НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. Ігоря СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Звіт з виконання комп'ютерного практикуму №3

ДОСЛІДЖЕННЯ БЕЗПЕЧНОЇ РЕАЛІЗАЦІЇ ТА ЕКСПЛУАТАЦІЇ ДЕЦЕНТРАЛІЗОВАНИХ ДОДАТКІВ

Виконали студенти
групи ФІ-32мн
Пелешенко Любов,
Панасюк Єгор,
Маринін Іван Павло

Перевірила:
Селюх П.В.

Київ — 2024

1

Мета роботи: отримання навичок роботи із децентралізованими додатками та оцінка безпеки інформації при їх функціонуванні

Постановка задачі: дослідження вимог OWASP (безпека web-додатків) та складання аналогічних вимог для обраної системи децентралізованих додатків.

1.1 Вимоги OWASP

OWASP (Open Web Application Security Project) – відкритий проєкт забезпечення безпеки web-додатків. Спільнота OWASP включає в себе корпорації, освітні організації і приватні особи з усього світу. Спільнота працює над створенням статей, навчальних посібників, документації, інструментів і технологій, які перебувають у вільному доступі. Фонд OWASP це благодійна організація, яка надає підтримку і здійснює управління проектами та інфраструктурою OWASP.

OWASP створив список з 10-ти найбільш небезпечних векторів атак на Web-додатки в якому зосереджені найнебезпечніші вразливості, які можуть коштувати деяким людям великих грошей, або підриву ділової репутації, аж до втрати бізнесу. OWASP TOP-10 не є офіційним стандартом, це лише інформаційний документ, який широко використовується багатьма організаціями, програмами виплати винагород за виявлені вразливості і експертами в області кібербезпеки для класифікації рівня небезпеки вразливостей. Верхній рядок TOP-10 займають вразливості, що дозволяють впровадження коду.

За останні роки рейтинг оновлювався декілька разів – в 2013 і 2017 роках. Проте в новій версії відбулися перестановки, а також додалися три нові типи вразливостей: XXE (External Entity Expansion, вразливість сайту

або додатка до впровадження коду XML), Insecure Deserialization і Insufficient Logging and Monitoring. Розглянемо цю останню версію вимог:

1) Injection (Ін'єкція)

2) Broken Authentication (Порушена автентифікація)

3) Sensitive Data Exposure (Розкриття конфіденційних даних)

4) XML External Entities (XXE) (Зовнішні сутності XML)

5) Broken Access Control (Порушений контроль доступу)

6) Security Misconfiguration (Неправильна конфігурація безпеки)

7) Cross-Site Scripting (XSS) (Міжсайтовий сценарій)

8) Insecure Deserialization (Небезпечна десеріалізація)

9) Using Components with Known Vulnerabilities (Використання компонентів з відомими вразливими місцями)

10) Insufficient Logging and Monitoring (Недостатня реєстрація та моніторинг)

Незважаючи на те, що цей список призначений для розробників веб-додатків, багато з цих вразливостей однаково застосовні до технології блокчейн.

Injection

Ін'єкційні атаки належать до широкого класу векторів атак. Ін'єкційні атаки відбуваються, коли зловмисники знаходять слабкі місця в перевірці вхідних даних і вводять ненадійні вхідні дані, які потім обробляються веб-програмою та змінюють заплановану роботу серверної обробки.

• Атаки SQL Injection відбуваються, коли введені користувачем дані не очищаються належним чином, і вони мають найбільший вплив на Інтернет. Коли зловмисники знаходять SQL-ін'єкцію, у більшості випадків вони можуть отримати дані з бази даних і в деяких випадках призводять до віддаленого виконання коду в цільовій системі. Атаки SQL-ін'єкцій мають багато форм, наприклад атаки на основі помилок, атаки на основі об'єднань, атаки наосліп і атаки поза діапазоном. Атаки на основі помилок відбуваються, коли ін'єкції створюють помилки SQL, які можуть розкрити дані в базі даних. Атаки на основі об'єднання — це коли ін'єкції

продовжують попередній оператор SQL для отримання даних із бази даних за допомогою пропозиції UNION, яка приєднує вторинний оператор до існуючого оператора. Атаки наосліп відбуваються, коли ін'єкції не повертають результатів або помилок, тому зловмисник повинен бути креативним, щоб визначити результат. Позасмугові атаки — це коли база даних не повертає вихідних даних, але дозволяє зловмиснику передавати вихідні дані до джерела, де вони можуть читати, наприклад, свого власного HTTP-сервера. Для пом'якшення таких атак розробники повинні належним чином перевіряти введені користувачем дані або використовувати підготовлені оператори під час спілкування з базою даних із веб-програми.

• Ін'єкції коду відбуваються, коли зловмисники використовують недоліки перевірки введення на веб-сайтах і програмному забезпеченні, що дозволяє зловмисникам виконувати довільні команди на цільовому хості. Код впроваджується на мові цільової програми та виконується інтерпретатором на стороні сервера для цієї мови. Існує кілька мов серверної частини, і зловмисники мають виконати перерахування, щоб визначити, яка мова використовується. Деякі з поширених мов - PHP, Java, Python і Ruby. Пом'якшення атак із впровадженням коду включає перевірку введених користувачем даних, видалення системних функцій із програми, обробку всіх даних як ненадійних і перевірку коду програми перед використанням.

• Протокол LDAP (Lightweight Directory Application Protocol) — це протокол, який використовується для розповсюдження списків інформації, організованих у дерева інформації каталогів, які зберігаються в базі даних LDAP.

Ін'єкції LDAP відбуваються, коли зловмисники маніпулюють можливостями пошуку протоколу, що подібно до ін'єкцій SQL. На малюнку 1 показано простий обхід автентифікації у формі входу з використанням впровадження LDAP. Засоби пом'якшення ін'єкцій LDAP включають фільтрацію спеціальних символів на прикладному рівні, щоб користувачі не могли вводити запити у веб-програму.

• Атаки ін'єкції XPATH відбуваються, коли веб-сайти створюють запит XPath для даних XML з інформації, наданої користувачем. Атака здійснюється, коли зловмисний користувач вводить на сайт за допомогою рядків запиту, що може призвести до несанкціонованого доступу або розкрити конфіденційну інформацію. Існує 2 типи атак у ін'єкціях XPATH, які є логічними атаками, коли зловмисники можуть вводити запити, які призводять до істинних або хибних відповідей сервера або сканування XML, що дозволяє зловмисникам витягувати конфіденційну інформацію, вводячи певні запити в XML-документ, який обробляється веб-сайтом. . Засоби пом'якшення цих типів атак подібні до SQL-ін'єкцій і вимагають очищення введених даних користувача або використання попередньо скомпільованих операторів, які передають введені користувачем дані як параметри замість виразів.

• Заголовки хосту використовуються, коли багато веб-програм розміщено на одній IP-адресі та використовуються для визначення того, яка програма має обробляти запит HTTP. Атаки ін'єкції заголовка хосту відбуваються, коли зловмисники вводять шкідливий хост у заголовок хосту HTTP, що дозволяє зловмисникам контролювати ін'єкції на першому віртуальному хості, що може призвести до виконання довільного коду в контексті цільового веб-сайту. Поширена атака, до якої може призвести введення заголовка хосту, — отруєння веб-кешу, коли зловмисники вводять вміст, який зберігається у веб-кеші та відображається будь-ким, хто запитує ресурс. Ще одна поширена атака – це отруєння скидання пароля, яке може виникнути через впровадження заголовків хосту, коли веб-додаток використовує заголовки хосту під час створення посилань для скидання пароля. Зловмисники можуть вставити заголовки хосту, щоб отруїти посилання для скидання, яке надсилається жертвам, дозволяючи зловмиснику отримати доступ до маркер скидання пароля та автентифікуватися як жертва. Для пом'якшення цього типу атак рекомендується додати хости в білий список, які можна вказати в заголовках хостів.

Погана обробка вхідних даних, безумовно, є потенційною проблемою в технології блокчейн. До запуску основної мережі EOS дослідники Qihoo 360 повідомили про вразливість у функції розбору смарт-контрактів EOS, яка дозволяла записувати за межі буфера. Дослідники розробили доказ концепції, де вразливість дозволила їм вийти з пісочниці EOS і запустити зворотну оболонку на зараженій машині. Якби ця вразливість була використана в працюючій мережі, це могло б дозволити зловмиснику скомпрометувати кожен вузол у мережі під час запуску зловмисного смарт-контракту, коли він був включений у дійсний блок.

Broken Authentication

Порушена автентифікація – це термін, який використовується для кількох вразливостей, які дозволяють зловмисникам використовувати та видавати себе за користувачів веб-додатків. Існують різні методи, за допомогою яких зловмисники можуть отримати облікові дані користувача або захопити сеанси користувача, щоб мати можливість видати себе за цих користувачів, наприклад, слабкі облікові дані або облікові дані, які можна вгадати, неправильно збережені облікові дані, як-от нехешовані паролі, які можна отримати за допомогою інших типів атак такі як SQL-ін'єкції, ідентифікатори сеансів, які розкриваються 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. Засоби пом'якшення такого роду атак включають хешування паролів, забезпечення того, щоб ідентифікатори сеансів не розкривалися в URL-адресах, встановлення тайм-аутів для сеансів, щоб сеанси відтворювалися через певний проміжок часу, забезпечення того, щоб паролі не надсилалися через незашифровані з'єднання, використання захисту паролів, наприклад мінімальний дозволена довжина та складність паролів, приховування імен користувачів і повідомлень про помилки пароля під час невдалого входу, щоб користувачів і паролі не можна було підмінити грубим методом, і забезпечення захисту від грубої сили, наприклад використання функцій блокування після 5 спроб. Ці проблеми можуть

дозволити зловмиснику маскуватися під законного користувача на тимчасовій або постійній основі.

Правильна реалізація функцій автентифікації є життєво важливою для належної роботи системи блокчейн, а широке використання криптографії з відкритим ключем означає наявність великої поверхні атаки.

Криптовалюта LISK є хорошим прикладом того, як помилки в дизайні дозволили атаку на автентифікацію в блокчейні. У LISK адреса користувача в блокчейні досягається хешуванням його відкритого ключа та скороченням результату до 64 бітів [4]. Цей метод гешування та скорочення генерації адрес є поширеним; однак коротка довжина адрес LISK і той факт, що адреси не прив'язані відразу до відкритих ключів, роблять LISK вразливим до атак.

Адреси в LISK пов'язані з відкритими ключами лише тоді, коли користувач ініціює транзакцію, надсилаючи значення з облікового запису або голосуючи за делегата. У результаті багато облікових записів, які лише отримували цінність, виявилися вразливими до атак. Зловмиснику потрібно виконати лише 264 операції генерації ключів, щоб знайти пару закритих/відкритих ключів, яка буде зіставлятися з заданою адресою. Націлювання на будь-яку з M адрес зменшує складність пошуку будь-якого збігу в M разів. Ця атака безперечно можлива, і деякі вразливі облікові записи містять мільйони доларів.

Sensitive Data Exposure

Розкриття конфіденційних даних — це вразливість, яка пояснюється сама собою. Якщо програма містить цінні дані, які повинні зберігатися в секреті, ці дані потрібно належним чином захистити.

Розкриття конфіденційних даних стосується веб-додатків, які не захищають таку інформацію, як паролі, фінансова інформація чи дані про здоров'я, що може призвести до того, що кіберзлочинці зловживатимуть цією інформацією для отримання несанкціонованого доступу до облікових записів користувачів, вчинення шахрайських дій, таких як здійснення онлайн-покупок із викраденим платежем. інформацію або шантажувати конфіденційними даними. Розкриття конфіденційних даних може призвести

до фінансових втрат, завдати шкоди репутації корпорацій, інформація чи активи яких були розкриті, і спонукати компанії оплачувати витрати на розслідування витоку даних. Захист від таких атак залежить від законодавчої бази країни та галузі, оскільки їх ігнорування може призвести до руйнівних фінансових результатів

Технологія блокчейн значною мірою схильна до цієї вразливості через відсутність розуміння цієї технології. Блокчейн є незмінним, тобто будь-які дані, що зберігаються в ньому, не можна видалити (без контролю кожного вузла в мережі). Більшість блокчейнів також є публічними, що дозволяє будь-кому завантажувати та зберігати повну копію даних у книзі.

У короткостроковій перспективі ця комбінація робить блокчейни вразливими для зусиль видобутку даних. Багато організацій спеціалізуються на видобутку загальнодоступних даних, які можна агрегувати в корисну інформацію. Ці дані можуть бути використані в правоохоронних органах, для корпоративного шпигунства та в інших цілях.

У довгостроковій перспективі будь-яка криптографія зламана. Квантові обчислення вже на горизонті, і, хоча вони не зламатимуть технологію блокчейну, вони дозволять розшифровувати будь-які дані, що зберігаються в блокчейні, зашифровані за допомогою класичних асиметричних алгоритмів шифрування. Якщо дані потрібно зберігати конфіденційними назавжди (наприклад, персональні дані, захищені законами про конфіденційність, як-от Загальний регламент захисту даних (GDPR) або Закон про перенесення та доступність медичного страхування (HIPAA), їх ніколи не слід зберігати в публічному блокчейні, навіть у зашифрований формат.

XML External Entities

Зовнішні сутності XML (XXE) — це вразливості, засновані на посиланнях на зовнішні сутності в документах XML. Ризик полягає в тому, що конфіденційні внутрішні файли, що зберігаються на веб-сервері, можуть бути доступні за допомогою цих посилань.

Атаки ін'єкції зовнішніх об'єктів XML, також відомі як ін'єкції XXE,

відбуваються, коли зловмисники зловживають аналізаторами розширюваної мови розмітки (XML) на веб-серверах, надсилаючи на веб-сервери спеціально створені шкідливі XML-документи, які обробляються та можуть призвести до відмови в обслуговуванні, віддаленого виконання коду або підробка запиту на сервері.

Ін'єкційні атаки XXE мають два типи атак: внутрішньосмугові атаки, коли зловмисники створюють шкідливий XML-файл, надсилають його для обробки та отримують результати в тій самій смузі, іншими словами, зловмисник отримує миттєву відповідь і виходить з -смугові атаки, які відбуваються, коли зловмисники створюють шкідливі файли XML, надсилають їх на обробку, але не отримують негайної відповіді від веб-сервера, що також називають сліпими ін'єкціями XXE.

Оскільки блокчейн не базується на XML, ця вразливість зазвичай не застосовується до технології блокчейну.

Broken Access Control

Порушений контроль доступу подібний до порушеної автентифікації, але відрізняється від нього. При порушеній автентифікації зловмисник видає себе за неавторизованого користувача, тоді як при порушеному контролі доступу зловмисник отримує неавторизований доступ до захищених функцій.

Порушений контроль доступу складається з кількох можливих векторів атак, таких як обхід перевірок контролю доступу, редагування облікових записів інших користувачів, підвищення привілеїв, неправильні конфігурації CORS, які дозволяють неавторизований доступ до обмежених API, маніпуляції метаданими через маркери контролю доступу, такі як веб-токени JSON (JWT) або доступ неавторизовані веб-сторінки як непривілейований користувач, що може призвести до того, що зловмисники зможуть контролювати бізнес-функції або можливість зловмисників отримати всі дані. Рекомендується використовувати списки контролю доступу та забороняти доступ до функціональних можливостей за допомогою серверного коду, де зловмисники не можуть отримати доступ

або контролювати метадані.

Погано реалізовані механізми контролю доступу є однією з основних вразливостей смарт-контрактів Ethereum. Гаманець із декількома підписами на основі смарт-контрактів Parity відомий тим, що його двічі використовували через уразливості контролю доступу. В обох випадках смарт-контракти Parity мали функцію, яка дозволяла власнику викликати її та вимагати права власності на контракт, але не перевіряла, чи її викликали лише один раз.

Під час першої атаки зловмисник викликав цю функцію, щоб взяти під контроль контракти гаманця Parity та злити з них збережені кошти. У другому випадку була атакована подібна функція в бібліотечному контракті, що використовується всіма гаманцями Parity. Зловмисник взяв контроль, а потім самознищив функцію, зробивши всі гаманці Parity непридатними для використання та спричинивши назавжди втрату приблизно 1% усього ефіру.

Security Misconfiguration

Неправильна конфігурація безпеки є ще однією загальною вразливістю OWASP. Це стосується використання незахищених стандартних конфігурацій програмного забезпечення або використання конфігурацій, які роблять систему вразливою до атак. Таким чином, це один із найпоширеніших типів уразливостей.

Помилки в конфігурації безпеки стосуються веб-програм, які були неправильно налаштовані таким чином, що залишають їх під загрозами безпеці. Вони можуть включати конфігурації брандмауера, відкриті порти адміністрування, які наражають програму на віддалені атаки, або застарілі програми, які намагаються зв'язатися з програмами, яких більше не існує. Необхідно переконатися, що конфігурації проходять належний процес забезпечення якості та що такі зміни ретельно перевіряються, перевіряються та підтверджуються, що зменшує поверхню атаки від цього типу вразливості.

Блокчейни реалізовані як програмне забезпечення, що працює на клієнтських машинах у одноранговій мережі, тому має сенс, що неправильні

налаштування безпеки можуть вплинути на безпеку. В одному випадку користувачі гаманця Ethereum налаштували свої гаманці для прослуховування та прийняття зовнішніх команд через RPC (порт 8545). Зловмисники, скориставшись цією вразливістю, змогли викрасти ефір на суму 20 мільйонів доларів

Cross-Site Scripting (XSS)

Веб-сайт вразливий до атаки міжсайтового сценарію, якщо він містить ненадійні дані на веб-сторінці без їх попередньої перевірки та дезінфекції. Ця вразливість дозволяє зловмиснику запускати сценарії в браузері жертви.

Міжсайтовий сценарій, також відомий як XSS, має багато форм, які призводять до різних результатів залежно від типу XSS, який виконується, але зазвичай відбувається, коли зловмисники впроваджують у веб-програму шкідливі сценарії, які потім розкривають конфіденційну інформацію, внутрішні служби або розкривають файли cookie привілейованих користувачів.

Існує три типи XSS.

• Збережений XSS. Збережений XSS – це коли зловмисник вводить корисне навантаження на веб-сервер, яке зберігається, тому, коли інший користувач запитує доступ до сторінки, корисне навантаження запускається.

• Відображений XSS. Відображений XSS – це коли зловмисник вводить дані POST або URL-адресу та не зберігає, але все ще відображає. За допомогою цього зловмисники можуть створити зловмисну URL-адресу, щоб надіслати це корисне навантаження користувачам і отримати від них конфіденційну інформацію під час її запуску, наприклад їхній сеанс файли cookie.

• XSS на основі DOM – це коли зловмисник вставляє свої зловмисні дані в об'єктну модель документа (DOM) і не відображається у вихідному коді HTML і зможе активувати лише через саму консоль DOM. Наслідки вразливості такі ж, як і XSS.

Пом'якшити XSS можна шляхом належної перевірки та нормалізації

даних, отриманих із ненадійних джерел.

Уразливості міжсайтових сценаріїв можуть впливати на блокчейн-системи кількома різними способами. Уразливості міжсайтових сценаріїв використовувалися в іншому програмному забезпеченні, щоб дозволити запустити зловмисне програмне забезпечення для криптомайнінгу на комп'ютері жертви.

Уразливості міжсайтових сценаріїв можуть безпосередньо впливати на безпеку блокчейну, якщо вони існують у блокчейн-провідниках. Дослідники блокчейнів відображають дані транзакцій, які є ненадійними даними, які потенційно перебувають під контролем зловмисника. Якщо вразливість XSS існує в комбінованому провіднику блокчейнів і гаманці, використання провідника може надати доступ до приватного ключа користувача та контроль над його обліковим записом.

Insecure Deserialization

Серіалізація зазвичай використовується для передачі наборів даних через мережу. Якщо код десеріалізації реалізовано неправильно, зловмисна передача може дозволити зловмиснику використати вразливу машину.

Небезпечні атаки десеріалізації виникають, коли програми намагаються перетворити шкідливі дані, якими керує зловмисник, у внутрішні структури даних, якими керує програма. Впроваджуючи спеціально створені корисні навантаження, зловмисники можуть отримати контроль над змінними, функціями та внутрішніми станами програми. Це часто призводить до вразливості віддаленого виконання коду, а також до уязвимості операційної системи веб-програми.

У коді, де відбувається серіалізація, масив елементів серіалізується в потік байтів, який потім транспортується та обробляється серверною частиною веб-сайту після десеріалізації. Коли відбуваються атаки десеріалізації, зловмисники копіюють серіалізовані об'єкти, але вводять шкідливий код для обробки серверною частиною. Засоби пом'якшення цього типу атак включають використання безпечніших форматів обміну даними, таких як JSON або YAML, замість власних двійкових форматів,

використання надійних функцій десеріалізації та бібліотек, включаючи перевірки цілісності під час обробки серіалізованих даних і обмеження обсягу та можливостей серіалізованих операцій.

Хоча жодна зареєстрована атака не використовувала вразливості десеріалізації, системи блокчейну зазвичай використовують серіалізацію для передачі транзакцій. Оскільки дані транзакцій знаходяться під контролем (потенційно зловмисних) користувачів, вразливий код десеріалізації може призвести до компрометації систем блокчейну.

Using Components with Known Vulnerabilities

Використання компонентів із відомими вразливими місцями стосується використання певного програмного чи апаратного забезпечення з відомими вразливими місцями, незалежно від того, чи їхнє виробництво було припинено чи закінчився термін служби. Зловмисники, швидше за все, використовуватимуть поширені або відомі експлойти, щоб отримати доступ до систем, а не виявляти нові вразливості. Захист від цієї вразливості вимагає відстеження залежностей програми, належної документації, видалення невикористаних залежностей, видалення мертвого коду та включення залежностей до політик оновлення програми, процедур і життєвого циклу обслуговування.

Повторне використання коду в смарт-контрактах Ethereum навіть більш поширене, ніж у програмах, не пов'язаних з блокчейном. Насправді менше 10% смарт-контрактів Ethereum не використовують код повторно. Оскільки багато програмістів смарт-контрактів мають обмежений досвід роботи з технологією та пов'язаними з нею ризиками, це означає, що багато смарт-контрактів у блокчейні Ethereum містять відомі вразливості через повторне використання коду.

Insufficient Logging and Monitoring

Хоча безпечні процеси розробки важливі, це лише половина успіху. Після розгортання системи також важливо реєструвати та відстежувати події в системі на наявність аномалій, які можуть сигналізувати про атаку. Якщо цього не зробити, система може стати вразливою для використання за

допомогою векторів атак, пропущених під час процесу проектування.

Недостатнє ведення журналів і моніторинг означає відсутність належних механізмів журналювання для допомоги в моніторингу та виявленні інцидентів безпеки. Це дозволяє зловмисникам здійснювати свою діяльність непоміченим, що значно ускладнює завдання виявлення інцидентів і реагування на атаки. Журнали використовуються не лише для відстеження активності зловмисників або виявлення помилок та інших аномальних дій, які можуть відбуватися в програмі. Крім того, багато нормативних вимог залежать від наявності належних механізмів реєстрації та моніторингу.

Деякі ключові моменти пом'якшення включають впровадження засобів журналювання ключових операцій програми, автоматичний моніторинг і механізми виявлення, забезпечення дотримання належної політики зберігання журналів, наприклад журнали захищені та не можуть бути видалені

Блокчейн створює незмінний журнал дій, які виконуються в системі, що робить його ідеальним для цілей реєстрації. Однак, незважаючи на те, що журналювання є чудовим, це марно, якщо ніхто не дивиться на журнали. Багато смарт-контрактів на блокчейні «запускають і забувають» і не контролюються власниками, що робить їх потенційно вразливими для використання без виявлення.

ВИСНОВКИ

У роботі було описано 10 найбільших уразливостей OWASP і те, як вони впливають на бізнес і користувачів веб-додатків. Було окреслено типи атак, способи захисту веб-серверів.

Вплив на бізнес і користувачів залежить від типу атаки та конфіденційності даних, але в кінцевому підсумку підприємства зазнають фінансових втрат. Деякі з фінансових втрат, які можуть бути понесені, включають: оплату розслідування інциденту, компенсацію клієнту, кампанії з контролю збитків, час простою, поки інцидент розслідується та вирішується, можливі судові позови та штрафи, якщо підприємства порушують нормативні обмеження, такі як Загальний захист даних Положення (GDPR), порушення яких може досягати штрафів у розмірі до 4% від річного обороту або 20 мільйонів євро, залежно від того, що більше. З точки зору клієнта, довіру неможливо виправити. Дослідження показали, що як тільки сталося порушення, клієнти передають свій бізнес конкурентам, які, здається, більш серйозно ставляться до безпеки

Підсумовуючи моніторинг виявлення та усунення зазначених вразливостей можуть допомогти компаніям і кінцевим користувачам захиститися від найпоширеніших атак і, таким чином, обмежити їх зону атаки та ризику. Крім того, базові знання про ці типи атак можуть допомогти компаніям проактивно захищатися від них і захищати свої активи та дані.