УДК 004.056.5 https://doi.org/10.35546/kntu2078-4481.2023.3.11

О. О. БОСКІН
старший викладач кафедри програмних засобів і технологій
Херсонський національний технічний університет
ORCID: 0000-0001-7391-0986

Н. В. КОРНІЛОВСЬКА
кандидат технічних наук, доцент,
доцент кафедри інформатики і комп'ютерних наук
Херсонський національний технічний університет
ORCID: 0000-0002-8331-8027

В. М. ПОЛІЩУК
кандидат технічних наук, доцент,
доцент кафедри автоматизації, робототехніки і мехатроніки
Херсонський національний технічний університет
ORCID: 0000-0002-8775-4977

Н. В. САРАФАННІКОВА
кандидат технічних наук, доцент,
доцент кафедри автоматизації, робототехніки і мехатроніки
Херсонський національний технічний університет
ORCID: 0009-0006-6609-016X

БЕЗПЕКА ВЕБ-ДОДАТКІВ ТА ХАКЕРСЬКІ АТАКИ

Питання безпеки веб-додатків та захисту від хакерські атаки є дуже важливою в сучасному світі, оскільки зростає кількість веб-додатків, які містять вразливості та можуть стати об'єктом хакерських атак. Необхідно приділяти достатню увагу заходам безпеки, щоб захистити дані користувачів від небажаного доступу та зловмисного використання. У рамках даної статті було розглянуто основні види хакерських атак на веб-додатки, такі як SQL injection (SQLi), Cross-site scripting (XSS), Cross-site request forgery (CSRF), Server-side request forgery (SSRF) та файлові атаки, а також наведені приклади заходів захисту від кожного типу атак. З метою забезпечення безпеки веб-додатків необхідно дотримуватись принципів безпеки, використовувати надійні паролі, оновлювати програмне забезпечення, вести моніторинг веб-додатків та регулярно проводити аудит безпеки.

В статті проведено огляд, систематизація і узагальнення публікацій з питань принципів безпеки веб-додатків та найбільш загрозливих хакерських атак.

Наведено сучасні методики та підходи до моделювання загроз, які допомагають забезпечити безпеку веб-додатків. Розгдянуті наступні методики моделювання: Threat Modeling, Security Testing, Risk Assessment, Penetration Testing, Security Audits, STRIDE, DREAD, VAST, PASTA, Trike, PTA (Penetration Testing and Assessment). На основі методик моделювання загроз запропоновано оновлені і вдосконалені основні принципи безпеки веб-додатків.

Метою даного дослідження є визначення основних принципів безпеки веб-додатків і виявлення найбільш поширених вразливостей, що допускають хакерські атаки, визначення засобів захисту від різних типів хакерських атак на веб-додатки. Основні результати дослідження. Здійснено уточнення основних принципів безпеки веб-додатків, запропоновані засоби захисту від різних типів хакерських атак на веб-додатки. Наукова новизна. Розроблено дієву методичну систему захисту від найбільш загрозливих хакерських атак.

Ключові слова: *веб-додаток, хакерська атака, SQL injection, Cross-site scripting, Cross-site request forgery, Server-side request forgery, файлова атака.*

O. O. BOSKIN
Senior Lecturer at the Software Tools and Technologies Department
Kherson National Technical University
ORCID: 0000-0001-7391-0986

N. V. KORNILOVSKA
Candidate of Sciences in Technology, Associate Professor,
Associate Professor at the Informatics and Computer Sciences Department
Kherson National Technical University
ORCID: 0000-0002-833-8027

V. M. POLISHCHUK
Candidate of Sciences in Technology, Associate Professor,
Associate Professor at the Automation, Robotics
and Mechatronics Department
Kherson National Technical University
ORCID: 0000-0002-8775-4977

N. V. SARAFANNIKOVA
Candidate of Sciences in Technology, Associate Professor,
Associate Professor at the Automation, Robotics
and Mechatronics Department
Kherson National Technical University
ORCID: 0009-0006-6609-016X

WEB APPLICATION SECURITY AND HACKER ATTACKS

The issue of web application security and protection against hacker attacks is very important in the modern world, as the number of web applications that contain vulnerabilities and may become the target of hacker attacks is growing. It is necessary to pay sufficient attention to security measures to protect user data from unwanted access and malicious use. This article discusses the main types of hacker attacks on web applications, such as SQL injection (SQLi), Cross-site scripting (XSS), Cross-site request forgery (CSRF), Server-side request forgery (SSRF), and file attacks, and provides examples of protection measures against each type of attack. In order to ensure the security of web applications, it is necessary to adhere to security principles, use strong passwords, update software, monitor web applications, and conduct regular security audits.

The article reviews, systematizes and summarizes publications on the principles of web application security and the most threatening hacker attacks.

Modern methodologies over and going are brought near the design of threats that help to provide safety of webaddit ions. Considered next methodologies of design : Threat Modeling, Security Testing, Risk Assessment, Penetration Testi ng, Security Audits, STRIDE, DREAD, VAST, PASTA, Trike, PTA (Penetration Testing and of Assessment). On the basis of methodologies the design of threats is offered the renewed and improved basic principles of safety of web-additions.

The goal of this research is to define the basic principles of web application security and identify the most common vulnerabilities that allow hacker attacks, as well as to determine the means of protection against various types of hacker attacks on web applications. The main results of the research. The basic principles of web application security have been clarified, and means of protection against various types of hacker attacks on web applications have been proposed. Scientific novelty. An effective methodological system of protection against the most threatening hacker attacks has been developed.

Key words: *web application, hacker attack, SQL injection, Cross-site scripting, Cross-site request forgery, Server-side request forgery, file attack.*

Постановка проблеми

У сучасному світі, наповненому технологіями, веб-програми завжди доступні для широкого загалу, на сьогоднішній день важко знайти організацію, підприємство, бізнес, які б не мали власного веб-ресурсу, або не використовували інтернет технології. На відміну від внутрішніх мережевих програм, веб-браузер доступний для всіх користувачів, які мають підключення до інтернету, і для зловмисників теж.

Розробники, на жаль, зазвичай не звертають увагу на принципи захисту веб-додатків. Команди здебільшого настільки зосереджені на функціональних аспектах проекту, таких, як кодування, графічний дизайн і зручність використання, що вони забувають витрачати час на те, щоб переконатися, що він функціонує стабільно та безпечно.

З кожним роком зростає кількість атак на веб-сайти. На жаль, не існує точних статистичних даних про кількість атак на веб-додатки за кожен окремий рік, оскільки ці дані зазвичай не є загальнодоступними. Статистику кібератак надають лише конкретні організації або вендори з метою власного аналізу та забезпечення безпеки своїх продуктів. Однак, деякі дослідження та звіти інформують про загальну тенденцію зростання кількості атак на веб-додатки в цілому.

Так, за даними звіту Verizon Data Breach Investigations Report 2021, атаки на веб-додатки були причиною 39% усіх кіберінцидентів, що були досліджені в 2020 році. Звіт також вказує, що використання атак на веб-додатки зросло з 20% в 2019 році до 27% у 2020 році.

Іншим джерелом є звіт Akamai Technologies про стан безпеки в Інтернеті, опублікований у 2021 році. Звіт стверджує, що атаки на веб-додатки залишаються однією з основних загроз для організацій, зокрема, було зафіксовано зростання кількості SQL-ін'єкцій на 57% в порівнянні з 2019 роком.

Загалом, ці дані підтверджують те, що атаки на веб-додатки залишаються найбільшою і серйозною загрозою для безпеки в Інтернеті та вимагають постійного моніторингу та заходів забезпечення безпеки.

Сайт практично кожної компанії зберігає та обробляє персональні дані користувачів, конфіденційну інформацію, реалізує прийом онлайн-платежів, і власники веб-ресурсів часто навіть не замислюються про те, що безпека веб-проекту постійно перебуває під загрозою. Навіть якщо крадіжка даних або порушення роботи сайту не є метою злому, то у кіберзлочинців існують також інші причини для цього – розсилання спаму або тимчасове використання веб-сервера для зберігання файлів (часто нелегального змісту).

Кіберзлочинці виконують зломи спеціально записаними автоматизованими скриптами. Вони моніторять інтернет у спробі зламати сайти, які мають відомі вразливості у програмному забезпеченні веб-додатків.

Виходячи з цього набуває актуальності питання визначення основних принципів безпеки веб-додатків.

Аналіз останніх досліджень і публікацій

Gary McGraw[1] детально розглядає проблему безпеки програмного забезпечення в цілому, включаючи безпеку веб-додатків, досліджує основні принципи безпеки веб-додатків, в тому числі такі питання, як ідентифікація загроз, захист від атак, захист від перехоплення та підробки даних, керування доступом, захист від зловживань та інше. Автор пропонує конкретні практичні поради та рекомендації щодо захисту веб-додатків від потенційних загроз та атак, а також описує інструменти та методи, які можуть допомогти розробникам створити безпечні веб-додатки. Наводить приклади та ілюстрації, що допомагають зрозуміти складні концепції та принципи безпеки веб-додатків.

Gary McGraw [2] висвітлює принципи безпеки веб-додатків з точки зору проектування, розробки та тестування програмного забезпечення. Пропонує широкий огляд безпекових загроз, які стосуються веб-додатків, включаючи атаки на рівні входу даних, вразливості, пов'язані з безпекою сеансу, а також атаки на сервер. В [3] надає рекомендації та кроки, які повинні бути виконані для підвищення безпеки веб-додатків, включаючи використання захисту від перехоплення, збереження безпеки паролів та забезпечення контролю доступу. Висвітлює принцип захисту від хибного введення даних та використання валідації введених даних. Автор надає рекомендації щодо безпеки веб-сервісів та використання безпечних методів передачі даних, таких як HTTPS та SSL / TLS. Також автор описує методології для розробки безпечних веб-додатків, такі як Secure Development Lifecycle (SDL), та висвітлює важливість забезпечення безпеки під час процесу розробки та тестування програмного забезпечення.

Howard та LeBlanc [4, 5] досліджують принципи безпеки програмного забезпечення, зокрема веб-додатків. Вони описують різні види загроз безпеці веб-додатків та методи їх захисту. Надають поради щодо безпеки при розробці веб-додатків, включаючи використання безпечних методів автентифікації та авторизації, валідацію введених даних, управління сесіями, захист від SQL Injection та Cross-Site Scripting атак, і багато іншого. Наводять приклади коду та практичні поради щодо тестування безпеки веб-додатків.

Джерело [6] є докладним посібником для хакерів та професіоналів з безпеки веб-додатків для виявлення та експлуатації вразливостей. Автори пропонують розглядати веб-додатки як системи, що мають певний рівень безпеки, але також мають певні слабкі місця, які можна використовувати для атак. Описано різні типи атак, такі як SQL injection, Cross-site scripting (XSS), CSRF, а також техніки експлуатації вразливостей. Автори надають знання та навички, необхідні для здійснення успішних атак на веб-додатки. Крім опису технік атак, висвітлено основні принципи безпеки веб-додатків, такі як захист від SQL injection, захист від XSS та CSRF. Автори підкреслюють важливість врахування аспектів безпеки веб-додатків ще на етапі їх розробки та тестування. Крім того, автори надають поради щодо виявлення та виправлення вразливостей, а також процесу здійснення пенетраційного тестування для оцінки рівня безпеки веб-додатків.

Adam Shostack [7] описує, як виконувати моделювання загроз для веб-додатків, щоб забезпечити їх безпеку. Автор наголошує на важливості врахування аспектів безпеки на ранніх етапах розробки веб-додатків, що дозволяє виявляти та виправляти потенційні проблеми безпеки на етапі проектування. Описує різні методики та підходи до моделювання загроз, а також надає поради та рекомендації щодо визначення ризиків та застосування відповідних контрмір для забезпечення безпеки веб-додатків. детально розглядаються основні принципи безпеки веб-додатків, такі як автентифікація, авторизація, шифрування даних та інші.

Chris Easttom [8] розглядає основні загрози, які можуть стати викликом для безпеки веб-додатків, такі як вразливості, що дозволяють виконання коду на віддаленому сервері, SQL-ін'єкції, атаки на автентифікацію та авторизацію, а також вразливості Cross-Site Scripting і Cross-Site Request Forgery. Для кожної з цих загроз автор розглядає причини виникнення, можливі наслідки для безпеки веб-додатків та запропоновані методи захисту від них. Крім того, автор розглядає принципи безпеки веб-додатків, такі як зменшення атакованої поверхні, забезпечення безпеки на всіх рівнях стеку веб-додатків.

Формулювання мети дослідження

Визначення основних принципів безпеки веб-додатків і виявлення найбільш поширених вразливостей, що допускають хакерські атаки, визначення засобів захисту від різних типів хакерських атак на веб-додатки.

Виклад матеріалу дослідження

Моделювання загроз – це процес ідентифікації потенційних загроз безпеці в системі, програмному забезпеченні або мережі з метою визначення вразливостей та вирішення проблем безпеки. Деякі методики та підходи до моделювання загроз, які допомагають забезпечити безпеку веб-додатків [1–3, 7, 9, 10]:

Threat Modeling: Threat modeling є однією з найбільш відомих методик моделювання загроз, в основі якої лежить ідентифікація потенційних загроз для системи, аналіз вразливостей та розробка стратегії зниження ризику. У цій методиці використовуються різні інструменти, наприклад, діаграми, таблиці, документація та інші інструменти для оцінки загроз.

Security Testing: Security testing є процесом виявлення вразливостей та потенційних загроз безпеці програмного забезпечення. Цей процес може включати в себе тестування на проникнення, тестування на злам та інші методи тестування безпеки.

Risk Assessment: Risk assessment включає в себе процес ідентифікації та аналізу загроз безпеці, оцінку ризиків та розробку стратегій зниження ризику. У цьому процесі використовуються різні інструменти, такі як аналіз SWOT (Strengths, Weaknesses, Opportunities, Threats – методика стратегічного планування, яка дозволяє оцінити сильні і слабкі сторони, можливості і загрози), аналіз потенційних причин виникнення проблем та інші методи.

Penetration Testing: Penetration testing є процесом активного виявлення вразливостей та потенційних загроз безпеці за допомогою симуляції атак на систему. Цей процес дозволяє виявити потенційні проблеми безпеки та розробити стратегії зниження ризику.

Security Audits: Security audits є процесом оцінки безпеки системи, мережі або програмного забезпечення на відповідність стандартам безпеки. Цей підхід дозволяє оцінити наявні безпекові проблеми, виявити потенційні вразливості та розробити план заходів з їх вирішення. До основних переваг security audit належать зниження ризику втрати даних, підвищення надійності системи, зменшення витрат на відновлення після атаки та забезпечення дотримання стандартів безпеки.

STRIDE: Цей підхід розроблений компанією Microsoft і включає 6 категорій загроз, які можуть виникнути у веб-додатках: виконання небезпечних операцій (Spoofing), підробка (Tampering), вставка коду (Repudiation), розголошення конфіденційної інформації (Information Disclosure), використання (Denial of Service) та підміна (Elevation of Privilege).

DREAD: Ця методика розглядає п'ять факторів, що впливають на загрози: Поширеність (Damage potential), Вплив на бізнес (Reproducibility), Вплив на відповідальність (Exploitability), Залежність від вразливості (Affected users), Частота виникнення (Discoverability).

VAST: Цей підхід використовується для виявлення і моделювання загроз шляхом проведення віртуальних атак на веб-додаток. Цей метод включає кілька етапів: виявлення векторів атак, сканування портів, сканування служб, визначення вразливостей і експлойтів, а також отримання доступу до системи.

PASTA: Цей підхід включає в себе п'ять етапів: Підготовчий аналіз (Preparation), Аналіз загроз (Threat modeling), Оцінка ризиків (Risk assessment), Розробка заходів забезпечення безпеки (Security design), Відстеження (Tracking).

Trike: Цей підхід включає в себе три етапи: розробка архітектури, оцінка загроз та вибір заходів забезпечення безпеки.

PTA (Penetration Testing and Assessment): Це підхід, який включає проведення тестування на проникнення з метою виявлення вразливостей в веб-додатках. Цей підхід використовує різні техніки тестування, включаючи сканування портів, сканування вразливостей, тестування на стійкість до атак, аналіз захисту від SQL-ін’єкції, XSS, CSRF та інших типів атак.

На основі методик моделювання загроз можна запропонувати наступні основні принципи безпеки веб-додатків:

1. Автентифікація: вимагати від користувачів введення ідентифікаторів та паролів для доступу до захищеної інформації.

2. Авторизація: забезпечення правильного рівня доступу до захищених ресурсів, щоб забезпечити конфіденційність та цілісність інформації.

3. Захист від SQL-ін’єкцій: запобігання виконанню небажаних запитів до бази даних, що можуть привести до руйнування даних.

4. Захист від міжсайтових скриптів: запобігання виконанню небажаних скриптів, які можуть зламати безпеку веб-сторінки.

5. Захист від переповнення буфера: запобігання збоїв, які можуть виникнути, коли ввід забезпечується даними більше, ніж максимально допустима кількість.

6. Захист від вразливостей, пов’язаних з файловою системою: запобігання доступу до конфіденційних даних, які можуть бути збережені в файловій системі веб-сервера.

7. Захист від атак з використанням перехоплення сесії: запобігання доступу до конфіденційної інформації шляхом перехоплення сесії користувача.

8. Захист від атак з використанням підроблення ідентифікатора: запобігання доступу до захищеної інформації шляхом підроблення ідентифікаторів користувача.

9. Захист від атак з використанням «вимагання пам'яті»: запобігання доступу до конфіденційних даних шляхом примушення збоїв у пам'яті веб-додатку.

10. Захист від атак з використанням перехоплення кукі: використання HTTPS:через захист від CSRF (Cross-Site Request Forgery) та захист від XSS (Cross-Site Scripting).

11. Використання криптографії для захисту даних, переданих між користувачем і сервером. Це включає в себе використання протоколів шифрування, таких як SSL/TLS, та застосування стійких алгоритмів шифрування.

12. Регулярне оновлення програмного забезпечення, в тому числі веб-серверів, баз даних, фреймворків та інших компонентів веб-додатків, щоб уникнути використання вразливостей, які вже були виправлені.

13. Відключення або обмеження доступу до функцій, які не використовуються в додатку, таких як виконання системних команд, виконання файлів на сервері та інші.

14. Використання відповідних правил контролю доступу для обмеження доступу користувачів до конфіденційної інформації та функцій додатку.

15. Використання систем моніторингу та реєстрації подій, які дозволяють виявляти та реагувати на некоректну роботу додатку, а також на підозрілу активність.

16. Використання систем резервного копіювання, щоб забезпечити можливість відновлення додатку у разі втрати даних або інших негативних наслідків.

17. Розуміння та дотримання відповідних правил та законодавства, пов'язаного з захистом даних користувачів, таких як Закон про захист персональних даних.

Хакерські атаки на веб-додатки можна класифікувати за різними критеріями, наприклад, за типом атаки, використовуваним вразливостям, метою атаки тощо. Наведемо найбільш поширені типи атак на веб-додатки:

1. SQL injection – це атака, коли хакер вставляє SQL-код в веб-форми, щоб отримати доступ до бази даних, або змінити, видалити або додати дані. SQL-ін'єкції можуть бути дуже шкідливими для веб-додатків, оскільки вони можуть дозволити хакеру виконувати будь-які дії з базою даних, навіть виконання команд на сервері (рис. 1).

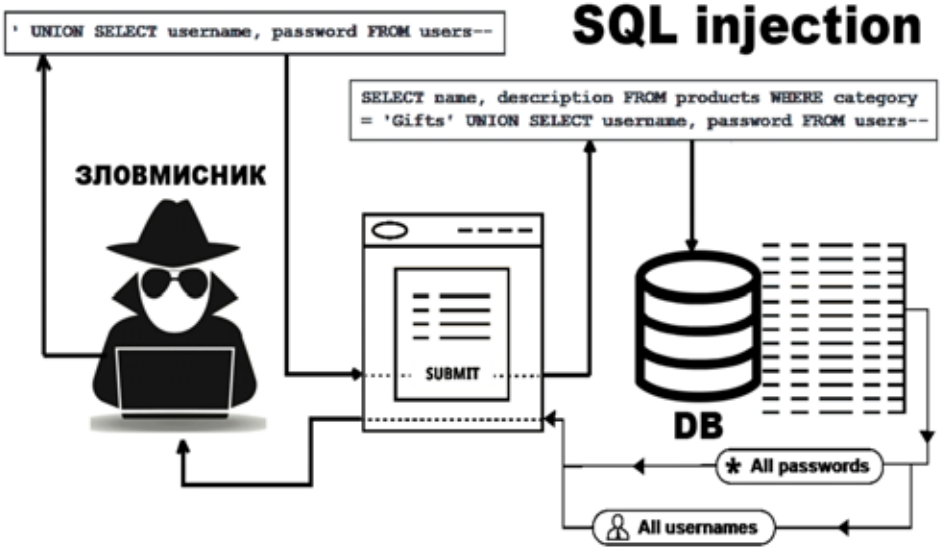

Рис. 1. Атака SQL injection

Основним принципом атаки SQLi є використання недостатньо фільтрованого вводу користувача, що дозволяє зловмиснику вставляти в SQL-запит шкідливий код. Наприклад, якщо веб-додаток використовує запит типу "SELECT * FROM users WHERE username = 'input'", зловмисник може вводити значення, які містять шкідливий SQL-код, наприклад, " ' OR 1=1; --". В цьому випадку весь запит стає "SELECT * FROM users WHERE username = " OR 1=1; --'", що дозволяє зловмиснику отримати доступ до всіх записів в таблиці users.

2. Cross-site scripting – це атака, при якій хакер вставляє скрипти в веб-сторінки, щоб отримати доступ до даних користувачів, які переглядають сторінки. Ця атака може дозволити хакеру виконувати будь-які дії від імені користувача, включаючи крадіжку паролів та іншої конфіденційної інформації.

Cross-site scripting є однією з найбільш поширених хакерських атак на веб-додатки. XSS полягає в тому, що зловмисники вбудовують скрипти в веб-сторінки, які відображаються для інших користувачів (рис. 2).

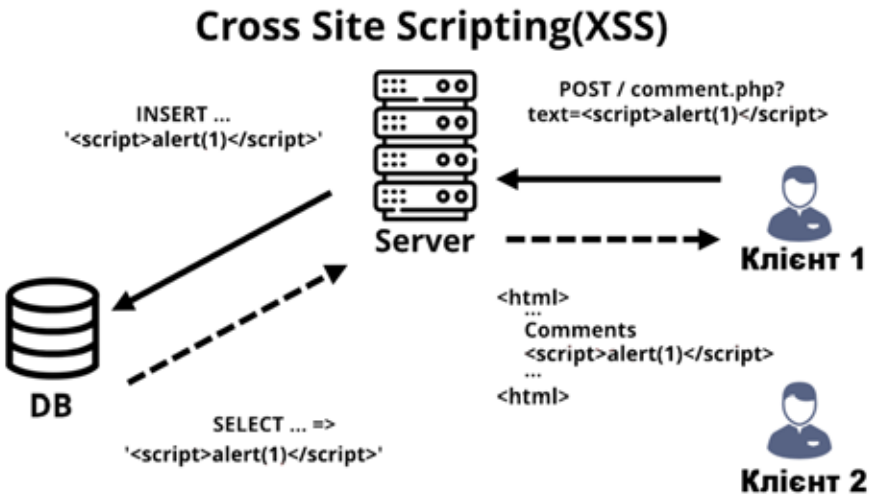

Рис. 2. Атака Cross-site scripting

Наприклад, зловмисники можуть вбудувати скрипт, який виконує певну дію, наприклад, крадіжку кукі або паролів користувачів, коли інший користувач відкриває відповідну веб-сторінку. За допомогою XSS-атак зловмисники можуть також перенаправляти користувачів на інші веб-сторінки, що містять шкідливий вміст.

Основні типи XSS-атак:

– збільшення вразливості введення даних: зловмисники вставляють скрипти в форми, поля для введення даних або параметри URL-адреси веб-сторінки;

– XSS за допомогою відображення даних: зловмисники вставляють скрипти в коментарі, повідомлення чи інші відображувані дані на веб-сторінці;

– розширення URL-адреси: зловмисники додають до URL-адреси скрипти, які виконуються, коли користувач переходить за відповідним посиланням.

3. Cross-site request forgery – це атака, при якій хакер використовує веб-сторінки, щоб змусити користувачів виконувати дії, які вони не хочуть виконувати. Наприклад, хакер може відправити електронний лист з посиланням, що виконує певну дію, таку як відправка грошей з банківського рахунку, якщо користувач натисне на посилання (рис. 3).

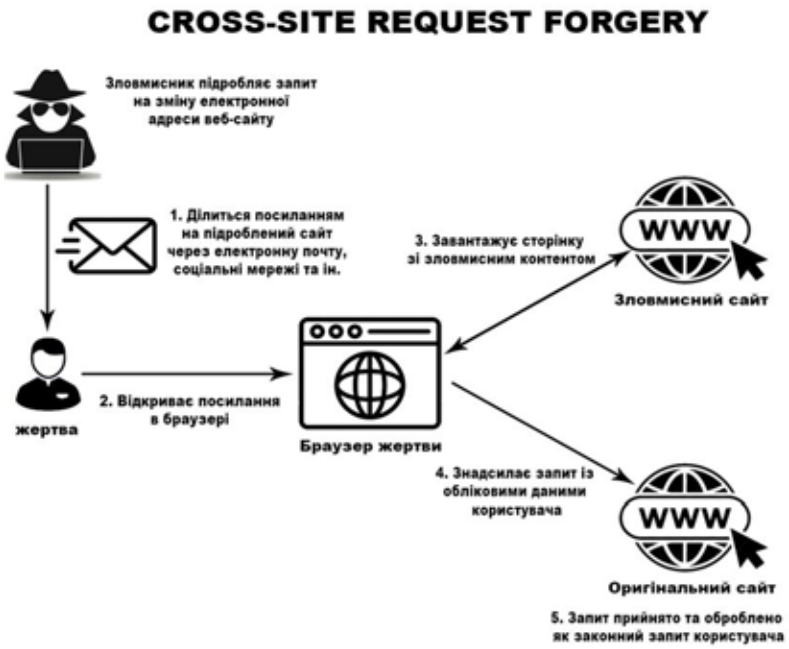

Рис. 3. Атака Cross-site request forgery

Основна ідея атаки полягає в тому, що зловмисник змушує авторизованого користувача перейти на підроблену сторінку, де за допомогою скрипту відправляється запит на сервер з підробленими параметрами. Якщо на сервері немає достатньої перевірки запитів, то він може виконати небажану дію без попередньої авторизації користувача, наприклад, відправити команду на видалення або зміну даних.

Приклад: зловмисник створює підроблену сторінку, що містить форму з внутрішнім командним кодом для видалення даних. Він надсилає посилання на цю сторінку авторизованому користувачеві в листі електронної пошти або через повідомлення в соціальній мережі. Якщо користувач перейде за цим посиланням, то його запит буде виконаний на сервері, і відбудеться видалення даних без попередньої авторизації користувача.

4. Server-side request forgery – це атака, коли хакер використовує вразливість веб-додатку, щоб змусити сервер виконувати запити до інших ресурсів в мережі, таких як бази даних, файлові системи або інші веб-сервери. Наприклад, нападник може використовувати сервер, щоб скопіювати або видалити дані зі стороннього сервера або виконати шкідливий код на іншому сервері (рис. 4).

Один з основних прикладів атаки SSRF полягає в тому, що зловмисник використовує вразливість на веб-сайті для відправки запиту до внутрішньої мережі організації. Наприклад, зловмисник може використовувати адресу IP внутрішньої мережі, яку він дізнався під час перегляду заголовків відповіді попереднього запиту до сервера. Після цього зловмисник може надіслати запит на внутрішній сервер з використанням цієї адреси IP, що дозволить йому виконати дії, такі як отримання конфіденційних даних або знищення внутрішніх ресурсів.

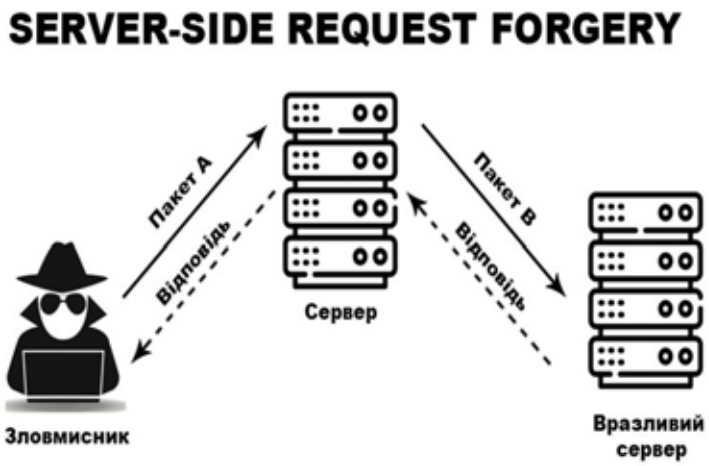

Рис. 4. Атака Server-side request forgery

Одним з методів захисту від атаки SSRF є перевірка вхідних даних та параметрів запитів на вміст небезпечних URL-адрес. Також може бути використана блокування доступу до внутрішньої мережі з зовнішнього сервера.

5. Файлові атаки – це атаки, коли хакер використовує вразливість веб-додатку, щоб завантажити зловмисний файл на сервер або змінити існуючі файли. Ці атаки можуть бути особливо небезпечними, якщо зловмисний файл містить шкідливий код,

Одним з прикладів файлової атаки, яка використовує вразливості в програмному забезпеченні, є атака на файлову систему сервера. Хакер може використовувати вразливості в програмному забезпеченні, такі як недостатні перевірки на введення даних або вразливості в сторонніх бібліотеках, щоб виконати код на сервері та отримати доступ до файлів, які повинні бути захищені.

Наприклад, якщо веб-додаток має вразливість, яка дозволяє хакеру виконати довільний код на сервері, то хакер може виконати команду, яка зчитує файли з сервера та надсилає їх на зловмисний сервер. Хакер може також змінювати файли на сервері, що може призвести до руйнування даних або порушення конфіденційності даних.

Пропонуємо засоби захисту від різних типів хакерських атак на веб-додатки.

1. Для захисту від SQL injection можна застосовувати наступні методи:

– використовувати підготовлені запити (Prepared Statements): це дозволяє попередньо компілювати запити і забезпечує захист від впливу введеного користувачем SQL-коду;

– використовувати змінні в запитах: замість жорстко заданого SQL-коду динамічно формувати запит з використанням змінних;

– використовувати фільтрацію введення користувача: перевіряти введення користувача на наявність SQL-коду та використовувати фільтри для очищення даних перед виконанням запиту;

– обмежувати права доступу користувачів до бази даних: це дозволить зменшити можливість впливу SQL-ін’єкції на базу даних;

– оновлювати ПЗ та застосовувати патчі: це дозволяє закрити вразливості, які можуть бути використані зловмисниками для здійснення атак.

– використовувати відповідні бібліотеки та фреймворки: вони містять вбудовані механізми захисту від SQL-ін'єкцій, які допомагають запобігти атакам.

2. Пропонуємо наступні методи захисту від Cross-site scripting:

– валідація даних на вході: перевірка введеного користувачем тексту на наявність потенційно небезпечних символів і забезпечення його безпечності до передачі на сервер;

– екранування символів: заміна потенційно небезпечних символів на безпечні еквіваленти, які не будуть інтерпретовані браузером як код;

– використання Content Security Policy (CSP): це механізм безпеки, який дозволяє обмежити джерела контенту, які можуть бути завантажені на сторінку, заборонити виконання скриптів з певних джерел;

– використання HTTP-only кукі: дозволяє заборонити доступ до кукі з JavaScript, що зменшує ризик XSS-атак, пов'язаних зі зловживанням кукі;

– обмеження введення користувача: налаштування обмежень для довжини введеного користувачем тексту та його формату, а також використання інших методів перевірки на коректність введення даних.

3. Для захисту від Cross-site request forgery рекомендуємо виконувати такі заходи:

– використовувати механізм захисту від CSRF, якщо він доступний у веб-фреймворку, який використовується для розробки веб-додатку;

– використовувати токени випадкової генерації, які прив'язані до конкретного користувача або сесії, і включати їх у форми та запити, які містять дії, які можуть змінити стан додатку;

– використовувати HTTP-заголовок Referer для перевірки того, що запити приходять зі сторінок, які належать до певного веб-сайту. Проте, цей метод може бути ненадійним, оскільки заголовок Referer може бути змінений або відсутнім у запиті;

– не використовувати метод GET для здійснення дій, які можуть змінити стан додатку, такі як видалення або редагування даних. Замість цього, використовувати метод POST або PUT, який дозволяє включати токени в запити;

– не зберігати конфіденційну інформацію в кукі або в GET-запитах, оскільки ці дані можуть бути підмінені при атаках CSRF. Використовувати метод POST для відправки конфіденційної інформації на сервер.

4. Пропонуємо основні заходи для захисту від SSRF:

– обмеження доступу до внутрішніх ресурсів: налаштування файрвола та мережевих налаштувань для забезпечення обмеження доступу до внутрішніх ресурсів, таких як бази даних, сервіси та інші сервіси;

– перевірка вхідних даних: перевірка вхідних даних, що надходять від користувача на наявність даних, які вказують на внутрішні ресурси, та відхилення запитів, що містять небезпечні дані;

– використання білого списку: використання білого списку для вказівки дозволених адрес віддалених серверів, з яких можуть бути здійснені запити;

– використання технологій безпеки: використання технологій безпеки, таких як Content Security Policy (CSP) та Subresource Integrity (SRI), для забезпечення контролю та перевірки вмісту, що завантажується з зовнішніх джерел;

– налаштування прав доступу: налаштування прав доступу на серверах та інші налаштування, що дозволяють обмежити можливість здійснення запитів на внутрішні ресурси;

– регулярні оновлення програмного забезпечення: регулярні оновлення програмного забезпечення, що використовується, включаючи серверну та клієнтську частини, для усунення виявлених вразливостей.

5. Для захисту від файлових атак рекомендуємо дотримуватись наступних засад:

– заборона виконання виконуваних файлів на сервері, які знаходяться поза директорією веб-сайту;

– відключення виконання PHP-файлів у директоріях, які не призначені для виконання скриптів;

– заборона віддаленого виконання коду (remote code execution) через веб-додатки;

– перевірка типів файлів, які користувач може завантажувати на веб-сайт;

– перевірка дозволів на завантаження файлів на сервер, заборона завантаження виконуваних файлів та обмеження розміру завантажуваного файлу;

– використання надійних методів шифрування для передачі файлів на сервер, таких як SSL;

– регулярне оновлення програмного забезпечення та встановлення патчів для усунення виявлених вразливостей.

Висновки

У зв'язку зі зростаючим попитом на веб-додатки та залежністю функціонування багатьох систем від них, безпека веб-додатків стала надзвичайно важливою та актуальною. Хакерські атаки на веб-додатки можуть відкрити доступ до конфіденційної інформації, зламати роботу додатка або навіть спричинити виток даних. Для запобігання цим атакам необхідно застосовувати різні техніки та практики безпеки веб-додатків.

До основних хакерських атак на веб-додатки відносяться SQL injection, Cross-site scripting, Cross-site request forgery, Server-side request forgery, а також різні види файлових атак. Кожна з цих атак має свої характеристики та методи захисту.

Основні принципи безпеки веб-додатків включають захист від атак на вході та виході, валідацію вхідних даних, обмеження доступу до ресурсів та використання шифрування. Для захисту від конкретних хакерських атак, таких як SQLi, XSS, CSRF та SSRF, необхідно використовувати спеціальні техніки та інструменти.

Для забезпечення безпеки веб-додатків також необхідно постійно оновлювати програмне забезпечення, використовувати складні паролі та двофакторну автентифікацію, робити резервні копії даних та запроваджувати регулярні аудити безпеки.

Список використаної літератури

1. McGraw, G. Software security: Building security in. Addison-Wesley Professional, 1st edition, 2006. 396 p.
2. McGraw, G. Software security: Building security in. Addison-Wesley Professional, 2nd edition 2018. 528 p.
3. John Viega, Gary R. McGraw. Building Secure Software. Addison-Wesley Professional. 2002. 528 p.
4. Michael Howard, David LeBlanc. Writing secure code (Vol. 19). Microsoft Press, 2002. 608 p.
5. Michael Howard, David LeBlanc & John Viega. 19 Deadly Sins of Software Development. New York, NY: McGraw-Hill, 2005. 348 p.
6. Stuttard, D., & Pinto, M. The web application hacker's handbook: Finding and exploiting security flaws. John Wiley & Sons, 2016. 912 p.
7. Shostack, A. Threat modeling: designing for security. John Wiley & Sons, 2014. 624 p.
8. Easttom, C. Computer security fundamentals (3rd ed.). Pearson, 2016. 448 p.
9. Mark Dowd, John McDonald, & Justin Schuh. The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities. Addison-Wesley Professional, 2006. 1200 p.
10. Eric Byres. Threat Modeling: Uncover Security Design Flaws Using the STRIDE Approach. Addison-Wesley Professional, 2014. 232 p.
11. Douglas Craig. Security Web Applications. O'Reilly Media, 2007. 296 p.
12. Bass, T., Clements, P., & Kazman, R. Software architecture in practice (3rd ed.). Addison-Wesley Professional, 2015. 624 p.
13. Felt, A. P., Finifter, M., Chin, E., Hanna, S., & Wagner, D. A survey of web security research. IEEE Transactions on Information Forensics and Security, 6(3), 2011. p. 1-17. DOI: 10.1109/TIFS.2011.2118713
14. Ristic, I. Bulletproof SSL and TLS: Understanding and deploying SSL/TLS and PKI to secure servers and web applications. Feisty Duck, 2013. 400 p.
15. Stamper, R. Information security: principles and practice. John Wiley & Sons, 2005. 488 p.
16. Mitchell, R. Web penetration testing with kali linux: Discover the power of Kali Linux, one of the most popular tools for penetration testing, using real-world scenarios. Packt Publishing Ltd., 2019. 488 p.
17. Westrum, E. F. Secure software design. Auerbach Publications, 2016. 318 p.
18. Florêncio, D., Herley, C., & van Oorschot, P. C. Passwords and the evolution of imperfect authentication. Communications of the ACM, 57(9), 2014. p. 78-87. DOI: 10.1145/2643132.2643136
19. Matt Bishop. Computer security: art and science. Addison-Wesley Professional, 2003. 1132 p.

References

1. McGraw, G. (2006). *Software security: Building security in*. Addison-Wesley Professional, 1st edition.
2. McGraw, G. (2018). *Software security: Building security in*. Addison-Wesley Professional, 2nd edition.
3. Viega, J., & McGraw, G. (2002). *Building Secure Software*. Addison-Wesley Professional.
4. Howard, M., & LeBlanc, D. (2002). *Writing secure code* (Vol. 19). Microsoft Press.
5. Howard, M., LeBlanc, D., & Viega, J. (2005). *19 Deadly Sins of Software Development*. New York, NY: McGraw-Hill.
6. Stuttard, D., & Pinto, M. (2016). *The web application hacker's handbook: Finding and exploiting security flaws*. John Wiley & Sons.
7. Shostack, A. (2014). *Threat modeling: designing for security*. John Wiley & Sons.
8. Easttom, C. (2016). *Computer security fundamentals (3rd ed.)*. Pearson.
9. Dowd, M., McDonald, J., & Schuh., J. (2006). *The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities*. Addison-Wesley Professional.
10. Byres., E. (2014). *Threat Modeling: Uncover Security Design Flaws Using the STRIDE Approach*. Addison-Wesley Professional.
11. Craig., D. (2007). *Security Web Applications*. O'Reilly Media.

12. Bass, T., Clements, P., & Kazman, R. (2015). *Software architecture in practice (3rd ed.).* Addison-Wesley Professional.

13. Felt, A. P., Finifter, M., Chin, E., Hanna, S., & Wagner, D. (2011). A survey of web security research. *IEEE Transactions on Information Forensics and Security*, *6*(3), 1-17. DOI: 10.1109/TIFS.2011.2118713

14. Ristic, I. (2013). *Bulletproof SSL and TLS: Understanding and deploying SSL/TLS and PKI to secure servers and web applications*. Feisty Duck.

15. Stamper, R. (2005). *Information security: principles and practice*. John Wiley & Sons.

16. Mitchell, R. (2019). *Web penetration testing with kali linux: Discover the power of Kali Linux, one of the most popular tools for penetration testing, using real-world scenarios*. Packt Publishing Ltd.

17. Westrum, E. F. (2016). *Secure software design*. Auerbach Publications.

18. Florêncio, D., Herley, C., & van Oorschot, P. C. (2014). Passwords and the evolution of imperfect authentication. *Communications of the ACM, 57*(9), 78-87. DOI: 10.1145/2643132.2643136

19. Matt Bishop. (2003). *Computer security: art and science*. Addison-Wesley Professional.