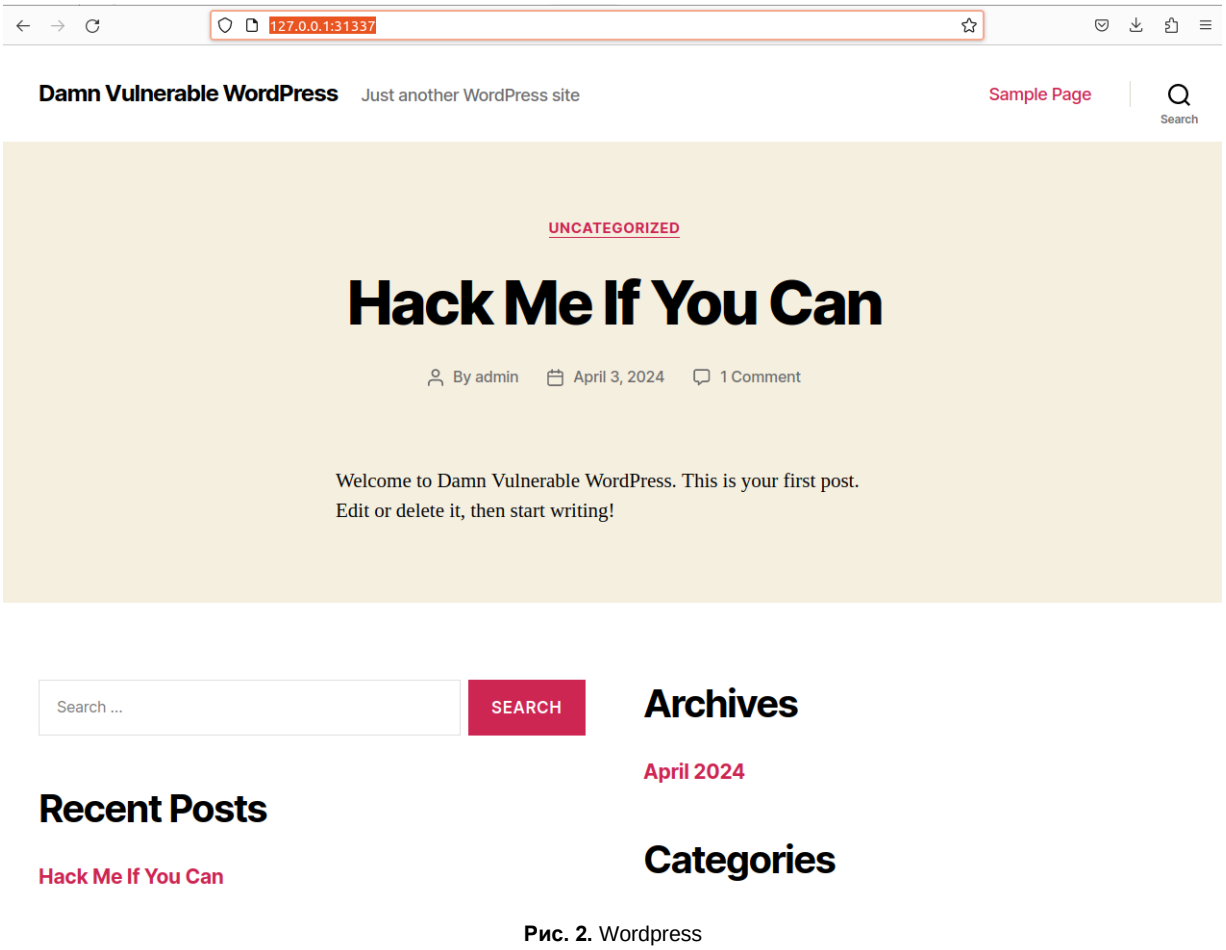

Рис. 2. Wordpress

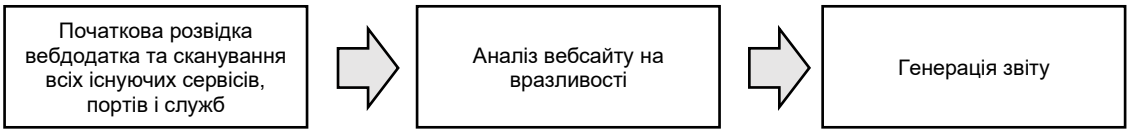

Рис. 3. Модель аналізу вебдодатка на CMS Wordpress

На першому етапі моделі використаємо інструмент NMap і Dirb для пошуку файлів / директорій, до яких має доступ звичайний користувач.

На другому етапі застосуємо інструменти для пошуку вразливостей, наприклад, WPScan, nikto, SQLmap, PwnXSS.

WPScan – це сканер, який широко використовують спеціалісти з кібербезпеки, які бажають визначити всі можливі слабкі місця в CMS WordPress. Сканер дозволяє виявляти вразливості у версіях систем, у плагінах, а також темах. Отже, використання такого програмного забезпечення – це спосіб забезпечити надійну безпеку вебдодатка.

До того ж WPScan здатний проводити брутфорс-атаку на WordPress. Використовуючи додаткові словники з даними, можна підібрати дані для авторизації.

На третьому етапі потрібно згенерувати звіт з усіма результатами. Звіт має містити детальний результат тестування вебдодатка за допомогою всіх згаданих вище утиліт і програм.

У віртуальному середовищі Oracle Virtual Box створено й налаштовано віртуально ОС Ubuntu 22.04, де було написано bin / bash скрипт для аналізу вебдодатка WordPress, про який вказано раніше. Цей скрипт забезпечує автоматизоване сканування вебдодатка за допомогою запуску утиліт, таких як NMAP, Dirb, Nikto, SQLMap, WPScan та PwnXSS. Зауважимо, що всі результати записують в окремий файл для подальшого вивчення всіх знайдених проблем безпеки вебдодатка.

На рис. 4 продемонстровано код скрипту.

На першому етапі отримано результати сканування NMap і Dirb. За допомогою nmap знайдено відкриті порти та служби (рис. 5).

Завдяки Dirb було знайдено приховані директорії та файли, до яких має доступ звичайний користувач (рис. 6).

Наступний крок – це сканування за допомогою утиліти Nikto.

Під час сканування вебдодатка утиліта виявила багато можливих вразливих місць, які здатні порушити безпеку додатків, наприклад, OSVDB 3092, 2695.

Знайдені OSVDB вразливості включають вразливість бібліотеки / плагіна "My Photo Gallery pre 3.6". Також за допомогою Nikto знайдено доступні директорії для звичайного користувача, які зловмисник може використати для проникнення в систему (рис. 7).

ISSN 2707-1758

```
 GNU nano 6.2                                  script.sh *
#!/bin/bash


echo "Start nmap"

nmap -p- -sV -sC -A 127.0.0.1 | tee -a "scan-results.txt"
echo "Finished"


echo "Start dirb"

dirb http://127.0.0.1:31337/ -f | tee -a "scan-results.txt"
echo "Finished"

echo "Start Nikto"

nikto -h http://127.0.0.1:31337/ | tee -a "scan-results.txt"
echo "Finished"


echo "Start SQLmap"

sqlmap -u 127.0.0.1:31337/?s=gngh | tee -a "scan-results.txt"
echo "Finished"


echo "Start WPSCAN"

wpscan --url http://127.0.0.1:31337/ --api-token IAph8RXF1kSeypxDAsXDVrmPr3AaFsHXimatV0YeeS4 | tee -a "scan-results.txt"
echo "Finished"

echo "Start PwnXSS"

python3 PwnXSS/pwnxss.py -u http://127.0.0.1:31337/ | grep -i "WARNING" | tee -a "scan-results.txt"
echo "Finished"


fi

exit 0
```

Рис. 4. Вихідний код скрипту

```
PORT      STATE SERVICE         VERSION
80/tcp    open  http            Apache httpd 2.4.52 ((Ubuntu))
|_http-server-header: Apache/2.4.52 (Ubuntu)
|_http-title: Welcome to Test.com!
631/tcp   open  ipp             CUPS 2.4
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: CUPS/2.4 IPP/2.1
|_http-title: Home - CUPS 2.4.1
3306/tcp  open  nagios-nsca     Nagios NSCA
| mysql-info:
|   Protocol: 10
|   Version: 8.0.36-0ubuntu0.22.04.1
|   Thread ID: 346279
|   Capabilities flags: 65535
|   Some Capabilities: Support41Auth, LongPassword, DontAllowDatabaseTableColumn, SupportsTransactions, LongColumnFlag, Sp
eaks41ProtocolOld, SupportsLoadDataLocal, SupportsCompression, InteractiveClient, Speaks41ProtocolNew, FoundRows, SwitchTo
SSLAfterHandshake, IgnoreSigpipes, IgnoreSpaceBeforeParenthesis, ODBCClient, ConnectWithDatabase, SupportsAuthPlugins, Sup
portsMultipleStatments, SupportsMultipleResults
|   Status: Autocommit
|   Salt: s     uU"gMb3x+\x1E%0\x015\x06\x7Fkr
|_  Auth Plugin Name: caching_sha2_password
31337/tcp open  hadoop-datanode Apache Hadoop 2.4.38 ((Debian))
| hadoop-datanode-info:
|_  Logs: http://127.0.0.1:31337/wp-login.php
| hadoop-tasktracker-info:
|_  Logs: http://127.0.0.1:31337/wp-login.php
|_http-generator: WordPress 5.3
|_http-title: Damn Vulnerable WordPress &#8211; Just another WordPress site
|_http-trane-info: Problem with XML parsing of /evox/about
31338/tcp open  http            Apache httpd 2.4.57 ((Debian))
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Apache/2.4.57 (Debian)
|_http-title: phpMyAdmin
33060/tcp open  mysqlx?
```

Рис. 5. Результати сканування NMap

```
Start dirb

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Apr 17 22:08:20 2024
URL_BASE: http://127.0.0.1:31337/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Fine tunning of NOT_FOUND detection

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://127.0.0.1:31337/ ----
+ http://127.0.0.1:31337/cgi-bin/ (CODE:200|SIZE:1633)
+ http://127.0.0.1:31337/favicon.ico (CODE:200|SIZE:0)
+ http://127.0.0.1:31337/index.php (CODE:301|SIZE:0)
+ http://127.0.0.1:31337/info.php (CODE:200|SIZE:6098)
+ http://127.0.0.1:31337/php.ini (CODE:200|SIZE:2)
+ http://127.0.0.1:31337/server-status (CODE:403|SIZE:277)
+ http://127.0.0.1:31337/xmlrpc.php (CODE:405|SIZE:5)

-----------------
END_TIME: Wed Apr 17 22:13:15 2024
DOWNLOADED: 4612 - FOUND: 7
Finished
```

Рис. 6. Результати сканування Dirb

ISSN 2707-1758

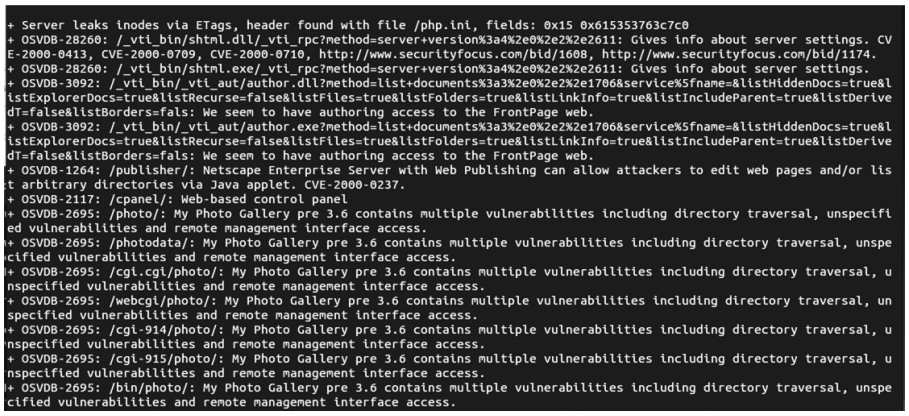

Рис. 7. Результати сканування Nikto

Наступний інструмент для виявлення можливих вразливостей – це SQLMap. Під час тестування не виявлено вразливостей (рис. 8).

Відтак запускається утиліта WPScan. Це дуже потужне програмне забезпечення, яке надає багато можливостей у дослідженні безпеки вебдодатка і не тільки. Сканер ідентифікував 52 різні вразливості, які потенційно можуть завдати шкоди для вебдодатка на WordPress. Виявлено вразливості встановлених плагінів і теми Wordpress (рис. 9).

Рис. 8. Результати сканування SQLMap

Рис. 9. Результати сканування WPScan

Останнім інструментом у дослідженні безпеки вебдодатка на WordPress є програмне забезпечення PwnXSS, яке тестує додаток щодо наявності можливих вразливостей міжсайтового сценарію XSS. Під час сканування він виявив потенційно цікаві посилання з різними ідентифікаторами, які можуть налічувати вразливості XSS (рис. 10).

Рис. 10. Результати сканування PwnXSS

Усі результати зберігають в окремому файлі, де зручно досліджувати й аналізувати знайдені вразливості вебдодатка. Отже, в результаті сканування вебдодатка, розробник або тестувальник, який запускає цей скрипт, може знайти чимало цікавої інформації про свій вебдодаток. Знайдені вразливості частково описуються самими сканерами, але не всі. Деякі знайдені вразливості потрібно досліджувати, тому що не обов'язково всі знайдені потенційно вразливі точки вебдодатка можуть бути вразливі. Найкориснішим цей сканер буде для вебдодатків, побудованих на CMS Wordpress, оскільки утиліта WPScan знаходить вразливі версії плагінів. Тому, якщо даний сканер знайшов вразливі плагіни, то їх необхідно терміново оновити, щоб зловмисники не скористались цими вразливостями до того, як ви їх виправите.

Дискусія і висновки

Атаки на основі JavaScript можуть мати різні форми і наслідки, від впровадження шкідливого коду до крадіжки конфіденційної інформації. Забезпечення безпеки вебдодатків і серверів є критично важливою задачею, оскільки недбалість у цьому питанні може призвести до серйозних наслідків, таких як втрата репутації і фінансові втрати.

Виявлення вразливостей у системі керування контентом WordPress на ранніх етапах має критичне значення з кількох причин. По-перше, це допомагає підвищити захист від можливих атак, оскільки розробники мають можливість оперативно виправити проблеми і випустити оновлення з виправленнями. Це зменшує ризик успішного злому або недозволених дій на сайтах, що працюють на WordPress.

Друга причина полягає у збереженні репутації. Уразливості, які залишаються непоміченими або не виправляються вчасно, можуть призвести до втрати контролю над сайтом або витоку конфіденційних даних, що серйозно нашкодить репутації компанії чи особистого бренду.

Третя причина – виконання нормативних вимог. Деякі стандарти безпеки (наприклад, PCI DSS для платіжних систем) передбачають виявлення та швидке усунення вразливостей як обов'язковий етап для дотримання вимог.

Зазначимо, що вчасне виявлення вразливостей допомагає зменшити ризик фінансових втрат, оскільки це дозволяє уникнути можливих наслідків, пов'язаних із збоями в роботі сайту, втратою даних чи клієнтів.

Розроблена модель та скрипт допоможе розробникам і тестувальникам прискорити процес виявлення вразливостей у Wordpress, оскільки вони можуть запустити один скрипт і в середньому через 10 хвилин отримати об'ємний і змістовний звіт із виявленими вразливостями. У такий сросіб оптимізується виявлення вразливостей через автоматизований запуск сканерів.

Внесок авторів: Сергій Бучик – концептуалізація, методологія; Андрій Куроєдов – аналіз джерел, підготовка огляду літератури, теоретичних засад дослідження, підготовка лабораторії для дослідження, проведення дослідження.

Список використаних джерел

Hoffman, A. (2024). *Web Application Security.* O'Reilly Media (2nd Ed.).

Liang, Y. (2014). *JavaScript Security.* PACKT Publishing.

Noman, M., Iqbal, M., & Manzoor, A. (2020). A Survey on Detection and Prevention of Web Vulnerabilities. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 11(6). http://dx.doi.org/10.14569/IJACSA.2020.0110665

Rokia, A., & El Habib, N. (2022). Deep Learning for Vulnerability and Attack Detection on Web Applications: A Systematic Literature Review. *Future Internet,* 14(4). https://doi.org/10.3390/fi14040118

Stuttard, D., & Marcus Pinto, M. (2008). *The Web Application Hacker's Handbook.* Wiley Publishing.

References

Hoffman, A. (2024). *Web Application Security.* O'Reilly Media (2nd Ed.).

Liang, Y. (2014). *JavaScript Security.* PACKT Publishing.

Noman, M., Iqbal, M., & Manzoor, A. (2020). A Survey on Detection and Prevention of Web Vulnerabilities. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 11(6). http://dx.doi.org/10.14569/IJACSA.2020.0110665

Rokia, A., & El Habib, N. (2022). Deep Learning for Vulnerability and Attack Detection on Web Applications: A Systematic Literature Review. *Future Internet,* 14(4). https://doi.org/10.3390/fi14040118

Stuttard, D., & Marcus Pinto, M. (2008). *The Web Application Hacker's Handbook.* Wiley Publishing.

Отримано редакцією журналу / Received: 30.10.24
Прорецензовано / Revised: 25.11.24
Схвалено до друку / Accepted: 30.11.24

ISSN 2707-1758

Serhii BUCHYK, DSc (Engin.), Prof.
ORCID ID: 0000-0003-0892-3494
e-mail: buchyk@knu.ua
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

Andrii KUROIEDOV, Student
ORCID ID: 0009-0007-5811-4798
e-mail: askuroyedov@gmail.com
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine

JAVASCRIPT-BASED WEB EXPLOIT ANALYSIS MODEL

Background. ***The task of ensuring the security of web applications and servers remains important and relevant in the face of the ever-increasing number of attacks in cyberspace. The use of various open-source content management systems (e.g. WordPress, Joomla, Open Journal Systems, Drupal), which are quite popular for creating websites due to their ease of installation and use, unfortunately, require constant updating not only to improve the content but also to ensure the security of the system. In this article, the authors focus on the WordPress system, although this approach can be used for other systems as well. The article emphasises the importance of early detection of vulnerabilities to prevent potential cyber threats and their negative consequences. The article proposes a model and a script designed to speed up the detection of vulnerabilities in WordPress applications. Automation of the scanning process with a custom script allows you to quickly detect vulnerabilities, ensuring prompt fixes and updates. This approach not only strengthens security, but also helps preserve the reputation of websites and brands, which is critical in today's digital environment.***

Methods. ***The methods of analysing JavaScript-based web exploits were used, taking into account the general principles of their analysis and taking into account the methodologies for analysing web applications for vulnerabilities.***

Results. ***An improved model of analysing a web application on CMS Wordpress based on a script that provides automated scanning of a web application by running the following utilities is presented: NMAP, Dirb, Nikto, SQLMap, WPScan and PwnXSS. All the results are recorded in a separate file for further study of all the found security issues of the web application.***

Conclusions. ***The developed model and script should help developers and testers speed up the process of identifying vulnerabilities in Wordpress, as they can run one script and get a voluminous and meaningful report with the identified vulnerabilities in a short time. This optimises vulnerability detection by automating the launch of scanners.***

Keywords: ***vulnerability, website, web application, web exploitation, web application analysis, vulnerability scanning, SQL, XSS, CSRF.***

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

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