Міністерство освіти і науки України

«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій

Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
завідувач кафедри кібербезпеки
та захисту інформації
_____ Н.В. Лукова-Чуйко
«18» червня 2021р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

дипломної роботи
бакалавра
(назва освітнього рівня)

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)

спеціальність 125 Кібербезпека
(код і назва спеціальності)

освітня програма Кібербезпека
(назва освітньої програми)

на тему: «Розробка елементів системи захисту інформації від cross-site scripting (XSS) загроз»

Виконавець: студентка IV курсу, групи КБ-42

_____ **Чуракова Єкатєріна Валеріївна**
(підпис) (прізвище ім'я по-батькові)

	Прізвище, ініціали	**Підпис**
Керівник	Мирутенко Л.В.	

Нормоконтроль	Зюбіна Р. В.	

Київ 2021

Міністерство освіти і науки України

«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій

Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
завідувач кафедри кібербезпеки
та захисту інформації
_____Н.В. Лукова-Чуйко
«11» листопада 2020 р.

ЗАВДАННЯ
на виконання дипломної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

Студентці **КБ-42** (група) **Чураковій Єкатєріні Валеріївні** (прізвище ім'я по-батькові)

Тема дипломної роботи Розробка елементів системи захисту інформації від cross-site scripting (XSS) загроз

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема дипломної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №2 від 11.11.2020 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Cross-Site Scripting атаки і вразливості, що до них призводять, SIEM-рішення для моніторингу мережі

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Опис видів і способі реалізації Cross-Site Scripting атак, аналіз існуючих рішень

для протидії та ідентифікації, архітектура та функціонал SIEM-систем, особливості написання правил кореляції у IBM QRadar SIEM, створення відповідного правила кореляції

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Розробка правила кореляції для виявлення XSS-активності, яке допоможе аналітикам швидко розслідувати пов'язані інциденти

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання:11 листопада 2020 року

Завдання видала		Л.В.Мирутенко
	(підпис)	(ініціали, прізвище)
Завдання прийняла до виконання		Є.В.Чуракова
	(підпис)	(ініціали, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	25.01.2021 – 27.01.2021	*виконано*
2	Аналіз літератури	28.01.2021 – 11.02.2021	*виконано*
3	Визначення шляхів реалізації XSS-атак	12.02.2021 – 24.02.2021	*виконано*
4	Огляд існуючих рішень протидії XSS-атакам	25.02.2021 – 24.03.2021	*виконано*
5	Аналіз функціоналу SIEM-систем	25.03.2021 – 07.04.2021	*виконано*
6	Підготовка системи до коректного збору трафіка	08.04.2021 – 20.04.2021	*виконано*
7	Написання правил кореляції	21.04.2021 – 05.05.2021	*виконано*
8	Оформлення пояснювальної записки	06.05.2021 – 08.06.2021	*виконано*
9	Підготовка до захисту	09.06.2021 – 21.06.2021	*виконано*

Завдання видала		Л.В. Мирутенко
	(підпис)	(ініціали, прізвище)
Завдання прийняла		Є.В. Чуракова

до виконання _____ (підпис) _____ (ініціали, прізвище)

Термін подання дипломної роботи до ЕК 08 червня 2021 року

РЕФЕРАТ

Дипломна робота складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел, має 64 сторінки основного тексту, 14 рисунків та 1 таблицю. Список використаних джерел містить 44 найменування і займає 5 сторінок.

Метою даної роботи є дослідження методів ідентифікації та протидії Cross-Site Scripting (XSS) атак.

У роботі проаналізована існуюча література з захисту від атак міжсайтового скриптингу, шляхів її реалізації та література про розробку засобів ідентифікації та протидії.

Розроблено правило кореляції для SIEM системи, яке ідентифікує можливі XSS атаки в середині мережі і надає аналітикам можливість швидкого розслідування інцидентів. Представлено процес підготовки системи до аналізу трафіка, що надходить, розробки і написання відповідного правила кореляції.

Ключові слова: Cross-Site Scripting атака, комп’ютерна мережа, ідентифікація вразливостей, SIEM система, аналіз логів, обробник подій, правила кореляції.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

XSS	–	Cross-Site Scripting
API	–	Application Programming Interface
(D)DoS	–	(Distributed) Denial-of-Service
DOM	–	Document Object Model
MFA	–	Multi Factor Authentication
OWASP	–	Open Web Application Security Project
SOP	–	Same Origin Policy
CSP	–	Content Security Policy
UIV	–	User Input Validation
SIEM	–	Security Information and Event Management
BB	–	Building Block
VPN	–	Virtual Private Network
QNI	–	QRadar Network Insights
CRE	–	Custom Rule Engine

ЗМІСТ

ВСТУП

За допомогою атак XSS кіберзлочинці можуть перетворити довірені веб-сайти на зловмисні, завдаючи тим самим надмірної шкоди і збиток не тільки жертвам, але й репутації власника довіреного веб-сайту. Веб-сайти, які є потенційно вразливими до XSS атак, зазвичай також є відкритими до значної кількості споріднених атак на систему користувача. Це може включати будь-що – від відображення невідповідного вмісту до завантаження шкідливого програмного забезпечення в систему без відома користувача.

У 2016 році компанія Palo Alto опублікувала результати дослідження найбільш розповсюджених пристроїв IoT з метою виявити слабкі місця у безпеці. Було визначено, що шістдесят відсотків пристроїв викликали загрозу безпеці своїх користувальницьких інтерфейсів саме через реалізацію XSS-атак. Сімдесят відсотків пристроїв з хмарними та мобільними компонентами дозволяють потенційному зловмиснику визначати дійсні облікові записи користувачів за допомогою викрадення даних облікових записів за допомогою міжсайтового скриптингу.

Крім того, заданими компанії Edgescan у 2020 році атаки міжсайтового скриптингу займають друге місце серед найбільш розповсюджених критичних атак веб-додатків (поступаючись лише SQL-ін’єкціям) і трапляються у 19 відсотках випадках.

Слід також зазначити, що за даними Impreva вразливість до атак міжсайтового скриптингу має найбільшу кількість пов’язаних веб-атак.

Тож, очевидно, що вразливості до атак XSS залишатимуться серйозною проблемою, незважаючи на значну кількість інструментів, які перевіряють код на їх наявність. Ін'єкційні вразливості можуть призвести до ланцюгових атак з

використанням пов’язаних з XSS загроз, що призводять до крадіжки особистих даних. Як приклад – переадресація запитів, які надходять, на сервер, викликаючи на ньому DDoS.

Об’єкт дослідження – процес захисту від атак міжсайтового скриптингу.

Предмет дослідження – механізми ідентифікації атак міжсайтового скриптингу.

Мета роботи – розробити правило кореляції для SIEM-систем, що ідентифікуватиме можливу cross-site scripting активність.

Завдання роботи:

- Визначення типів XSS-атак та шляхів їх реалізації.
- Аналіз рішень, наявних у науковій літературі, для виявлення і протидії Cross-Site Scripting.
- Аналіз можливостей, які надаються SIEM-рішеннями для безпеки мережі і аналізу потенційного XSS-трафіка.
- Розгляд функціоналу та архітектури SIEM-системи IBM QRadar.
- Використання можливостей QRadar для своєчасної ідентифікації та протидії активності, що може бути пов’язана з XSS-атаками.

Методи дослідження дипломної роботи:

- комплексний аналіз;
- структурний аналіз;
- порівняння;
- системний підхід.

Практичне значення роботи полягає у тому, що запропоновані в дипломній роботі рішення можуть бути використані, як елементи в SIEM системах.

РОЗДІЛ 1
АНАЛІЗ АТАК І НАУКОВИХ ПІДХОДІВ ДО ЇХ ІДЕНТИФІКАЦІЇ

1.1 Опис вразливості та визначення шляхів реалізації XSS

Станом на 2020 рік XSS (Cross-Site scripting) вразливість займає 7 місце в OWASP Top 10 [1]. Атаки міжсайтових сценаріїв (XSS) – це різновид ін’єкції, при якій шкідливі сценарії вводяться на надійні веб-сайти. Атаки XSS реалізовуються, коли зловмисник використовує веб-програму для надсилання зловмисного коду, як правило, у формі сценарію на стороні браузера, іншому кінцевому користувачеві. Зловмисник може використовувати XSS для надсилання шкідливого сценарію на сторону жертви. Браузер кінцевого користувача за неправильної конфігурації виконає сценарій [2]. Оскільки за замовчуванням сценарій вважається таким, що походить із надійного джерела, зловмисний сценарій може отримати доступ до будь-яких файлів cookie, маркерів сеансів або іншої конфіденційної інформації, що зберігається браузером і використовується на веб-сайті. Ці скрипти також можуть переписати вміст HTML-сторінки.

Реалізація атаки дозволяє зловмисникові здійснити наступні типи атак:

- Крадіжка куки

Зловмисник може отримати доступ до кукі-записів жертви, пов'язаних з веб-сайтом, використовуючи *document.cookie*, відправити їх на свій власний сервер і використовувати їх для вилучення конфіденційної інформації, наприклад, ідентифікаторів сеансів.

- Кейлоґґер

Зловмисник може зареєструвати слухача подій клавіатури, використовуючи *addEventListener*, а потім відправити всі натискання клавіш

користувача на свій сервер, таким способом отримавши конфіденційну інформацію.

- Фішинг

Зловмисник може вставити підроблену форму для входу на сторінку, використовуючи маніпуляції DOM, встановивши *action* атрибути форми на свій власний сервер, а потім отримати від користувача необхідну конфіденційну інформації.

Атаки типу Cross-Site Scripting класифікуються за двома критеріями: вектору атаки та способу взаємодії із користувачем.

Існує три вектори XSS атак, які зазвичай націлені на браузери користувачів [3]:

- Reflected XSS: Додаток або API включає неперевірені та неперевірені користувацькі дані як частину виводу HTML. Успішна атака може дозволити зловмисникові виконувати довільний HTML та JavaScript у браузері жертви. Зазвичай користувачеві доведеться взаємодіяти з деяким шкідливим посиланням, яке вказує на сторінку, контрольовану зловмисником, наприклад, зловмисні веб-сайти, що рекламують рекламу тощо.
- Stored XSS: Додаток або API зберігає невидалені дані користувача, які згодом переглядаються іншим користувачем або адміністратором. Зберігається XSS часто вважають високим або критичним ризиком.
- DOM (Document Object Model) XSS: фреймворки JavaScript, односторінкові програми та API, які динамічно включають керовані зловмисником дані на сторінку, є вразливими до DOM XSS. В ідеалі додаток не надсилатиме контрольовані зловмисником дані до небезпечних API.

Кожен с цих типів атак проводиться у декілька етапів.

Reflected XSS

1. Зловмисник надає жертві шкідливі дані, наприклад, за допомогою фішінгу.
2. Браузер жертви надсилає шкідливі дані на сервер у запиті.

3. Сервер вбудовує у відповідь шкідливі дані, як правило, у HTML. Шкідливі дані з’являються у відповіді у тому місці, де браузер (або сценарії, що працюють у браузері) буде розглядати їх як активний вміст.

4. Браузер надає (або іншим чином обробляє) відповідь, запускаючи шкідливі дані як команди.

Stored XSS

1. Зловмисник вставляє шкідливі дані в базу даних сервера.

2. Браузер жертви надсилає запит на сервер.

3. Сервер вбудовує у відповідь шкідливі дані, як правило, у HTML. Шкідливі дані з’являються у відповіді у тому місці, де браузер (або сценарії, що працюють у браузері) буде розглядати їх як активний вміст.

4. Браузер надає (або іншим чином обробляє) відповідь, запускаючи шкідливі дані як команди.

DOM XSS

1. Зловмисник надає жертві шкідливі дані, наприклад, за допомогою фішінгу.

2. Сценарії, що працюють у браузері, використовують шкідливі дані під час маніпулювання DOM на стороні клієнта як частину своєї звичайної роботи.

3. Браузер відображає (або іншим чином обробляє) DOM, запускаючи шкідливі дані як команди

Крім того, XSS поділяється на два основні види за критерієм способу взаємодії із жертвою:

1. Пасивні: Жертві необхідно виконати певні дії, щоб викликати оброблювач подій і запустити шкідливий скрипт. Для цього використовується соціальна інженерія нижчого рівня. Як тільки користувач взаємодіє з об’єктом соціальної інженерії, запуститься шкідливий скрипт. У випадку відсутності активності з боку жертви, код не буде активований.

2. Активні: Зловмиснику не потрібно заманювати жертву за спеціальними посиланнями, оскільки код вбудовується у бази даних або у виконувані файлі на сервері. Від користувача не потрібно ніякої активності. У форми введення, як правило, встановлений спеціальний обробник подій, що автоматично активується при взаємодії зі сторінкою. У підсумку всі користувачі, які перейшли за посиланням, стануть жертвами зловмисника.

Є і так звані «сліпі» XSS (blind XSS) – це варіант Stored XSS. Суть їх роботи полягає в тому, що зловмисник не зможе відразу побачити результат, або результат роботи пейлоада буде виводитися, наприклад, на якийсь інший сторінці, наприклад, форми зворотного зв'язку або публікації відгуків з попередньою модерацією повідомлень. Надіславши в тексті повідомлення скрипт, зловмисник не побачить жодної реакції від веб-додатки, проте на стороні адміністратора або модератора сервісу скрипт відпрацює, викликавши написану зловмисником функцію.

Головна проблема пошуку blind XSS в тому, що необхідно враховувати безліч контекстів, в яких може виконуватися код, наприклад, всередині одинарних або подвійних лапок, різних атрибутів.

Типові атаки XSS включають викрадення сеансів, поглинання облікового запису, обхід багатофакторної аутентифікації (MFA), заміну або деформацію вузла DOM (наприклад, троянські панелі входу), атаки на браузер користувача, такі як завантаження зловмисного програмного забезпечення, реєстрація ключів та інші атаки на стороні клієнта.

Основна мета міжсайтового скриптинга – крадіжка файлів cookie користувачів за допомогою вбудованого на сервері скрипта з відбором необхідних даних та використання їх для подальших атак та взломів. Зловмисник здійснює атаку користувачів не напряму, а з використанням уявного веб-сайту, який відвідує жертва, та вводить спеціальний JavaScript. У браузері у користувачів цей код відображається як єдина частина сайту. При цьому відвідуваний ресурс стає учасником XSS-атаки (рис. 1.1).

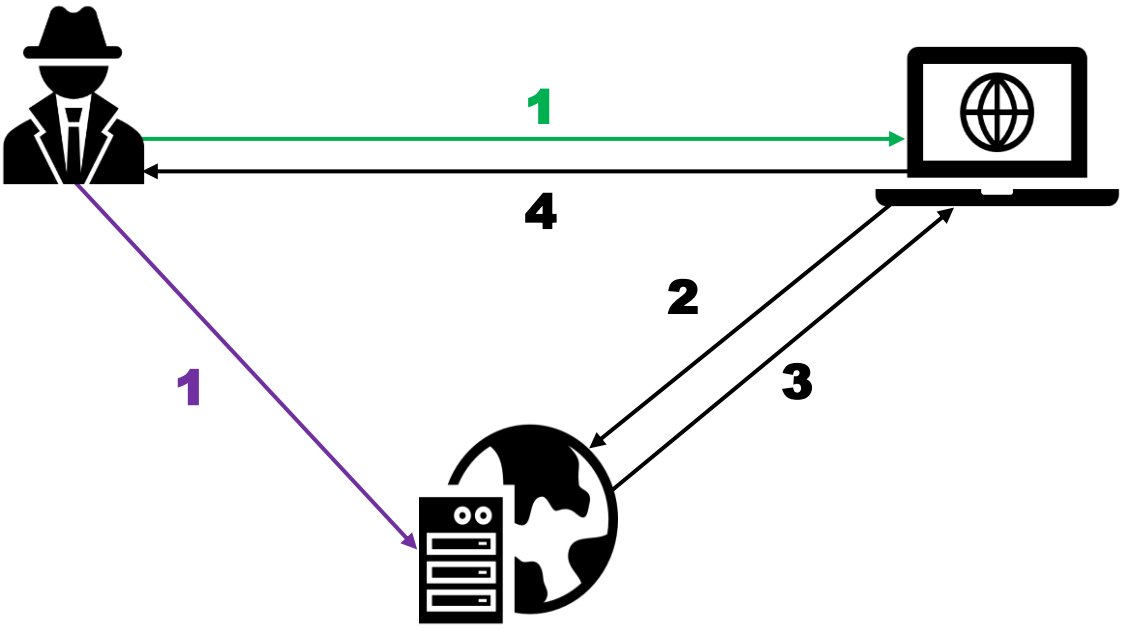


Рисунок 1.1. 1 - Зловмисник вводить шкідливі дані; (через соціальну інженерію у випадку Reflected та DOM; на сервери БД у разі збережених); 2 - Запит від жертви (вже включаючи шкідливі дані для Reflected атака або звичайний для Stored атаки); 3 – Сервер отримує шкідливі дані, включаючи їх у відповідь; 4 - Зловмисник отримує потрібну йому інформацію

Для реалізації атаки необхідна наявність суб'єктів XSS, які братимуть участь в процессі. Загалом, в атаці XSS присутні три учасники: веб-сайт, жертва, і зловмисник.

1. Веб-сайт видає HTML-сторінки для користувачів запитали їх. База даних веб-сайту є базою даних, яка зберігає деякі введені користувачами дані на сторінках сайту.

2. Жертва – це звичайний користувач веб-сайту, який звертається до сторінки за допомогою браузера.

3. Атакуючий – це зловмисник, який має намір розпочати атаку на жертву за рахунок використання веб-вразливості на сайті. Сервер зловмисника – це веб-сервер під контролем зловмисника для крадіжка конфіденційної інформації жертви.

Запуск зловмисного коду JavaScript можливий лише у браузері жертв, тому веб-сайт, який зайде користувачем, повинен мати веб-вразливості і

піддаватись атакам типу XSS. Для вчинення атаки зловмисник спочатку перевіряє ресурси на наявність уявних засобів через XSS за допомогою автоматизованих скриптів або ручного режиму пошуку. Зазвичай це стандартні форми, які можуть надсилати та приймати запрошення (коментарі, пошук, зворотний зв'язок). Проводиться повний збір сторінок із формами вводу та кожна з них сканується на наявність вразливостей. У найпростішому випадку перевірити наявність вразливості до XSS-атак можливо за допомогою запиту:

<script> alert ("cookie:" + document.cookie) </script>

Розробники основних популярних CMS регулярно усувають виявленні вразливості, але через можливість розширення функціоналу за рахунок модулів і плагінів, що створюються сторонніми розробниками, шанси на використання вразливостей, схильних до XSS атак, значно зростають, особливо в Joomla, DLE, Bitrix, Wordpress.

Ще один можливий варіант пошуку – використання сторінок, які обробляють GET-запити. У цьому випадку у посиланні на веб-ресурс має бути наявний параметр, замість якого у запиті можна передати скрипт.

У якості потенційних ризиків вразливості до XSS-атак, слід також враховувати, сторінки з помилками. Деякі сервери містять спеціальні повідомлення "*404 Page Not Found*" або повідомлення про помилки серверу, що описують запитувану сторінку або передані параметри. Якщо ці елементи не відфільтровані, вони забезпечують ідеально пропущений ґрунт для введення XSS.

Перевірка введення користувачем також є бар’єром, який захищає веб-додатки від атак на рівні програми. Більшість засобів тестування UIV не можуть виявити уразливості, пов’язані із семантикою.

Як було зазначено раніше, веб-додатки часто використовують код JavaScript, який вбудований у веб-сторінки для підтримки динамічної поведінки на стороні клієнта. Цей код сценарію виконується в контексті веб-браузера користувача. Щоб захистити середовище користувача від зловмисного коду

JavaScript, браузери використовують механізм пісочного боксу, який обмежує доступ скрипта лише до ресурсів, пов’язаних із початковим сайтом. Інколи ці механізми безпеки виходять з ладу, якщо користувача можна заманити завантажити шкідливий код JavaScript із проміжного, надійного сайту. У цьому випадку зловмисному сценарію надається повний доступ до всіх ресурсів (наприклад, маркерів автентифікації та файлів cookie), які належать довіреному веб-сайту. Висока гнучкість схем кодування HTML є однією з причини достатньо легкої реалізації та складної ідентифікації цього типу атак. Це пропонує зловмисникові багато можливостей для обходу вхідних фільтрів на стороні сервера, які повинні запобігти введенню шкідливих сценаріїв на надійні сайти. Крім того, розробка рішення на стороні клієнта непроста через складність ідентифікації коду JavaScript як зловмисного.

Загалом, зловмисний код може бути вбудований у скрипт наступними шляхами:

- Помилки у браузері

Через помилки браузер може виконувати скрипти в SVG, порушувати правило Same Domain Policy.

- Відсутність екранування спецсимволів HTML

Щоб браузер гарантовано не прийняв рядок за тег HTML, потрібно заекранувати п'ять символів: ' "& <>. Сервер може екранувати не всі символи (відомий недолік PHP), або веб-програміст просто забуває заекранувати рядок. Зазвичай в базах даних текст зберігається неекранованим, і екранувати потрібно всі призначені для користувача рядки кожен раз, коли вони вбудовуються в HTML. Багато сайтів дозволяють форматування тексту за допомогою будь-якої мови розмітки (HTML, BBCode, вікірозмітка). Часто не проводиться повний лексичний аналіз мови розмітки, а лише перетворення в «безпечний» HTML за допомогою регулярних виразів. Це спрощує програмування, однак вимагає досконального розуміння, якими шляхами скрипт може проникнути в результуючий HTML-код.

- Відсутність фільтрації атрибутів і їх значень в дозволених тегах

Типовим прикладом відсутності фільтрації атрибутів і їх значень є посилання *<a href="javascript:DoSomething()">*. Навіть якщо зовнішні посилання дозволені, протоколи *javascript:* і *data:* є потенційно небезпечними.

- Підміна кодування в заголовку сторінки

Сучасні браузери намагаються визначити кодування сторінки на ходу і інтерпретують html відповідно до цієї кодуванням. У разі, якщо тег <title> розташований до тега <meta> і заповнюється даними користувачів, хакер може вставити зловмисний html-код в кодуванні UTF-7, обійшовши таким чином фільтрацію деяких символів (зокрема символ відкриття тегу). Для захисту від даної уразливості слід явно вказувати кодування сторінки до будь-яких користувальницьких полів. HTML 5 прямо забороняє підтримку браузерами кодування UTF-7 як потенційно небезпечною.

- Через SQL-код

Якщо сайт допускає використання SQL-коду і виводить вміст БД без екранування можна провести атаку. Впровадженням SQL-коду в БД записується сторінка, що містить шкідливий код. Отримання доступу до цієї сторінки неминуче призведе до потрапляння шкідливого скрипта до браузеру жертви.

1.2 Аналіз наукових джерел для визначення існуючих підходів до протидії XSS

Аналіз наукових досліджень XSS-атак показав, що тема запобігання та протидії XSS є дуже актуальною, тому вона висвітлюється в публікаціями у багатьох збірниках конференцій та журналах. Більшість досліджень зосереджують увагу на запобіганні атакам та виявленню вразливостей. Методи динамічного аналізу становлять більшість серед рішень, запропонованих різними дослідженнями. XSS все ще залишається великою проблемою для веб-

додатків, незважаючи на велику кількість різноманітних рішень, що пропонуються на даний момент. Не існує єдиного рішення, яке зможе ефективно пом'якшити атаки XSS.

В існуючих рішеннях з детектування вразливостей до XSS-атак, таких як XSSF, XenoTix, Wapiti, XSpider [4], Nemesida Scanner [5], Acunetix Online Web Security Scanner [6], пошук здійснюється тільки у відкритій частині веб-ресурсу, тобто тієї, до якої можеа отримати доступ без попередньої авторизації. Це є істотним недоліком, так як XSS-вразливість може перебувати в недоступній для пошуку частині веб-ресурсу.

Спираючись на дослідження [7], можна стверджувати що двома домінуючими типами є атаки SQL-Injection (SQLi) та Cross Site Scripting Attacks (XSS), що складають 30% загальної кількості Інтернет-атак. У зазначеній роботі було запропоновано систему, яка використовує алгоритм MD5 та правила вираження граматики, маніпульовані в зворотному проксі, для зниження ризику введення SQL та XSS. Ця система пропонує серверне рішення для протидії атакам XSS. Система була протестована на стандартних тестових програмах, і у роботі продемонстровано вдосконалення при виявленні та обмеженні первинних атак XSS.

На основі аналізу основних механізмів класу вразливості автори роботи [8] запропонували підхід до оснащення сучасних мов програмування обов’язковими засобами для явного та безпечного генерування коду, які забезпечують чітке розділення між даними та кодом. Використовуючи приклад реалізації для мов Java та HTML/JavaScript відповідно, автори показали способи реалізації та застосування описаного підходу.

У роботі [9] науковцями було розроблене рішення Noxes, який є рішенням для запобігання атакам між сценаріями на стороні клієнта. Noxes діє як веб-проксі і використовує як ручні, так і автоматично генеровані правила, щоб знизити кількість можливих спроб виконання міжсайтових сценаріїв. За словами авторів Noxes ефективно захищає від витоку інформації із середовища

користувача, вимагаючи мінімальних взаємодій з користувачем та зусиль з налаштування.

Для вирішення проблеми валідації вхідних даних науковці [10] запропонували підхід для створення тестових вхідних даних для UIV на основі аналізу інформації на стороні клієнта. Зокрема, використовуючи інформацію поля вводу для генерації дійсних входів і подальшого порушення дійсних входів для генерування недійсних тестових входів. За результатами емпіричного дослідження розробники запевняють, що у порівнянні з існуючими сканерами вразливостей, їх підхід є більш ефективним у пошуку семантичних вразливостей UIV для веб-додатків.

Методи тестування «чорної скриньки» є загальним підходом для поліпшення якості програмного забезпечення та виявлення помилок перед розгортанням. Але процес, який піддає веб-програми сукупності неправильно сформованих входів з метою виявити помилки перевірки вхідних даних часто не можуть перевірити значну частину логіки веб-програми, особливо у випадку сторінок, до яких можна отримати доступ лише після заповнення складних форм, які перевіряють правильність наданих значень. Тому були розроблені [11] інструменти автоматизованого тестування, які можуть знайти відображені та збережені вразливості міжсайтових сценаріїв (XSS) у веб-додатках. Прикладом такої розробки є, з ядром, яке являє собою сканер уразливості чорних ящиків. Цей сканер вдосконалений методами, що дозволяють генерувати більш комплексні тестові кейси та досліджувати більшу частину програми. Експерименти дослідників демонструють, що подібний підхід здатний більш ретельно тестувати програми та виявляти більше помилок, ніж низка сканерів вразливості з відкритим кодом та комерційною мережею.

Атаки введення веб-додатків, такі як введення SQL та міжсайтові сценарії (XSS), є основною загрозою безпеці Інтернету. Кілька недавніх досліджень описують використання динамічного забруднення для зниження цих загроз. Наприклад, у роботі [12] представлено додаткове кодування символів, новий

підхід до динамічного забруднення на рівні символів, що дозволяє ефективно та точно розповсюджуватись через межі серверних компонентів, а також між серверами та клієнтами через HTTP. При цьому підході кожен символ має два кодування, які можна використовувати для розрізнення надійних та ненадійних даних. Невеликі модифікації лексичних аналізаторів компонентів, такі як інтерпретатор коду додатків, система управління базами даних та веб-браузер, дозволяють їм стати доповнюючими компонентами, здатними використовувати цю альтернативну схему кодування символів для забезпечення політики безпеки, спрямованої для запобігання ін'єкційним атакам, продовжуючи нормально функціонувати в інших відношеннях. Подібні підходи створені для подолання деяких слабких місця динамічного забруднення і забезпечення точного захисту від постійних міжсайтових атак сценаріїв, оскільки інформація про забруднення зберігається, коли дані передаються в базу даних і згодом отримуються прикладною програмою.

Більшість веб-додатків містять уразливі місця безпеки. Прості та природні способи створення веб-програми схильні до атак SQL-ін’єкцій та міжсайтових сценаріїв, а також до інших менш поширених вразливостей. У відповідь на це було розроблено багато інструментів для виявлення або зниження загальних вразливостей веб-додатків. Існуючі методи або вимагають зусиль розробника сайту, або схильні до помилкових спрацьовувань. У роботі [13] представлений повністю автоматизований підхід до безпечного зміцнення веб-додатків. Він базується на точному відстеженні забрудненості даних та спеціальному перевірці небезпечного вмісту лише в частинах команд і результатів, що надходять з ненадійних джерел. На відміну від попередньої роботи тих же авторів, в якій все, що походить від забрудненого введення, забруднене, новий підхід точно відстежує забрудненість у межах значень даних.

У роботі [14] представлено вивчення та вдосконалення LAPSE – програмного забезпечення для захисту, заснованого на статичному аналізі коду

для виявлення вразливостей системи безпеки в Java EE Applications. LAPSE був розроблений групою SUIF Стенфордського університету як плагін для Eclipse Java IDE. Останній стабільний випуск плагіна, LAPSE 2.5.6, датується 2006 роком, і він застарів із точки зору кількості виявлених вразливостей та його інтеграції з новими версіями Eclipse. Однак, розробники працюють над інтеграцією нової версії LAPSE +, розширеної версії LAPSE 2.5.6. Ця нова версія плагіна розширює функціональність попередньої, оновлюючись для роботи з Eclipse Helios, забезпечуючи більш широкий каталог вразливостей та вдосконалення для аналізу коду. Крім того, у процесі розробки було введено версію командного рядка LAPSE +, щоб зробити цей інструмент незалежним від Eclipse Java IDE. Ця версія командного рядка містить генерацію XML-звітів про потенційні уразливості, виявлені в програмі.

Уразливість міжсайтових сценаріїв (XSS) в основному спричинена неправильною роботою веб-додатків при обробці введених користувачем даних, вбудованих у веб-сторінки. Незважаючи на те, що розробники та аудитори безпеки часто використовують ультрасучасні захисні методи кодування та методи виявлення вразливості, недоліки XSS все ще залишаються у багатьох додатках через складність застосування цих методів, неадекватне впровадження цих методів та/або відсутність розуміння проблеми XSS. Для вирішення цієї проблеми деякі розробники пропонують підхід до аудиту коду, який відновлює захисну модель, реалізовану у вихідному коді програми, та пропонує вказівки для перевірки адекватності відновленої моделі проти атак XSS. На основі можливих схем реалізації захисних методів кодування підхід витягує всі такі захисні засоби, реалізовані для захисту кожного потенційно вразливого виводу HTML. Потім він вводить варіант графіка потоку управління, який називається графом потоку забрудненої інформації, як модель для перевірки адекватності артефактів захисту XSS. Автори [15] оцінили запропонований метод на основі експериментів на семи веб-додатках на базі Java. В аудиторських експериментах підхід був ефективним у відновленні всіх

захисних характеристик XSS, реалізованих у тестованих. Вилучені артефакти також виявились корисними для фільтрації хибнопозитивних випадків, про які повідомляв метод виявлення вразливості, та корисні для виправлення вразливих розділів коду.

Більшість веб-додатків використовують базу даних як задню частину для зберігання такої важливої інформації, як облікові дані користувачів, фінансова інформація та інформація про платежі, статистика компаній тощо. Ці веб-сайти постійно націлені на сильно мотивованих зловмисних користувачів для отримання грошової вигоди. Кілька вразливостей на стороні клієнта та сервера, таких як введення SQL та сценарії між сайтами, виявляються та використовуються зловмисними користувачами. Атаки введення SQL та уразливості сценаріїв між сайтами займають перше місце у списку десяти вразливостей проекту відкритих веб-програм безпеки. Запропоновано та використовується низка підходів до безпеки, таких як безпечне кодування, шифрування, статичний та динамічний аналіз коду для захисту веб-програм, але статистика показує, що ці вразливості все ще проявляються вгорі. У роботі [16] представлено інтегровану модель для запобігання атакам ін'єкції SQL та відображенню міжсайтової атаки сценаріїв у реалізації на основі PHP. Ця модель є більш ефективною для запобігання атаці введення SQL та відображеної атаки сценаріїв між сайтами у виробничому веб-середовищі. Механізм розділений на два режими, безпечний режим та виробниче середовище. У безпечному режимі створюється модель запиту безпеки для ін'єкції SQL та дезінфікуючу модель для відображеної атаки сценаріїв між сайтами для кожного ідентифікованого SQL-запиту для атак SQL-ін'єкції та вхідних точок входу для відображених атак сценаріїв між сайтами. У виробничому середовищі вхідні записи, що створюють динамічні запити SQL, перевіряються на основі моделі запиту безпеки, що генерується в безпечному режимі, а звичайний введений користувачем текст перевіряється за допомогою дезінфікуючої моделі, що входить до коду в безпечному режимі. Результати та

аналіз показують, що запропонований підхід є простим та ефективним для запобігання загальним вразливостям введення SQL та відображеним вразливостям сценаріїв між сайтами.

Було виявлено, що майже 70% останніх атак у веб-програмах були здійснені навіть тоді, коли системи були захищені добре прокладеними брандмауерами та системами виявлення вторгнень. За підрахунками аналітиків, більше 20% атак відбулися через вразливості між сайтами (XSS). За даними, понад 40% вразливостей, підтверджених у рамках загальної вразливості (CVE), були засновані на PHP Script у 2006 році. Із цих вразливостей, заснованих на PHP, 45% класифікуються як XSS. Науковці [17] організували ці помилки у просту таксономію та зіставили CVE із загальним переліком слабкості Mitre Corp, у результаті чого було створене загальне перерахування вразливостей до XSS-атак. За допомогою переліка можна розпізнавати загальні типи шаблонів розробників, що призводять до помилок кодування в PHP, і, як наслідок, до вразливості XSS, тоді як розробники можуть виявляти та виправляти наявні помилки під час побудови програмного забезпечення.

Вразливості міжсайтових сценаріїв (XSS) дають можливість хробакам швидко поширюватися серед широкого кола користувачів на популярних веб-сайтах. На сьогоднішній день виявлення хробаків XSS в основному не досліджено. У роботі [18] автори запропонували клієнтське рішення для виявлення хробаків XSS. Воно засноване на розумінні того, що хробак XSS повинен поширюватися від одного користувача до іншого шляхом реконструкції та розповсюдження свого корисного навантаження. Описаний у статті підхід створений для запобігання розповсюдженню хробаків XSS шляхом моніторингу вихідних запитів, які надсилають корисні навантаження, що самовідтворюються, перехоплюючи всі HTTP-запити на стороні клієнта і порівнюючи їх із вбудованими скриптам. Також автори запевняють, що реалізували міжплатформене розширення Firefox, яке здатне виявляти всі існуючі самореплікаційні хробаків XSS, що поширюються на стороні клієнта.

Результати тестування рішення показують, що воно вимагає низьких накладних витрат і не повідомляє про помилкові спрацьовування при тестуванні на популярних веб-сайтах.

Міжсайтові атаки сценаріїв є однією з основних загроз безпеці сучасних веб-додатків. Сучасні підходи до подолання вразливостей міжсайтових сценаріїв покладаються на механізми захисту на основі сервера або клієнта. Хоча ефективні для багатьох атак, механізми захисту на стороні сервера можуть зробити клієнта вразливим, якщо сервер погано виправлений. З іншого боку, клієнтські механізми можуть спричинити значні витрати на клієнтській системі. Тому деякі розробники [19] пропонують гібридні рішення клієнт-сервер, яке поєднує переваги обох архітектур. Такі рішення на основі проксі використовують сильні сторони як виявлення аномалій, так і контролю аналізу потоку, щоб забезпечити точне виявлення. Доцільність та точність такого підходу продемонстровано, зокрема у роботі [20], за допомогою розширеного тестування з використанням реальних сценаріїв міжсайтових сценаріїв.

Noncespaces, веб-програма запропонована науковцями [21], рандомізує HTML-теги та атрибути в кожному документі, перш ніж доставити їх клієнту. Поки зловмисник не може вгадати випадкове відображення, клієнт може розрізнити надійний вміст, створений веб-програмою, та ненадійний вміст, наданий зловмисником. Для реалізації Noncespaces з мінімальними змінами у веб-додатках використовується популярна архітектура веб-додатків для автоматичного застосування Noncespaces до статичного вмісту, обробленого за допомогою популярного механізму шаблонів PHP. Щоб продемонструвати ефективність запропонованого рішення, розробники удосконалюють мову політики для Noncespaces, впроваджують режим навчання для сприяння розробці політики та проводять тестування безпеки сформованої політики для великих веб-додатків.

Cross Site "Scripter" (також відомий як XSSer) – це автоматичний фреймворк для виявлення і експлуатації XSS-вразливостей в веб-додатках [22].

Містить кілька опцій для обходу певних фільтрів і спеціальні техніки впровадження коду.

Ключові особливості інструменту:

- можливість створення зображень або відеофайлів з XSS-пейлоадом для подальшого завантаження в веб-додаток;
- пошук для впровадження XSS за допомогою Google Dorks запитів;
- перевірка наявності XSS-фільтрів у цільового веб-додатки;
- опція генерації пейлоада для спроб обходу систем WAF;
- виявлення Blind XSS.

XSStrike – це пакет виявлення XSS, оснащений чотирма рукописними синтаксичними аналізаторами, інтелектуальним генератором корисного навантаження, потужним механізмом фаззінга і швидким сканером [23]. Він розпізнає відповідь за допомогою декількох аналізаторів і потім обробляє корисні дані, які гарантовано будуть працювати за допомогою контекстного аналізу, інтегрованого в механізм фаззінга і має наступні можливості:

- фаззінг;
- технологія злому контексту;
- інтелектуальна генерація пейлоадов;
- підтримка методів GET & POST;
- підтримка файлів cookie;
- виявлення WAF;
- пейлоади ручної роботи для фільтрації та WAF-ухилення;
- приховане виявлення параметрів.

Крім того, слід згадати найпростіші способи захисту, що засновані на забороні сценаріїв між сайтами, тобто відмові користувачам у можливості використовувати будь-яку форму HTML у своїх даних. Якщо все ж таки необхідно дозволити HTML, треба грамотно розроблювати і застосовувати процедури фільтрації. У багатьох великих високопрофільних сайтах були

виявлені дірки XSS в результаті дірок циклів фільтрації, включаючи Yahoo та Hotmail.

Один із найефективніших видів реалізації фільтрації – це дозволити дуже обмежений список HTML та заборонити всі інші теги. Це можна реалізувати, розділивши текстовий блок на всі знаки <, а потім прочитавши до першого пробілу в кожному елементі, щоб побачити, який тип тегу. Якщо тег був впізнаваний і дозволений, зміщення закриваючого тегу має бути захоплене і замінений підрядок чистою його атрибутною версією. Якщо тег не був дозволений, він має бути видалений.

Наприклад, для побудови фільтрації та автоматичної заміни спецсимволів може бути використаний наступний код:

$filter = array("<", ">");

$_GET['q']=str_replace ($filter, "|", $_GET['q'])

У багатьох сучасних фреймворках реалізований захист від XSS-атак. Механізм XSS Filtering в Code Igniter, штатний функціонал шаблонізаторів у Django і RoR, Web Protection Library і механізм Request Validation в ASP.NET.

XSS спрямована не просто на виконання довільного сценарію, а на його виконання в контексті джерела конкретного сайту з метою обходу політик єдиного джерела (SOP). Це важлива концепція безпеки для деяких мов програмування на стороні клієнта, таких як JavaScript [24]. Політика дозволяє сценаріями, що знаходяться на сторінках одного сайту, доступ до методів і властивостей один одного без обмежень, але запобігає доступ до більшості методів і властивостей для сторінок на різних сайтах. Однакові джерела – це джерела, у яких збігаються три ознаки: домен, порт ір-протокол. В результаті обходу правил SOP, можливе отримання доступу до даних і функціоналу клієнтської частини веб-додатки в рамках користувальницької сесії і з правами її користувача. У цьому випадку XSS є атакою, в першу чергу, на веб-додаток, що реалізує загрозу саме в ньому, а не тільки в браузері користувача.

Однак при впровадженні пейлода з одного сайту на інший SOP не використовуватиметься для сайту з впровадженим пейлоадом. Тому на зміну SOP прийшов CSP [25], основне призначення якого полягає в тому, щоб захистити користувача від загроз міжсайтового виконання сценаріїв. CSP описує безпечні джерела завантаження ресурсів, встановлює правила використання вбудованих стилів, скриптів, а також динамічної оцінки JavaScript. Найголовніше – завантаження з ресурсів, що не входять до «білого списку», блокується.

У дослідженні [26] були опубліковані результати аналізу способів обходу XSS-захисту на рівні додатків. Для реалізації використовувались кілька віртуальних машин, на яких запускалися популярні браузери Google Chrome, Opera, Mozilla Firefox і Internet Explorer. Дослідник вивчав комерційні та відкриті продукти: F5 Big IP, Imperva Incapsula, AQTRONIX WebKnight, PHP-IDS, Mod-Security, Sucuri, QuickDefence, Barracuda. У результаті для кожного продукту був представлений хоча б один XSS-вектор, який дозволяв здійснити обхід захисту, підтверджуючи факт відсутності універсального надійного рішення протидії XSS-атакам.

Цілий ряд рішень пропускали JS-події "*onwheel*" і "*onshow*" – вони дозволяють виконати шкідливий сценарій при прокручуванні мишею і при показі елемента меню відповідно. Зокрема, до цієї помилку виявилися схильні F5 Big IP, Barracuda. Quick Defense також виявився нездатним виявити ін'єкцію шкідливого коду за допомогою JS-подій "*onsearch*" і "*ontoggle*".

За допомогою подвійного URL-кодування, а також техніки, що дозволяє представити будь-JS-код за допомогою набору з 6 символів, дослідник обійшов XSS-фільтри відразу декількох рішеннях. Крім того, правила PHP-IDS містили помилку, яка дозволяла зловмисникові обійти фільтри за допомогою використання тега *svg*. А також була виявлена можливість семібітного представлення даних в кодуванні us-ascii, що сприймається браузерами Internet Explorer 6 і 7: *¼script¾alert (¢ xss ¢) ¼ / script¾*.

Висновки за розділом 1

Міжсайтовий скриптинг (XSS) – це атака націлена на ін’єкцію коду, що дозволяє зловмисникові виконати шкідливий JavaScript в браузері іншого користувача.

XSS-атаки часто поділяються на три типи:

- Збережені XSS, де шкідливий рядок формується з використанням бази даних веб-сайту.
- Відображені XSS, де шкідливий рядок формується із запиту жертви.
- DOM-моделі XSS, де уразливість виникає в коді на стороні клієнта, а не на боці серверного коду.

Існує багато способів реалізації атаки, серед яких помилки у браузері, відсутність екранування спецсимволів HTML, відсутність фільтрації атрибутів і їх значень в дозволених тегах, підміна кодування в заголовку сторінки, через SQL-код.

Для реалізації атаки необхідна наявність суб’єктів XSS, які братимуть участь в процесі. Загалом, в атаці XSS присутні три учасники: веб-сайт, жертва, і зловмисник.

XSS досі є однією з найпоширеніших вад безпеки веб-сайтів, оскільки вразливими є навіть такі ресурси, як Facebook, Twitter та багато інших, які зазнали цієї помилки. Завдяки своїй розповсюдженості та небезпеці, XSS входить у топ-10 найнебезпечніших недоліків безпеки в OWASP (Open Web Application Security Project). Крім того, вразливими також є пристрої з мобільними та хмарними компонентами і деяке мережеве обладнання.

Підвищує рівень вразливості також використання застарілих версій JavaScript, які все ще є широко використовуваними. Крім того, міжсайтовий скриптинг має найбільшу кількість пов’язаних атак, до реалізації яких відкриває доступ вразливість XSS.

Методи протидії міжсайтовому скриптингу можуть включати наступні дії:

1. Використання екранізації вхідних\вихідних даних через зміну вбудованих функції для очищення коду від шкідливих скриптів. До них можуть відноситись такі функції, як *htmlspecialchar()*, *htmlentities()* та *strip_tags()*.

2. Для кожної веб-сторінки необхідно вказати кодування (наприклад, ISO-8859-1 або UTF-8) для користувацьких полів.

3. Встановлення прапорця *HttpOnly*, який робить клієнтські кукі недоступними через сценарії JavaScript.

4. Використання політику безпеки вмісту (CSP, Content Security Policy), заголовку, який дозволяє в явному вигляді оголосити «білий список» джерел, з яких можна завантажити дані.

Однак, зазначені методи можуть істотно скоротити ризик атаки, але не усунути загрозу в повній мірі.

Існує також багато автоматичних засобів пошуку вразливості і сканерів безпеки, основним недоліком яких є пошук тільки у відкритій частині веб-ресурсу і ігнорування тієї частини, для якої необхідна авторизація. Також, кожний з інструментів використовує свої алгоритми пошуку вразливості, які базують на відомих шаблонах і сигнатурах. Тож, навіть при відсутності вразливості за результатами сканування, ресурс все ще може залишатись вразливим для нестандартної експлуатації.

Аналіз досліджень XSS щодо вразливостей та можливих атак показав, що тема запобігання та протидії XSS є дуже актуальною, тому вона висвітлюється в публікаціях у багатьох збірниках конференцій та журналах. Більшість досліджень зосереджені на запобіганні атакам та виявленню вразливих місць у конфігурації додатків. Методи динамічного аналізу представляють більшість рішень, запропонованих різними дослідженнями. XSS все ще є великою проблемою для веб-додатків, незважаючи на велику кількість різноманітних

рішень, що доступні в даний час. Не існує єдиного рішення, яке зможе ефективно протидіяти атакам XSS у ста відсотках випадків.

У багатьох сучасних фреймворках реалізований захист від XSS-атак. Механізм XSS Filtering в Code Igniter, штатний функціонал шаблонізаторів у Django і RoR, Web Protection Library і механізм Request Validation в ASP.NET. Але слід зазначити наступне:

- серверні фреймворки у веб-додатках мало ефективні проти DOM-based XSS;
- жоден з існуючих нині механізмів автоматичного захисту від XSS не є універсальним і позбавленим тих чи інших обмежень, які необхідно враховувати;
- функціонал, реалізований цими механізмами, спрямований на боротьбу з конкретним класом атак і не може захистити від появи в коді вразливостей недостатньої обробки даних.

Виходячи із зазначених висновків, було вирішено використати функціонал SIEM-систем для конфігурації її таким чином, щоб мати можливість ідентифікації та протидії Cross-Site Scripting атакам. Таким чином, були поставленні наступні завдання для їх вирішення у наступних розділах дипломної роботи.

- Огляд структури і архітектури SIEM-систем
- Аналіз можливостей, які надаються SIEM-рішеннями для безпеки мережі.
- Ознайомлення з SIEM-системою IBM QRadar.
- Використання можливостей QRadar для формування елементів системи захисту для своєчасної ідентифікації та протидії активності, що може бути пов’язана з XSS-атаками.

РОЗДІЛ 2
МОЖЛИВОСТІ SIEM-СИСТЕМИ ДЛЯ ВИРІШЕННЯ ЗАДАЧ ІДЕНТИФІКАЦІЇ

2.1 Огляд і структура Security Information and Event Management (SIEM) систем

Інструменти SIEM є важливою частиною системи захисту даних: вони агрегують дані з декількох систем та аналізують ці дані, щоб виявити ненормальну поведінку або потенційні кібератаки. Інструменти SIEM надають центральне місце для збору подій та сповіщень. Основний принцип системи SIEM полягає в тому, що дані про безпеку інформаційної системи збираються з різних джерел, і результат їх обробки надається в єдиному інтерфейсі, доступному для аналітиків безпеки, що полегшує вивчення характерних особливостей, відповідних інцидентів безпеки. SIEM являє собою об'єднання систем управління інформаційною безпекою (Security Information Management, SIM) і управління подіями безпеки (Security Event Management, SEM) в єдину систему управління безпекою. Сегмент SIM, в основному, відповідає за аналіз історичних даних, намагаючись поліпшити довгострокову ефективність системи і оптимізувати зберігання історичних даних. Сегмент SEM, навпаки, робить акцент на вивантаженні з наявних даних певного обсягу інформації, за допомогою якого можуть бути негайно виявлені інциденти безпеки. У міру зростання потреб в додаткових можливостях безперервно розширюється і доповнюється функціональність даної категорії продуктів [27].

Технологія SIEM часто допомагає компаніям зменшити кількість порушенн безпеки та покращити виявлення загроз. Інфографіка AlienVault та "2019 SIEM Survey Report" показали, що 76 відсотків фахівців з кібербезпеки повідомили, що використання інструментами SIEM в їх організації призвело до зменшення кількості порушень безпеки. Крім того, 46 відсотків респондентів

заявили, що платформа SIEM у їх організації виявляє принаймні половину всіх інцидентів у сфері безпеки [28].

Крім того, інструменти SIEM, як правило, забезпечують звітування про відповідність – те, що є надзвичайно цінним для підприємств, які повинні дотримуватись Загального регламенту про захист даних (General Data Protection Regulation, GDPR) та інших установчих документів щодо захисту даних. Інструменти SIEM часто оснащені можливостями звітування про відповідність, що гарантує те, що IT-команди можуть використовувати ці інструменти для швидкого виявлення та вирішення проблем безпеки, перш ніж вони призведуть до порушень.

Інструменти SIEM також допомагають пришвидшити реагування та ліквідацію подій. Інструменти SIEM, як правило, прості в розгортанні, і їх часто можна використовувати в поєднанні зі сторонніми інструментами безпеки бізнесу [29]. Таким чином, інструменти SIEM іноді зменшують необхідність наймати додаткових фахівців з кібербезпеки.

Однією з головних цілей використання SIEM-систем є підвищення рівня інформаційної безпеки в наявній архітектурі за рахунок забезпечення можливості маніпулювати інформацією про безпечність та здійснювати попереджувальне управління інцидентами і подіями безпеки в близькому до реального часу режимі.

Випереджаюче управління інцидентами і подіями безпеки полягає в прийнятті рішень ще до того, як ситуація стане критичною [30]. Таке управління може здійснюватися з використанням автоматичних механізмів, які прогнозують майбутні події на основі історичних даних, а також автоматичного підстроювання параметрів моніторингу подій до конкретного стану системи.

SIEM представлено додатками, приладами або послугами, і використовується також для журналювання даних і генерації звітів з метою сумісності з іншими бізнес-даними [31].

Поняття управління подіями інформаційної безпеки (SIEM), введене фахівцями з компанії Gartner в 2005 році, описує функціональність збору, аналізу та подання інформації від мережевих пристроїв і пристроїв безпеки, додатків ідентифікації (управління обліковими даними) і управління доступом, інструментів підтримки політики безпеки і відстеження вразливостей, операційних систем, баз даних і журналів додатків, а також відомостей про зовнішні загрози [32]. Основна увага приділяється управлінню привілеями користувачів і служб, службами каталогів і іншим змінам конфігурації, а також забезпечення аудиту та огляду журналів, реакцій на інциденти.

Основні можливості, які надаються SIEM-системами, включають наступні:

- Агрегація даних: управління журналами об'єднує дані з багатьох джерел, включаючи мережу, безпеку, сервери, бази даних, програми, забезпечуючи можливість консолідації даних, що контролюються, щоб уникнути пропуску важливих подій.
- Співвідношення: шукає загальні атрибути та пов’язує події разом у значущі пакети. Ця технологія надає можливість виконувати різноманітні методи кореляції для інтеграції різних джерел, щоб перетворити дані на корисну інформацію. Кореляція, як правило, є функцією частини управління подіями безпеки повного рішення SIEM.
- Оповіщення: автоматизований аналіз корельованих подій
- Інформаційні панелі: інструменти можуть брати дані про події та перетворювати їх на інформаційні діаграми, щоб допомогти побачити шаблони або визначити діяльність, яка не формує стандартний шаблон.
- Відповідність: додатки можуть використовуватися для автоматизації збору даних про відповідність, виготовлення звітів, що адаптуються до існуючих процесів безпеки, управління та аудиту.
- Зберігання: використання довгострокового зберігання історичних даних для полегшення співвідношення даних з часом та забезпечення

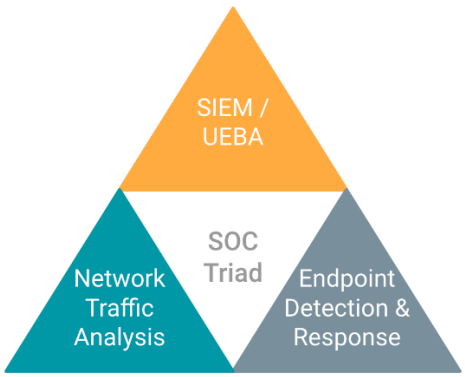
збереження, необхідного для вимог відповідності. Тривале збереження даних журналу є критично важливим при розслідуваннях, оскільки малоймовірно, що виявлення порушення мережі буде відбуватися саме під час порушення.

- Криміналістичний аналіз: можливість пошуку в журналах на різних вузлах та у різні періоди часу на основі конкретних критеріїв.

Системний аналіз більшості даних та спостереження за подіями відбувається над даними, що збираються з брандмауерів, антивірусного ПЗ, мобільних пристроїв, об'єктів мережевої інфраструктури, відеоспостереження, СУБД, DLP систем. Зібрані дані зберігаються в репозиторіях, видаються за запитом моделей аналізу даних [33]. Результатом роботи моделей є висновки у визначеній довільній формі, оперативна кореляція даних про події, а також згенеровані попередження.

Багатофункціональні SIEM можуть інтегруватися з хмарними сервісами для отримання даних про хмарну інфраструктуру або додатки SaaS (Software as a Service).

Також, SIEM-системи є одним із компонентів, які формують SOC (Security Operation Center) [34]. Основною місією SOC є моніторинг та попередження безпеки. Це включає збір та аналіз даних для виявлення підозрілої діяльності та покращення безпеки організації. Дані про загрози збираються з брандмауерів, систем виявлення вторгнень, систем запобігання вторгненню, а також SIEM-систем (рис 2.1). Оповіщення розсилаються членам команди SOC, як тільки виявляються розбіжності, ненормальні тенденції чи



інші показники компрометації.

Рисунок 2.1. SIEM-система є необхідною складовою Security Operation Center

Як правило, SIEM-система розгортається над захищеною інформаційною системою та має архітектуру «джерела даних» → «зберігання даних» → «сервер додатків». SIEM-рішення є інтегрованими пристроями (все-в-одному) або комплексами, що складаються з кількох компонентів. Розширена архітектура передбачає більшу продуктивність та найкращі можливості для масштабування, а також дозволяє розширити SIEM-рішення у складній IT-інфраструктурі.

Важливим компонентом систем є агенти, які виконують первинну обробку та фільтрацію, а також збирають події безпеки. Після цього, зібрана та відфільтрована інформація про події безпеки надходить у сховище даних, де вона зберігається у внутрішньому форматі представлення з метою подальшого використання та аналізу сервера додатків. Передача інформації від джерел даних може здійснюватися кількома способами:

1. Джерело ініціює передачу подій (наприклад, відправляє по syslog-протоколу);
2. Події з джерела забираються пасивно.

Для збору передачі необхідної інформації використовується агентний збір та безагентний збір інформації, причому в деяких SIEM-системах для частини джерел доступні обидва способи. Агентний спосіб передбачає використання спеціальної програми-агента, безагентний – налаштування джерела подій, такі як створення додаткових облікових записів, дозвіл віддаленого доступу і/або використання додаткових протоколів.

Для вирішення поставлених завдань SIEM-системи першого покоління застосовують нормалізацію, фільтрацію, класифікацію, агрегацію, кореляцію та пріоритетність подій, а також генерацію звітів та попереджень [35]. У SIEM-системах нового покоління до них слід додати також аналіз подій, інцидентів та

їх наслідків, а також прийняття рішень та візуалізацію. У загальному випадку алгоритм роботи SIEM-системи виглядає так, як на рисунку 2.2.

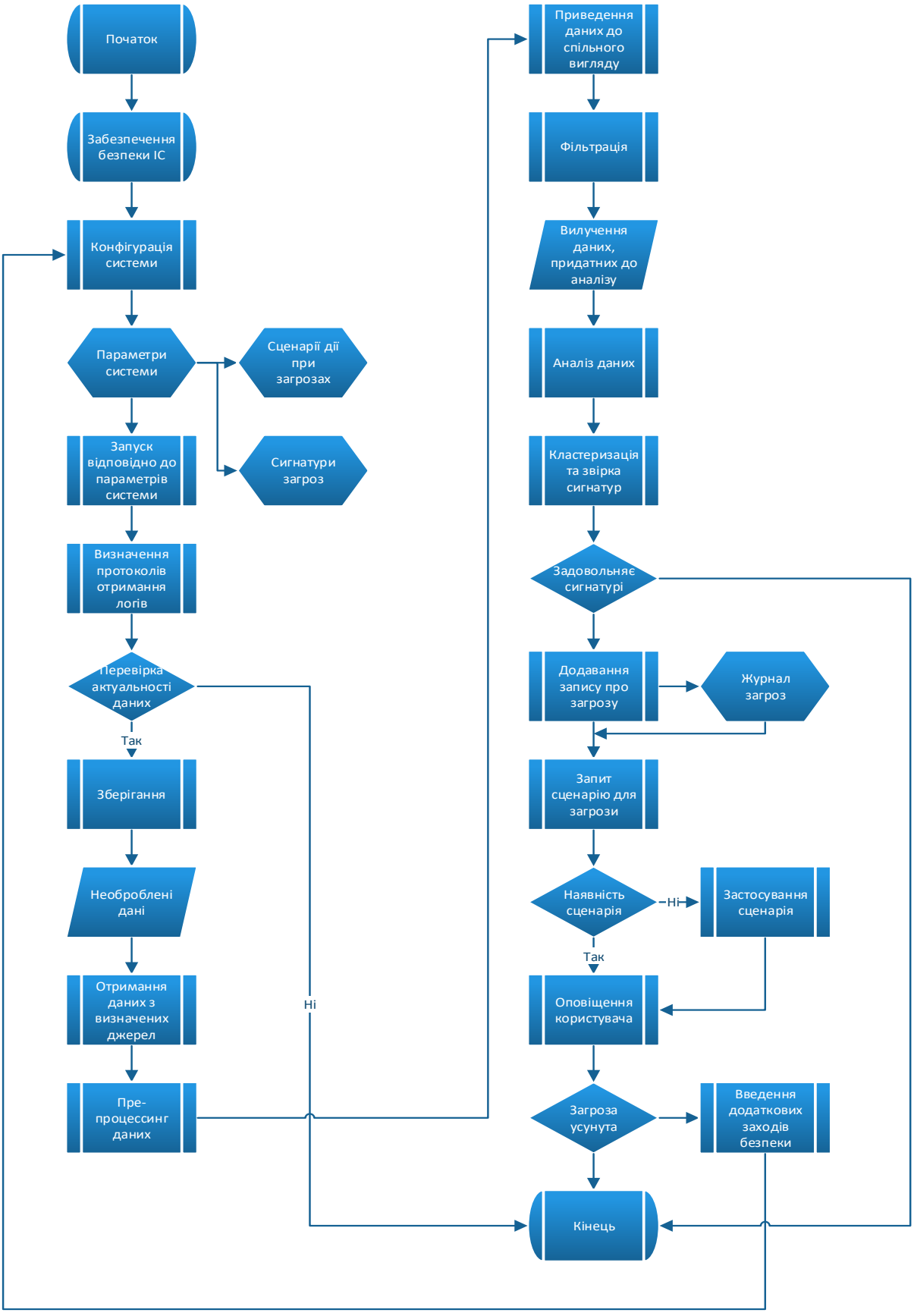


Рисунок 2.2. Алгоритм функціонування SIEM

Нормалізація приводить формати записів журналів, зібраних з різних джерел, до єдиного внутрішнього формату, який потім буде використаний для їх зберігання та подальшої обробки. Фільтрація подій безпеки полягає у видаленні надлишкових подій з потоків, які надходять у систему. Класифікація дозволяє визначити їх належність заданими класам. Агрегація об'єднує події, які схожі за визначеними ознаками. Кореляція виявляє взаємозв'язок між різнорідними подіями. Пріоритетність визначає значущість і критичність подій безпеки на основі правил, визначених у системі. Аналіз подій, інцидентів та їх наслідків включає процедури моделювання подій, атак та їх наслідків, аналіз вразливостей та захищеності систем, визначення параметрів порушників, оцінки ризику, прогнозування подій та інцидентів. Генерація звітів та попереджень означає формування, передачу, відображення або друк результатів функціонування. Візуалізація передбачає представлення в графічному вигляді даних, що характеризують результати аналізу безпеки та стану захищеної системи та її елементів.

Поряд з правилами кореляції, SIEM також може використовувати моделі. Моделі дещо відрізняються від правил кореляції, але якщо їх правильно застосувати, це може бути настільки ж корисним. Замість того, щоб використовувати кореляцію "один на один", модель вимагає декількох кроків, щоб викликати попередження. Зазвичай це означає одноразове спрацювання правила, за яким слідує аномальна поведінка. Прикладом може слугувати ситуація входу користувача з іншого місця, ніж зазвичай, а потім здійснення великої передачі файлів [36]. Це може бути надзвичайно корисним, оскільки одна подія не обов'язково означає компрометацію сервера або мережі організації.

2.2 Можливості SIEM системи IBM QRadar

IBM QRadar Security Information and Event Management (SIEM) допомагає фахівцям з безпеки забезпечити високу точність виявлення загроз і розстановки пріоритетів. Дане рішення дає можливість швидко реагувати на інциденти і мінімізувати їхні наслідки. Консолідація подій в журналах і потоків даних, що надходять в мережу з тисяч пристроїв, кінцевих точок і додатків, дає QRadar можливість зіставляти різноманітну інформацію, об'єднувати взаємопов'язані події і видавати точкові попередження для прискорення аналізу і усунення інцидентів [37]. QRadar SIEM доступний як локально, так і в хмарному варіанті.

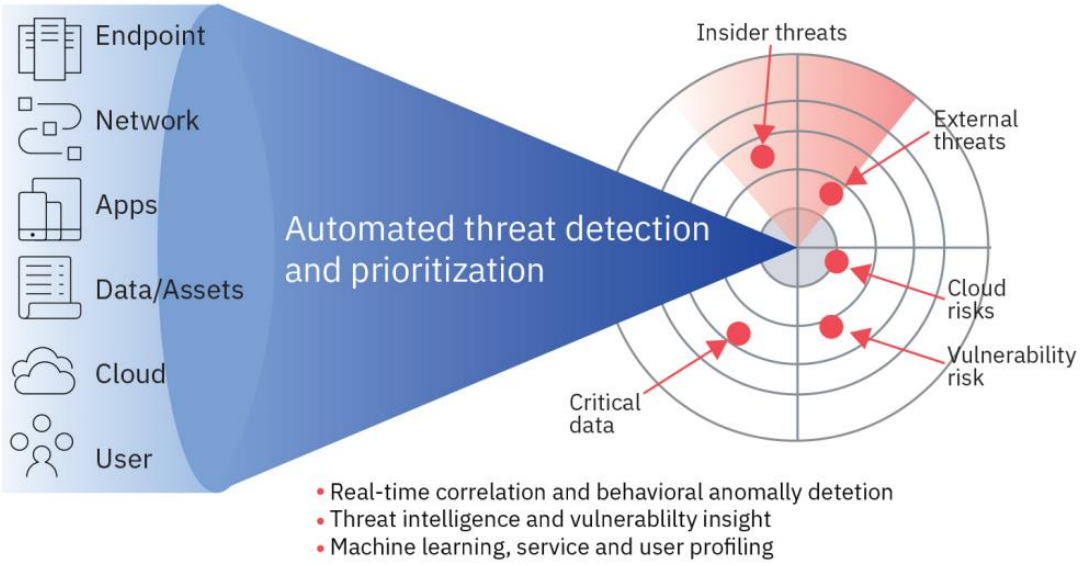
Підприємницькі мережі можуть охоплювати традиційну локальну IT-інфраструктуру, хмарні та операційні технології (OT), які вимагають певного нагляду для ефективного захисту активів, точного виявлення загроз та підтримання відповідності. Перш ніж команди безпеки можуть розпочати аналіз даних для виявлення та управління загрозами, вони повинні спочатку мати централізований доступ до різних даних безпеки. QRadar дозволяє організаціям отримувати централізовану, всебічну видимість заблокованих середовищ, збираючи, аналізуючи та нормалізуючи дані журналу та потоку.

QRadar SIEM призначений для автоматичного аналізу та кореляції активності в кількох джерелах даних, включаючи журнали, події, потоки мережі, активність користувачів, інформацію про вразливості та інформацію про загрози для виявлення відомих та невідомих загроз (рис 2.3).

QRadar SIEM інтелектуально корелює та аналізує різноманітну інформацію, включаючи такі дії:

- Події безпеки: із брандмауерів, віртуальних приватних мереж, системи виявлення вторгнень, системи запобігання вторгненню, бази даних тощо.
- Мережеві події: від комутаторів, маршрутизаторів, серверів, хостів тощо

- Контекст мережевої діяльності: контекст програми рівня 7 із мережевого та додаткового трафіку.
- Хмарна активність: від SaaS та IaaS середовищ, таких як Office365, SalesForce.com, Amazon Web Services (AWS), Azure та Google Cloud.
- Контекст користувача та ресурсу: контекстні дані з систем управління ідентифікацією та доступом, а також зі сканерів вразливостей.
- Події кінцевих точок: із журналу подій Windows, рішень Sysmon, EDR тощо.
- Журнали програм: від рішень планування корпоративних ресурсів (ERP), баз даних додатків, додатків SaaS тощо.



- Інформація про загрози: з джерел, таких як IBM X-Force [38].

Рисунок 2.3. QRadar SIEM збирає, аналізує та співвідносить дані з найрізноманітніших джерел для виявлення та визначення пріоритетів найбільш критичних загроз, що вимагають розслідування.

QRadar включає сотні попередньо побудованих випадків використання безпеки, алгоритми виявлення аномалій, правила та кореляційні політики в реальному часі для виявлення відомих і невідомих загроз. У міру виявлення загроз рішення вирішує події безпеки, пов’язані з ними, в єдині пріоритетні попередження, відомі як *оффенси* (offenses) [39]. Оффенси автоматично

визначаються пріоритетними як за ступенем тяжкості загрози, так і за критичністю залучених активів. В рамках кожного оффенса аналітики безпеки можуть побачити повний ланцюжок дій загрози з однієї консолі. Звідси аналітики можуть легко детально аналізувати конкретні події або потоки мережі, щоб розпочати розслідування, слід призначити оффенс конкретному аналітику (користувачу QRadar, що має необхідні повноваження). Оффенси автоматично оновлюються, коли виникає нова пов'язана діяльність, щоб аналітики могли бачити найсвіжішу інформацію в будь-який момент часу. Цей підхід допомагає аналітикам безпеки легко зрозуміти найбільш критичні загрози в навколишньому середовищі, надаючи наскрізне розуміння кожного потенційного інциденту, одночасно зменшуючи загальний обсяг попереджень.

QRadar має досвід та ресурси, необхідні для допомоги організаціям у вирішенні питань ризику та регуляторного впливу шляхом надання стандартних пакетів відповідності General Data Protection Regulation (GDPR), Federal Information Security Management Act (FISMA), Sarbanes-Oxley (SOX), HIPAA, ISO 27001, Payment Card Industry Data Security Standard (PCI DSS) та інші. Ці пакети безкоштовно включені в ліцензію QRadar SIEM і доступні в IBM Security App Exchange.

Спеціальні правила перевіряють події, потоки та порушення, щоб виявити незвичну активність у вашій мережі. Нові правила створюються за допомогою комбінацій *AND* та *OR* існуючих тестів правил. Правила виявлення аномалій перевіряють результати пошуку збереженого потоку або подій, щоб виявити, коли у мережі трапляються незвичні моделі трафіку. Вони також вимагають збереженого пошуку, згрупованого навколо загального параметра.

QRadar створює оффенси, коли події, потоки або те і інше відповідають критеріям тесту, визначеним у правилах.

QRadar аналізує таку інформацію:

- Вхідні події та потоки
- Інформація про активи

- Відомі вразливості
- Правило, яке створило правопорушення, визначає вид правопорушення.

Меджистрейт (magistrate) надає пріоритет оффенсу і призначає величину магнітуди на основі кількох факторів, включаючи кількість подій і шкалу Severity-Credibility-Relevance.

Event Collectors у QRadar збирають події з місцевих та віддалених джерел, нормалізують ці події та класифікують їх на категорії низького та високого рівня. Для потоків QRadar QFlow Collectors зчитують пакети, які передаються мережею, або отримують потоки від інших пристроїв, а потім перетворюють мережеві дані в записи потоків. Кожен Event Processor обробляє події або дані потоку з Event Collectors. Flow Processors перевіряють та співвідносять інформацію, щоб вказати на зміни поведінки або порушення політики. Custom Rule Engine (CRE) обробляє події та порівнює їх із визначеними правилами для пошуку аномалій. Коли умова правила виконується, Event Processor генерує дію, яка визначена у відповіді на правило. CRE відстежує системи, які беруть участь у інцидентах, вносить події в оффенси та генерує повідомлення.

Кожен із типів правил у QRadar перевіряється на вхідні дані з різних джерел у реальному часі. Існує кілька типів тестів правил. Деякі перевіряють наявність простих властивостей із набору даних. Інші тести правил є більш складними. Вони відстежують кілька послідовностей, подій, потоків та порушень протягом певного періоду та використовують лічильник, який знаходиться за одним або кількома параметрами, перш ніж спрацьовує відповідь на правило.

Event Rules, або правила подій

Перевіряють вхідні дані журналу, які обробляються в реальному часі обробником подій QRadar. Створюється правило події для виявлення однієї події або послідовностей подій. Наприклад, щоб відстежувати мережу на неуспішні спроби входу, доступ до декількох хостів або розвідувальну подію,

за якою слідує експлойт, адміністратор створює правило події. Зазвичай, правила подій створюють оффенс як відповідь.

Flow Rules, або правила потоку

Перевіряють вхідні дані потоку, які обробляються процесором потоку QRadar. Можна створити правило потоку для виявлення одного потоку або послідовностей потоку. Загальноприйнятим для правил потоку є створення оффенсів у відповідь.

Загальні правила

Перевіряють дані щодо подій та потоків. Наприклад, можна створити загальне правило для виявлення подій та потоків, які мають певну IP-адресу джерела. Загальноприйнятими є загальні правила, що створюють оффенс як відповідь.

Offence Rules

Перевіряють параметри оффенса, щоб викликати більше відповідей. Наприклад, відповідь створюється, коли порушення відбувається протягом певної дати та часу. Offence Rules обробляє порушення лише тоді, коли до нього внесені зміни. Наприклад, коли додаються нові події або система планує перегляд порушення для повторної його оцінки. Offence Rules зазвичай надсилають повідомлення як відповідь електронною поштою [40].

Більшість тестів правил оцінюють одну умову, наприклад, існування елемента в збірці посилальних даних або тестування значення щодо властивості події. Для складних порівнянь можна перевірити правила подій, побудувавши запит AQL (події і потоки у QRadar зберігаються у базі даних Ariel, у той час, як дані оффенсів та конфігурація системи зберігаються у базі даних PosrtgreSQL) з умовою *WHERE*. Можна використовувати всі функції умови *WHERE* для написання складних критеріїв пошуку, які можуть усунути необхідність у проведенні численних індивідуальних тестів. Наприклад, використовуючи умову AQL *WHERE*, можна перевірити, чи відстежується вхідний SSL або веб-трафік на наборі посилань.

Запуск тестів властивостей події, потоку або оффенсу, надає такі можливості, як визначення IP-адреса джерела, критичності події або аналізу швидкості мережевих потоків.

Функції надають можливість використовувати будівельні блоки (Building Blocks) та інші правила, щоб створити функцію, яка тестуватиме багато подій, багато потоків або багато оффенсів [41]. Також можна підключити правила, що використовують функції, які підтримують логічні оператори, такі як *OR* та *AND*. Наприклад, для підключення правила подій, можна використовувати умову *WHEN*, що виявити події, які відповідають наступній функції правил.

Будівельні блоки групують загальновживані тести для побудови складної логіки функцій, яку можна було використовувати в правилах.

Будівельні блоки використовують ті самі тести, що й правила, але не мають пов'язаних з ними дій. Вони часто налаштовані на тестування груп IP-адрес, привілейованих імен користувачів або колекцій імен подій. Наприклад, можна створити будівельний блок, який включає IP-адреси всіх поштових серверів або веб-серверів у мережі, а потім використовувати цей будівельний блок в іншому правилі, щоб виключити ці хости [42]. За замовчуванням будівельні блоки подаються як настанови, які можна переглянути та відредагувати відповідно до потреб мережі.

Налаштування будівельних блоків для визначення хостів (*BB: HostDefinition*), дозволяє QRadar виявляти та класифікувати більше серверів у мережі. Якщо певний сервер не виявляється автоматично, його можна додати до мережевих ассетів вручну відповідного будівельного блоку визначення хостів. Ця дія гарантує, що відповідні правила застосовуються до певного типу сервера. Окремі пристрої у BB можуть бути вручну замінені на діапазони IP-адрес типу CIDR, що відповідатиме заданій мережевій ієрархії (Network Hierarchy) [43].

Правила виявлення аномалій перевіряють результати пошуку збереженого потоку або подій, щоб виявити, коли у мережі трапляються

незвичні моделі трафіку. Правила виявлення аномалій вимагають збереженого пошуку, згрупованого навколо загального параметра, та увімкненого графіку часових рядів. Зазвичай пошук повинен накопичувати дані, перш ніж правило аномалії повертає будь-який результат, який визначає закономірності аномалій, порогових значень або змін поведінки.

Правила аномалії

Перевіряють трафік подій та потоків на наявність змін у короткочасних подіях, порівняно з більш тривалими часовими рамками. Наприклад, нові служби або програми, які з'являються в мережі, веб-сервер виходить з ладу, брандмауери, які всі починають блокувати трафік.

Порогові правила

Тестують події або потоки на активність, яка перевищує заданий діапазон або значно менше нього. Використовуються ці правила для виявлення змін використання смуги пропускання в додатках, призупинених служб, кількості користувачів, підключених до VPN, та виявлення великих вихідних передач трафіка.

Поведінкові правила

Тестують події або потоки на зміни обсягу, які відбуваються за звичайними шаблонами, щоб виявити відхилення. Наприклад, поштовий сервер, який раптово починає спілкується з багатьма хостами, або IPS (система захисту від вторгнень), яка починає генерувати численні дії попередження.

Поведінкові правила визначають норму, характерну для заданої властивості, за попередньо визначений період. Період визначає базовий графік порівняння оцінюваного параметра. Якщо період встановлений, наприклад, на 1 тиждень, поведінка властивості протягом цього тижня вивчається, а потім використовується у тесті правил, генеруючи попередження про будь-які суттєві зміни.

Після встановлення поведінкового правила період автоматично коригується. Коли дані періоду вивчаються, вони постійно оцінюються таким

чином, щоб зростання завантаженості трафіка відображалося протягом періоду. Більш тривалий час дії поведінкового правила значно підвищує його точність.

Висновки за розділом 2

Однією з головних цілей використання SIEM-систем є підвищення рівня інформаційної безпеки в наявній архітектурі за рахунок забезпечення можливості маніпулювати інформацією про безпечність та здійснювати попереджувальне управління інцидентами і подіями безпеки в близькому до реального часу режимі.

Основні можливості, які надаються SIEM-системами:

- Агрегація даних: управління журналами об'єднує дані з багатьох джерел, включаючи мережу, безпеку, сервери, бази даних, програми, забезпечуючи можливість консолідації даних, що контролюються, щоб уникнути пропуску важливих подій.
- Співвідношення: шукає загальні атрибути та пов’язує події разом у значущі пакети.
- Оповіщення: автоматизований аналіз корельованих подій
- Інформаційні панелі: інструменти можуть брати дані про події та перетворювати їх на інформаційні діаграми, щоб допомогти побачити шаблони або визначити діяльність, яка не формує стандартний шаблон.
- Відповідність: додатки можуть використовуватися для автоматизації збору даних про відповідність, виготовлення звітів, що адаптуються до існуючих процесів безпеки, управління та аудиту.
- Зберігання: використання довгострокового зберігання історичних даних для полегшення співвідношення даних з часом та забезпечення збереження, необхідного для вимог відповідності.
- Криміналістичний аналіз: можливість пошуку в журналах на різних вузлах та у різні періоди часу на основі конкретних критеріїв.

Для збору передачі необхідної інформації використовується агентний збір та безагентний збір інформації. Агентний спосіб передбачає використання спеціальної програми-агента, безагентний – налаштування джерела подій, такі як створення додаткових облікових записів, дозвіл віддаленого доступу і/або використання додаткових протоколів.

QRadar SIEM призначений для автоматичного аналізу та кореляції активності в кількох джерелах даних, включаючи журнали, події, потоки мережі, активність користувачів, інформацію про вразливості та інформацію про загрози для виявлення відомих та невідомих загроз

QRadar SIEM інтелектуально корелює та аналізує інформацію про:

• Події безпеки

• Мережеві події

• Контекст мережевої діяльності

• Хмарна активність

• Контекст користувача та ресурсу

• Події кінцевих точок

• Журнали програм

Спеціальні правила перевіряють події, потоки та порушення, щоб виявити незвичну активність у вашій мережі. Нові правила створюються за допомогою комбінацій *AND* та *OR* існуючих тестів правил. Правила виявлення аномалій перевіряють результати пошуку збереженого потоку або подій, щоб виявити, коли у мережі трапляються незвичні моделі трафіку [44]. Вони також вимагають збереженого пошуку, згрупованого навколо загального параметра.

QRadar створює оффенси, коли події, потоки або те і інше відповідають критеріям тесту, визначеним у правилах. Для їх створення QRadar аналізує таку інформацію:

- Вхідні події та потоки
- Інформація про активи
- Відомі вразливості

Кореляція подій відповідно до системних правил є важливою функцією системи QRadar. Загалом правила можна поділити та три види:

- Правила аномалії – перевіряють трафік подій та потоків на наявність змін у короткочасних подіях, порівняно з більш тривалими часовими рамками.
- Порогові правила – тестують події або потоки на активність, яка перевищує заданий діапазон або значно менше нього.
- Поведінкові правила - тестують події або потоки на зміни обсягу, які відбуваються за звичайними шаблонами, щоб виявити відхилення.

Таким чином визначено, що функціонал SIEM-системи IBM QRadar і особливості написання для нього правил кореляції є достатніми для створення на базі нього елементів системи захисту мережі від Cross-Site Scripting атак.

РОЗДІЛ 3
ПІДГОТОВКА СИТЕМИ ТА НАПИСАННЯ ПРАВИЛА КОРЕЛЯЦІЇ

3.1 Підготовка системи до коректного збору даних та аналізу потоків

Почати слід з правильної конфігурації надходження трафіка до маджистрейту IBM QRadar SIEM, а саме мережевих потоків. IBM QRadar фіксує трафік із дзеркальних портів у мережі за допомогою IBM QRadar QFlow Collector. Колектор QRadar QFlow Collector увімкнено за замовчуванням, тоді як дзеркальний порт підключений до інтерфейсу контролю QRadar. Типові місця розташування дзеркальних портів включають ядро, DMZ, сервер та комутатори.

QRadar QFlow Collector забезпечує для консолі QRadar видимість даних мережевих даних рівня 7 за моделлю OSI, тобто даних рівня додатків, та аналіз потоку мережевого трафіку незалежно від порту, на якому працює додаток. Наприклад, якщо протокол Internet Relay Chat (IRC) взаємодіє через порт 7500 (TCP), QRadar QFlow Collector ідентифікує трафік як IRC і забезпечує захоплення пакетів початку коммунікації.

Колектори QRadar QFlow не є механізмами повного захоплення пакетів, тому необхідно налаштувати кількість вмісту, який фіксується за потік. Розмір захоплення за замовчуванням – 64 байти. Однак, для реалізації завдання роботи необхідно відрегулювати це налаштування до максимального значення, яке скаладає 256 байт, щоб захопити більше вмісту на потік. Збільшення розміру захоплення збільшує мережевий трафік колектора QRadar QFlow, тож необхідно переконатися у наявності вільного місця на диску.

Наступним кроком, для більш глибоко аналізу даних, що надходять, у режимі реального часу, що є необхідним для виявлення аномалій, до основної

системи QRadar було встановлено додаток QRadar Network Insights (QNI). IBM QRadar Network Insights забезпечує поглиблений огляд мережевих комунікацій в режимі реального часу, щоб розширити можливості розгортання IBM QRadar. Завдяки глибокому аналізу мережевої активності та вмісту додатків, QRadar Network Insights надає можливість модулю QRadar Sense Analytics виявляти активність загроз, які інакше залишалася б непоміченою.

QRadar Network Insights забезпечує поглиблений аналіз як метаданих мережі, так і вмісту додатків для виявлення підозрілої активності, яка прихована серед звичайного трафіку, та вилучення вмісту, щоб забезпечити QRadar видимість діяльності мережевих загроз. Інтелектуальна інформація, яку надає QRadar Network Insights, легко інтегрується з традиційними джерелами даних та інформацією про загрози, щоб розширити можливості виявлення, аналізу та виявлення загроз QRadar.

QRadar Network Insights забезпечує видимість різноманітних випадків використання, включаючи:

- Виявлення та аналіз шкідливих програм
- Виявлення фішингу електронної пошти
- Інсайдерські загрози
- Виявлення бокового зміщення атаки
- Захист від ексфільтрації даних

Для коректної інсталяції додатку необхідно визначити правильну версію, відповідно до таблиці 3.1.1. Далі, з репозиторію додатків FixCentral слід завантажити визначений інсталяційний файл з розширенням *.iso* і змонтувати його у серверну директорію */media/cdrom*. Для того, щоб розпочати інсталяцію, слід запустити команду */media/cdrom/setup*. Після успішної інсталяції сервер буде перезавантажений.

Далі слід додати керований хост QRadar Network Insights до QRadar на вкладці *Адміністратор* у розділі *Керування системою та ліцензіями*. У списку *Дисплей* слід обрати пункт *Системи*, а у меню *Дії* обрати функцію *Додати*

хост. Для коректного налаштування параметрів керованого хоста, необхідно вказати фіксовану IP-адресу та кореневий пароль для доступу до оболонки операційної системи на пристрої. Застосування ліцензійного ключа проводиться на вкладці *Адміністратор, Керування системою та ліцензіями*. При наявності файла з ліцензійнім ключем на сервері, він відобразиться у списку. Після застосування ліцензії додаток QNI почне функціонування.

При первинному встановленні IBM QRadar Network Insights налаштовано на отримання максимум 64 байт необроблених даних корисного навантаження. Тому необхідно переналаштувати додаток QNI збільшивши розмір необробленого корисного навантаження до 32 768 байт, максимального значення корисного навантаження, яке не впливає на продуктивність. Якщо розмір максимального необробленого корисного навантаження QNI перевищує довжину захоплення вмісту QFlow, частина байтів корисного навантаження можуть бути скорочені. Тому необхідно переконатися, що QFlow збирає дані того самого розміру або більше, ніж розмір корисного навантаження QRadar Network Insights.

Таблиця 3.1.

Відповідність версій додатку QNI та основної системи QRadar

QRadar Network Insights appliances	**Appliance ID**
QRadar Network Insights 1901	6300
QRadar Network Insights 1910	6400
QRadar Network Insights 1920	6200
QRadar Network Insights 1940	6600
Software or virtual appliance installations	6500

Під час запуску міжсайтової атаки сценаріїв або тестування вразливості веб-сайту до неї зловмисник може спочатку видати простий тег форматування HTML, такий як <*b*> напівжирний, <*i*> курсив або <*u*> для підкреслення. Крім того, він може спробувати тривіальний тег сценарію, такий як <*script*> *alert*

("OK") </script>. Це, ймовірно, тому, що більшість друкованої та Інтернет-літератури про XSS використовують цей сценарій як приклад для визначення того, чи сайт вразливий до XSS. Ці спроби можна тривіально виявити. Однак досвідчений зловмисник може спробувати замаскувати весь рядок, ввівши його шістнадцяткові еквіваленти. Тож тег *<script>* буде виглядати як *%3C%73%63%72%69%70%74%3E*. З іншого боку, зловмисник може насправді використовувати проксі веб-застосунку і змінити автоматичне перетворення браузером спеціальних символів, таких як *<%3C>* до *%3E*. Отже, URL-адреса атаки буде містити кутові дужки замість їх шістнадцяткових еквівалентів, як це зазвичай трапляється. Наступний регулярний вираз перевіряє наявність атак, які можуть містити HTML-відкриваючі теги та закриваючі теги <> з будь-яким текстом усередині. Цей вираз буде ловити спроби використання *<b>* або *<u>* або *<script>*. Регулярний вираз враховує регістр. Також потрібно перевірити наявність дужок, а також їх шістнадцяткових еквівалентів, або *(%3C|<)*. Щоб виявити шістнадцяткове перетворення всього рядка, треба перевірити наявність чисел, а також знака % у даних, що передаються, іншими словами, використання [a-z0-9%]. Іноді це може спричинити помилкові спрацьовування, але більшу частину часу виявлятиме фактичну атаку.

Тож, було сформовано декілька шаблонів пейлоаду, відповідно до яких перевірятиметься вміст даних 7 рівня у потоках, які надходять у маджистрейт QRadar.

/((\%3C)|<)((\%2F)|\/)*[a-z0-9\%]+((\%3E)|>)/ix

- ((\%3C)|<) – перевіряє символ відкриття або шістнадцятковий еквівалент
- ((\%2F)|\/)* – коса риска для закриваючого тегу або його шістнадцяткового еквівалента
- [a-z0-9\%]+ – перевіряє буквено-цифровий рядок усередині тегу або шістнадцяткове представлення цих

- ((\%3E)|>) – перевіряє символ закриття або шістнадцятковий еквівалент

/((\%3C)|<)((\%69)|i|(\%49))((\%6D)|m|(\%4D))((\%67)|g|(\%47))[^\n]+((\%3E)|>)/I

- (\%3C)|<) – відкриваюча дужка або шістнадцятковий еквівалент
- (\%69)|i|(\%49))((\%6D)|m|(\%4D))((\%67)|g|(\%47) – літери *'img'* у різних комбінації ASCII, або шістнадцяткові еквіваленти верхнього чи нижнього регістру
- [^\n]+ – будь-який символ, крім нового рядка, що слідує за *<img*
- (\%3E)|>) – перевіряє символ закриття або шістнадцятковий еквівалент

/((\%3C)|<)[^\n]+((\%3E)|>)/I

Цей вираз просто шукає початковий HTML-тег та його шістнадцятковий еквівалент, за яким слід один або кілька символів, відмінних від нового рядка, а потім завершальний тег або його шістнадцятковий еквівалент. Це може призвести до кількох помилкових спрацьовувань залежно від структури веб-програми та веб-сервера, але гарантовано буде виявлено все, що навіть віддалено нагадує Cross-Site Scripting атаку.

3.2 Написання правил кореляції для IBM QRadar SIEM з метою виявлення XSS атак

Тож, для написання правила кореляцій маємо всі необхідні вхідні дані і відповідну конфігурацію системи.

Для початку, необхідно запустити Rule Wizard у панелі оффенсів системи QRadar. Відповідно до того, що для аналізу мережевого трафіка використовується додаток QNI, створюване правило кореляції має відноситись до типу *Flows* (рис. 3.1).

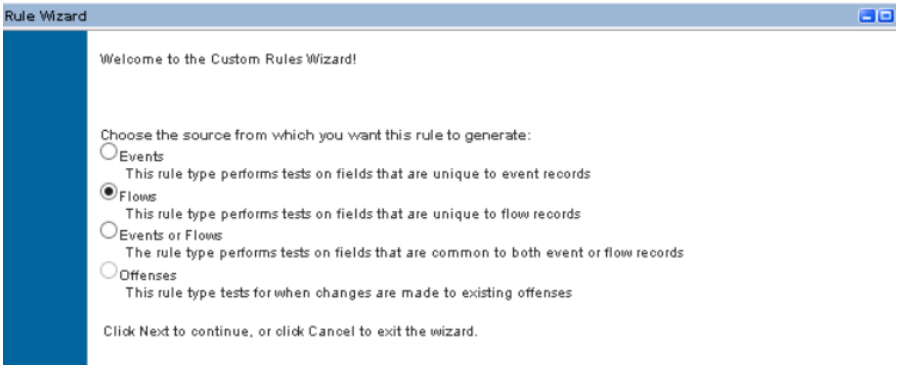


Рисунок 3.1. Конфігурація типу правила

У корпоративній мережі система IBM QRadar SIEM має розподілену архітектуру (рис. 3.2).

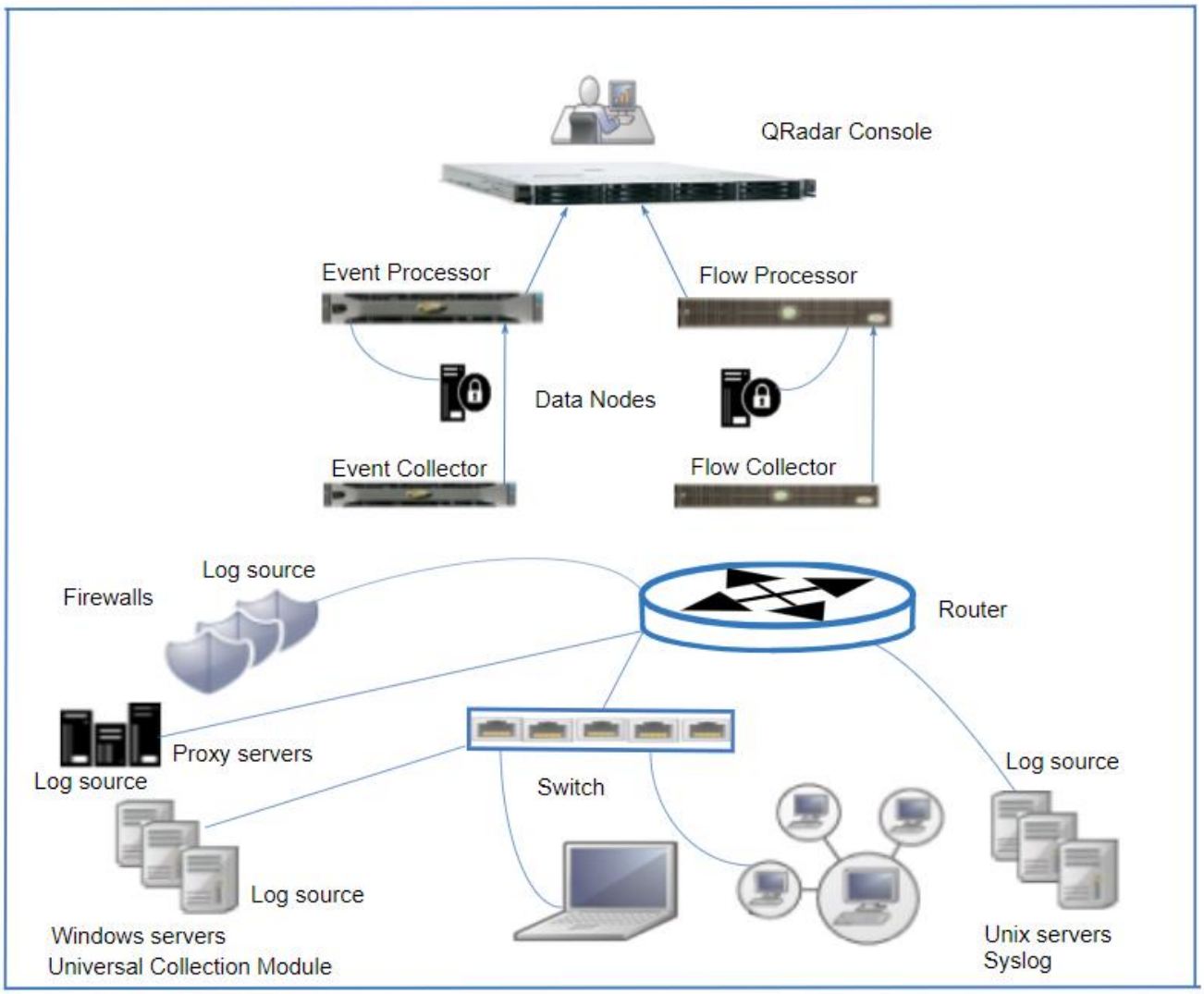


Рисунок 3.2. Взаємозв'язок архітектурних компонентів QRadar SIEM

Custom Rule Engine, або CRE (модуль, що виконує тестування за правилами), знаходиться у консолі, а також у кожному із колекторів даних, тому генерація нових оффенсів відбуватиметься у кожному вузлі окремо. Таким чином, доцільна конфігурація створюваного правила виду *Global* (рис. 3.3).

Рисунок 3.3. Конфігурація виду правила

Наступним кроком є вибір необхідних тестів. Слід пам’ятати, що порядок перевірки відповідає послідовності тестових правил у списку, при цьому, перехід до наступного правила відбудеться тільки у випадку, якщо тестування за попереднім правилом пройшло успішно. Тому першою умовою було визначено (рис. 3.4):

Рисунок 3.4. Тестування на відповідність шаблону регулярних виразів

У якості *regex* були використані шаблони пейлоаду 3.1.1, 3.1.2 та 3.1.3.

Наступне тестування перевіряє контекст трафіку, що надходить. Загалом, для QRadar SIEM існує чотири типи контексту: Local-to-Local (місцевий-до-місцевого), Local-to-Remote (місцевий-до-віддаленого), Remote-to-Local (віддалений-до-місцевого), Remote-to-Remote (віддалений-до-віддаленого). Контекст і напрямок потоків визначається автоматично відповідно до логіки мережевої ієрархії, тому тип Remote-to-Remote свідчить про недоліки конфігурації QRadar. У випадку XSS підозрілий трафік має надходити від віддаленої системи до адреси, що належить внутрішньому діапазону, визначеному у мережевій ієрархії. Тому другим тестом визначена така умова (рис. 3.5):

Рисунок 3.5. Тестування на відповідність контексту і напрямку потоку

Останнім правилом буде тестування IP-адреси: якщо задіюється адреса, яка належить, наприклад, веб-серверу, тест поверне позитивний результат і правило спрацює (рис. 3.6):

Рисунок 3.6. Тестування на відповідність IP-адреси

Наступним кроком слід віднести правило до однієї або декількох категорій загроз (рис. 3.7)

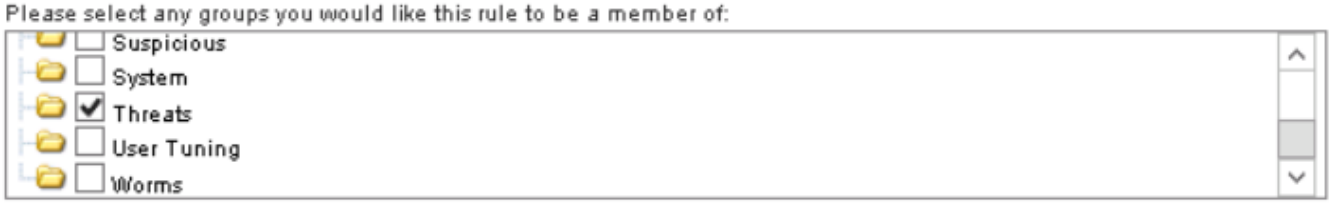


Рисунок 3.7. Віднесення правила до категорії

Магнітуда оффеса – це показник важливості оффенса у системі. IBM QRadar використовує величину магнітуди, щоб визначити пріоритети оффенса та допомогти аналітику визначити, які оффенси слід розслідувати першими.

Рівень магнітуди обчислюється на основі шкали Severity-Credebility-Relevance.

- Relevance (Відповідність) визначає вплив порушення на мережу. Наприклад, якщо порт відкритий, актуальність велика.
- Severity (Достовірність) вказує на цілісність порушення, що визначається рейтингом довіри, встановленим у джерелі журналу. Довіра зростає, коли кілька джерел повідомляють про одну і ту ж подію.
- Credebility (Серйозність) вказує на рівень загрози, яку створює джерело стосовно того, наскільки підготовлений пункт призначення до нападу.

Створюванному правилу були присвоєні значення за шкалою Severity-Credebility-Relevance, а також необхідні дії у випадку спрацювання правила (рис. 3.8).

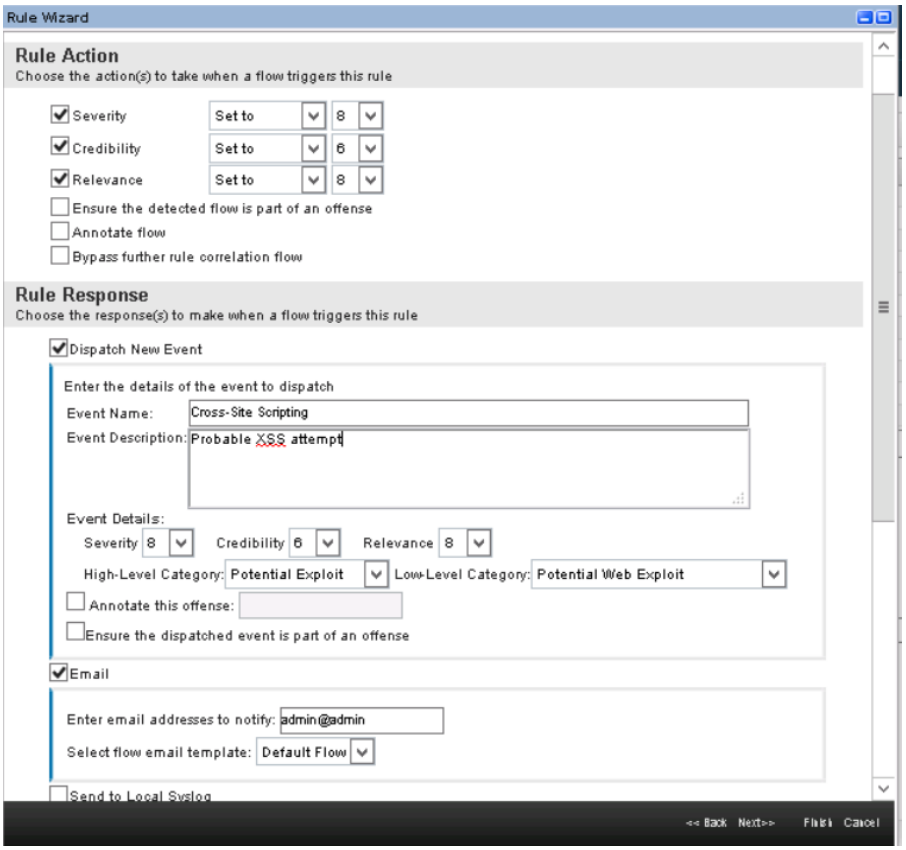


Рисунок 3.8. Визначення магнітуди оффенсу для створюваного правила

Таким чином було створене правило кореляції (рис. 3.9.), яке виявлятиме потенційну XSS активність у мережі (рис. 3.10).

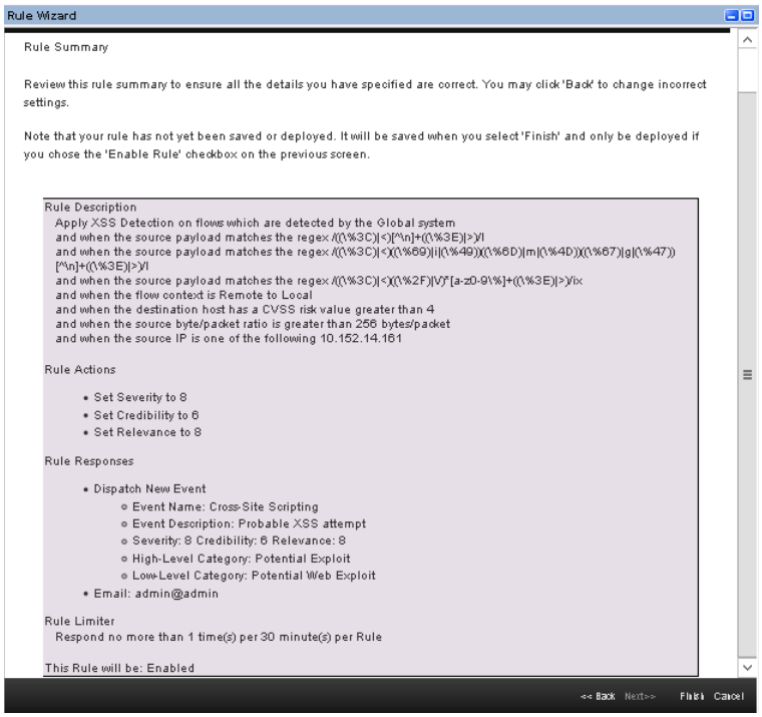


Рисунок 3.9. Підсумкові параметри створеного правила кореляції

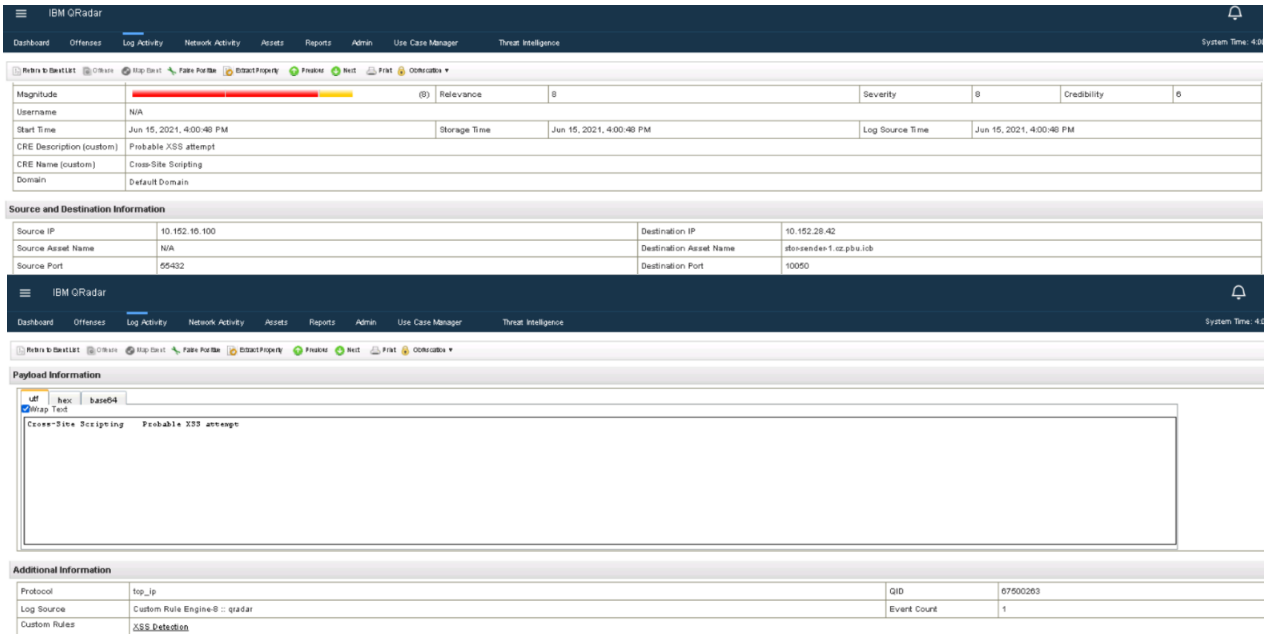


Рисунок 3.10. Інформація про оффенс у консолі QRadar SIEM, що згенерувався в результаті задіяння створеного правила.

Висновки за розділом 3

У розділі були використанні можливості IBM QRadar SIEM для формування елементів системи захисту для своєчасної ідентифікації та протидії активності, що може бути пов’язана з XSS-атаками.

Для реалізації завдання роботи був проаналізований функціонал QFlow. Його параметри були відрегульовані таким чином, щоб отримувати максимальну кількість байтів для аналізу із мережевих потоків, яке складає 256 байт.

Був встановлений і сконфігурований відповідно до поставленого завдання додатковий модуль QRadar Network Insights, який забезпечує поглиблений огляд мережевих комунікацій в режимі реального часу, щоб розширити можливості розгортання IBM QRadar. Завдяки глибокому аналізу мережевої активності та вмісту додатків, QRadar Network Insights надає можливість модулю QRadar Sense Analytics виявляти активність загроз, які інакше залишалася б непоміченою.

Було сформовано декілька шаблонів пейлоаду, відповідно до яких перевірятиметься вміст даних 7 рівня у потоках, які надходять у маджистрейт QRadar.

За допомогою Rule Wizard було написано правило кореляції для виявлення активності, яка може бути безпосередньо пов’язана із атаками міжсайтового скриптингу. Створеному правилу були присвоєні значення за шкалою Severity-Credebility-Relevance, а також необхідні дії у випадку спрацювання правила.

ВИСНОВКИ

Міжсайтовий скриптинг (XSS) – це атака націлена на ін’єкцію коду, що дозволяє зловмисникові виконати шкідливий JavaScript в браузері іншого користувача.

XSS-атаки часто поділяються на три типи:

- Збережені XSS, де шкідливий рядок формується з використанням бази даних веб-сайту.
- Відображені XSS, де шкідливий рядок формується із запиту жертви.
- DOM-моделі XSS, де уразливість виникає в коді на стороні клієнта, а не на боці серверного коду.

Однією з головних цілей використання SIEM-систем є підвищення рівня інформаційної безпеки в наявній архітектурі за рахунок забезпечення можливості маніпулювати інформацією про безпечність та здійснювати попереджувальне управління інцидентами і подіями безпеки в близькому до реального часу режимі.

Основні можливості, які надаються SIEM-системами:

- Агрегація даних: управління журналами об'єднує дані з багатьох джерел, включаючи мережу, безпеку, сервери, бази даних, програми, забезпечуючи можливість консолідації даних, що контролюються, щоб уникнути пропуску важливих подій.
- Співвідношення: шукає загальні атрибути та пов’язує події разом у значущі пакети.
- Оповіщення: автоматизований аналіз корельованих подій
- Інформаційні панелі: інструменти можуть брати дані про події та перетворювати їх на інформаційні діаграми, щоб допомогти побачити шаблони або визначити діяльність, яка не формує стандартний шаблон.

- Відповідність: додатки можуть використовуватися для автоматизації збору даних про відповідність, виготовлення звітів, що адаптуються до існуючих процесів безпеки, управління та аудиту.
- Зберігання: використання довгострокового зберігання історичних даних для полегшення співвідношення даних з часом та забезпечення збереження, необхідного для вимог відповідності.
- Криміналістичний аналіз: можливість пошуку в журналах на різних вузлах та у різні періоди часу на основі конкретних критеріїв.

Спеціальні правила перевіряють події, потоки та порушення, щоб виявити незвичну активність у вашій мережі. Нові правила створюються за допомогою комбінацій *AND* та *OR* існуючих тестів правил. Правила виявлення аномалій перевіряють результати пошуку збереженого потоку або подій, щоб виявити, коли у мережі трапляються незвичні моделі трафіку. Вони також вимагають збереженого пошуку, згрупованого навколо загального параметра.

QRadar створює оффенси, коли події, потоки або те і інше відповідають критеріям тесту, визначеним у правилах.

Для реалізації завдання роботи був проаналізований функціонал QFlow.

Був встановлений і сконфігурований відповідно до поставленого завдання додатковий модуль QRadar Network Insights, який забезпечує поглиблений огляд мережевих комунікацій в режимі реального часу, щоб розширити можливості розгортання IBM QRadar

Було сформовано декілька шаблонів пейлоаду, відповідно до яких перевірятиметься вміст даних 7 рівня у потоках, які надходять у маджистрейт QRadar.

За допомогою Rule Wizard було написано правило кореляції для виявлення активності, яка може бути безпосередньо пов’язана із атаками міжсайтового скриптингу. Створеному правилу були присвоєні значення за шкалою Severity-Credibility-Relevance, а також необхідні дії у випадку спрацювання правила.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 OWASP. OWASP Top Ten Web Application Security Risks [Електронний ресурс] / OWASP – Режим доступу до ресурсу: https://owasp.org/www-project-top-ten/

2 A7:2017-Cross-Site Scripting (XSS) [Електронний ресурс] – Режим доступу до ресурсу: https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_(XSS)

3 Хранимые, отображаемые и DOM-based XSS: выявление и блокирование [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/pentestit/blog/535642/

4 XSpider XSS Detection Tool [Електронний ресурс] – Режим доступу до ресурсу: https://www.ptsecurity.com/ru-ru/products/xspider/

5 Nemesida Scanner [Електронний ресурс] – Режим доступу до ресурсу: https://www.pentestit.ru/nemesida-scanner/

6 Acunetix Vulnerability Scanner [Електронний ресурс] – Режим доступу до ресурсу: https://www.acunetix.com/vulnerability-scanner/

7 Hydara I. Current state of research on cross-site scripting (XSS) – A systematic literature review / I. Hydara, H. Zulzalil, N. Admodisastro. // Information and Software Technology. – 2014

8 Hidhaya F. Intrusion Protection against SQL Injection and Cross Site Scripting Attacks Using a Reverse Proxy / F. Hidhaya, A. Geetha. // Recent Trends in Computer Networks and Distributed Systems Security. – 2012. – С. 252–263

9 Secure Code Generation for Web Applications / M.Johns, C. Bayerlein, R. Giesecke, J. Posegga. // Engineering Secure Software and Systems. – 2010. – С. 96–113

10 Kirda E. Client-side cross-site scripting protection / E. Kirda, N. Jovanovic, G. Vigna. // Computers & Security. – 2009. – №7. – С. 592–604

11 Li N. Perturbation-based user-input-validation testing of web applications / N. Li, T. Xie, C. Liu. // Journal of Systems and Software. – 2010. – №11. – С. 2263–2274

12 Kirda E. Leveraging User Interactions for In-Depth Testing of Web Applications / E. Kirda, C. Krueger, S. McAllister. // Recent Advances in Intrusion Detection. – 2008. – С. 191–210

13 Mui R. Preventing Web Application Injections with Complementary Character Coding / R. Mui, P. Franckl. // Computer Security – ESORICS 2011. – 2011. – С. 80–99

14 Automatically Hardening Web Applications Using Precise Tainting / S.Guarnieri, D. Green, J. Shirley, D. Evans. // Security and Privacy in the Age of Ubiquitous Computing. – 2015. – С. 295–307.

15 Shar. Auditing the XSS defence features implemented in web application programs / Shar, Tan. // IET Software. – 2012. – С. 377 – 390

16 Shar. Automated removal of cross site scripting vulnerabilities in web applications / Shar, Tan. // Information and Software Technology. – 2012. – С. 467–678

17 Sharma P. Integrated approach to prevent SQL injection attack and reflected cross site scripting attack / P. Sharma, R. Johari. // International Journal of System Assurance Engineering and Management. – 2012. – №3. – С. 343–351

18 Sivakumar. Constructing a “Common Cross Site Scripting Vulnerabilities Enumeration (CXE)” Using CWE and CVE / Sivakumar, Garg. // Information Systems Security. – 2017. – С. 277–291

19 Sun F. Client-Side Detection of XSS Worms by Monitoring Payload Propagation / F. Sun, L. Xu, Z. Su. // Computer Security – ESORICS 2019. – 2019. – С. 538–554

20 Sundareswaran S. XSS-Dec: A Hybrid Solution to Mitigate Cross-Site Scripting Attacks / S. Sundareswaran, A. Squicciarini. // Data and Applications Security and Privacy XXVI. – 2012. – С. 223–238

21 Van Gundy M. Noncespaces: Using randomization to defeat cross-site scripting attacks / M. Van Gundy, H. Chen. // Computers & Security. – 2012. – №4. – С. 612–628.

22 Bisht P. WebAppArmor: A Framework for Robust Prevention of Attacks on Web Applications / P. Bisht, M. Zhou, K. Gondi. // Information Systems Security. – 2010. – С. 3–26

23 Zimmer D. Real World XSS [Електронний ресурс] / David Zimmer. – 2008. – Режим доступу до ресурсу: http://www.xssed.com/article/21/Paper_Real_World_XSS

24 Cross-Site Scripting [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Cross-Site_Scripting

25 Li C. Cross-Site Scripting Guardian: A Static XSS Detector Based on Data Stream Input-Output Association Mining / C. Li, Y. Wang, C. Miao. // Applied Science. – 2020

26 Ahmed M. EVADING ALL WEB-APPLICATION FIREWALLS XSS FILTERS / Mazin Ahmed. – 2015

27 SIEM [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/SIEM

28 https://en.wikipedia.org/wiki/Security_information_and_event_manage ment [Електронний ресурс] – Режим доступу до ресурсу: Security information and event management

29 SIEM: ответы на часто задаваемые вопросы [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/172389

30 Обзор решений SIEM (Security information and event management) [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/company/roi4cio/blog/528770/

31 2019 SIEM TRENDS AND USAGE [Електронний ресурс] – Режим доступу до ресурсу: https://cybersecurity.att.com/resource-center/infographics/siem-trends-and-usage

32 2019 SIEM Survey Report [Електронний ресурс] – Режим доступу до ресурсу: https://cybersecurity.att.com/resource-center/analyst-reports/siem-survey-report

33 What is a SIEM and what are the benefits for business? [Електронний ресурс] – Режим доступу до ресурсу: https://cybersecurity.att.com/blogs/security-essentials/siem-what-is-it-and-why-does-your-business-need-it

34 What is a Security Operations Center (SOC)? [Електронний ресурс] – Режим доступу до ресурсу: https://www.microfocus.com/en-us/what-is/security-operations-center

35 Королев И. Анализ проблематики системы управления информацией и событиями безопасности в информационных системах / И. Королев, В. Попов, В. Ларионов. // Инновации в науке. – 2018. – №12. – С. 19–26

36 SIEM: What Is SIEM, How It Works, and Useful Resources [Електронний ресурс] – Режим доступу до ресурсу: https://logsentinel.com/blog/siem-what-is-siem-how-it-works-and-useful-resources

37 IBM QRadar SIEM [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/ru-ru/products/qradar-siem

38 Custom Rules [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qradar-on-cloud?topic=rules-custom

39 QRadar Rules anf Offenses [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qsip/7.4?topic=phase-qradar-rules-offense

40 Rules [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qradar-on-cloud?topic=siem-rules

41 Tunning Building Blocks [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qsip/7.4?topic=blocks-tuning-uilding#t_tuning_guide_tuning_building_blocks

42 IBM® QRadar building blocks [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qsip/7.4?topic=phase-qradar-building-blocks

43 Editing building blocks [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qradar-on-cloud?topic=rules-editing-building-blocks

44 Anomaly detection rules [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibm.com/docs/en/qradar-on-cloud?topic=rules-anomaly-detection