УДК 004.6
DOI: 10.31673/2409-7292.2025.014076

Коваль О. Р., Гарасимчук О. І.

ОРГАНІЗАЦІЯ ЗАХИСТУ ДАНИХ ПІД ЧАС ЇХ ОБРОБКИ У ДОДАТКУ APACHE SPARK

Розглянуті проблеми, пов'язані з безпекою даних в Apache Spark. Праця зосереджена на таких ключових аспектах, як управління доступом, захист конфіденційної інформації та запобігання атакам на рівні обробки даних. Досліджено, що однією з головних загроз є витік даних через неправильну конфігурацію доступу до кластерів або неавторизоване виконання завдань. Крім того, небезпеку становлять атаки на рівні серіалізації даних, що можуть використовувати вразливості механізмів передачі даних між вузлами. Важливо також враховувати можливі загрози, пов'язані з використанням сторонніх бібліотек, які можуть містити шкідливий код або мати відомі вразливості. Врахування цих проблем допоможе користувачам Apache Spark підвищити рівень безпеки своїх обчислювальних середовищ і мінімізувати ризики витоку даних. Захищені механізми аутентифікації та авторизації, а також шифрування переданих даних дозволяють значно зменшити ймовірність несанкціонованого доступу. Додатково, застосування політик безпеки на рівні конфігурації кластерів та ізоляції середовищ виконання дозволяє уникнути впливу потенційно шкідливих процесів. Також важливо здійснювати регулярний моніторинг та аудит активності в системі, що дозволяє своєчасно виявляти та реагувати на підозрілі дії. На основі найпоширеніших проблем, з якими стикаються компанії та користувачі Apache Spark, було проаналізовано основні загрози, що впливають на безпеку даних. У ході дослідження розглядалися такі відомі вразливості, як CVE-2023-22946, CVE-2022-31777, CVE-2022-33891, CVE-2021-38296 та CVE-2020-9480. Кожна з цих вразливостей могла призвести до витоку даних, несанкціонованого виконання коду або інших загроз для цілісності та конфіденційності інформації. Аналіз показав, що як правило ключові проблеми пов'язані з неправильним керуванням доступом, недостатньою перевіркою вхідних даних та вразливостями у механізмах обробки запитів. З урахуванням цих загроз були розроблені рекомендації щодо їх усунення та мінімізації ризиків. Використання актуальних механізмів аутентифікації та авторизації, регулярне оновлення програмного забезпечення, а також ізоляція робочих середовищ дозволяють значно зменшити ймовірність експлуатації відомих вразливостей. Крім того, моніторинг системних журналів і аналіз поведінки запитів допомагає виявляти підозрілі дії та оперативно реагувати на потенційні атаки.

Ключові слова: Hadoop, Apache Spark, HDFS, RDD, Spàrk кластер. AES, TLS/SSL, безпека даних, логи, аутентифікація, керування доступом.

Вступ

У сучасному світі інформаційні системи зосереджені на обробці величезних обсягів даних, що постійно генеруються. Тому для досягнення значущих результатів у промисловості, бізнесі чи науці надзвичайно важливо вміти ефективно аналізувати ці дані, використовуючи сучасні інструменти та технології [1-2]. Оптимальна робота з великими даними дозволяє отримувати цінні інсайти, автоматизувати процеси та приймати обґрунтовані рішення.

Apache Spark [3-4] є високопродуктивною платформою для обробки великих даних, що забезпечує ефективні розподілені обчислення. Це платформа з відкритим кодом для обробки великих даних, що робить його ключовим інструментом для розробників і фахівців у цій сфері. Spark може працювати як на окремому ноутбуці, так і на великих кластерах із тисячами серверів, що забезпечує його високу гнучкість і масштабованість. Завдяки цьому він стає універсальним рішенням для компаній і дослідників, які прагнуть швидко аналізувати великі обсяги даних.

Apache Spark має низку переваг, які зробили його одним із найпопулярніших проєктів в екосистемі Hadoop. Його широкі можливості дозволяють ефективно обробляти великі обсяги даних, забезпечуючи високу швидкість, зручність для розробників і підтримку різних типів завдань. Серед основних переваг можна виділити:

– *Висока швидкодія*. Spark забезпечує швидке виконання аналітичних запитів незалежно від обсягу даних, використовуючи кешування в пам'яті та оптимізовані алгоритми обробки, що значно прискорює виконання операцій порівняно з традиційними підходами.

– *Зручність для розробників***.** Spark підтримує кілька популярних мов програмування, зокрема Java, Scala, R і Python, а також включає бібліотеки для широкого спектра завдань – від

виконання SQL-запитів до потокової обробки даних і машинного навчання, що дає змогу розробникам працювати у знайомому середовищі. Високорівневі API спрощують розподілену обробку, дозволяючи писати ефективний код із мінімальними зусиллями.

– *Гнучка підтримка різних навантажень*. Apache Spark підходить для широкого спектра завдань, включаючи інтерактивні запити, потокову аналітику, машинне навчання та обробку графів. Завдяки цьому одна програма може поєднувати різні робочі навантаження, що спрощує розгортання складних аналітичних рішень. Це робить Spark універсальним інструментом для аналізу даних, який можна адаптувати під конкретні потреби бізнесу чи наукових досліджень.

Аналіз літературних джерел та формулювання проблеми

Безпека Apache Spark охоплює набір методів, інструментів і передових практик, спрямованих на захист програм і даних від несанкціонованого доступу, втрати чи пошкодження. Вона включає механізми захисту даних як під час зберігання, так і при передачі, а також забезпечує ефективну автентифікацію та авторизацію користувачів. Крім того, безпека Spark гарантує цілісність процесів обробки та зберігання даних, створюючи надійне середовище для роботи з великими обсягами інформації. Але незважаючи на це його використання супроводжується низкою ризиків [5-6], серед яких уразливості, пов’язані з автентифікацією, виконанням коду та підвищенням привілеїв. Аналіз безпеки платформи виявив потенційні загрози, які можуть бути використані зловмисниками для несанкціонованого доступу до даних та обчислювальних ресурсів.

Враховуючи зростаючий обсяг і критичність даних, важливо забезпечити їхню конфіденційність, цілісність і доступність під час обробки в Spark-додатках

Apache Spark широко використовується для обробки великих обсягів даних, що робить безпеку критично важливою для запобігання несанкціонованому доступу та забезпечення цілісності даних. У середовищах, де обробляється конфіденційна інформація, порушення безпеки може призвести до втрати даних, порушення приватності та юридичних наслідків.

Spark часто працює у розподілених обчислювальних середовищах, що робить його вразливим до різноманітних векторів атак, які необхідно враховувати для захисту даних та інфраструктури. Захищене середовище Spark мінімізує ризик простоїв, спричинених зловмисними діями або витоками даних. Така стабільність є необхідною для організацій, які використовують Spark для критично важливих операцій, де будь-яке порушення може призвести до фінансових втрат і шкоди репутації.

Для підвищення рівня захищеності розробниками запропоновано ряд рішень, серед яких оновлення до останніх версій Apache Spark, обмеження використання proxy-user, посилене налаштування безпекових конфігурацій та впровадження сучасних методів шифрування. Крім того, запропоновано контроль мережевого доступу та регулярний аудит безпекових налаштувань для запобігання можливим атакам.

Застосування машинного навчання у поєднанні зі Spark Streaming та MapReduce сприяє автоматизованому виявленню аномалій, що значно покращує загальний рівень безпеки корпоративних систем.

Також варто зазначити, що для забезпечення безпеки даних у Apache Spark існує декілька широко використовуваних підходів. Одним із найпопулярніших є аутентифікація та авторизація за допомогою Kerberos або LDAP, що забезпечує захист доступу до кластерів[7]. Шифрування даних у стані спокою та під час передавання реалізується за допомогою TLS/SSL та механізмів Transparent Data Encryption (TDE)[8]. Для моніторингу активності та виявлення загроз використовуються журнали подій і системи виявлення вторгнень (IDS), такі як Apache Ranger або AWS GuardDuty[9-10].

Крім того, існують засоби для запобігання витоку даних, такі як Data Loss Prevention (DLP)[11], які допомагають виявляти та блокувати передавання конфіденційної інформації за

межі організації. Також Apache Spark має широке налаштування з безпекою, але на жаль дуже мало потрібних функцій по замовчуванню вимкнені [12].

Окрім вищезгаданих методів, цікавою альтернативою є використання кастомних джерел і приймачів даних у Spark, що дозволяє застосовувати спеціалізовані підходи до безпеки [13]. Нестандартні джерела та приймачі даних у Spark можуть бути використані для підвищення рівня безпеки та виявлення потенційних загроз у кібербезпеці.

Метою роботи є виявлення найбільш поширених проблем безпеки при обробці даних у Apache Spark, аналіз існуючих методів захисту, їх переваг та недоліків і як наслідок знаходження найкращих рішень для забезпечення безпеки даних під час їхньої обробки. Аналізуючи сучасні технології та підходи, необхідно визначити найбільш ефективні стратегії захисту, які можна адаптувати до специфічних вимог Apache Spark. Також необхідно розробити підхід до захисту даних, який повинен поєднувати перевірені методи безпеки та гнучкі механізми контролю доступу для ефективного захисту інформації в умовах масштабованих розрахунків. Це б дозволило створити комплексний підхід, що допоможе мінімізувати потенційні ризики витоку конфіденційної інформації, покращити існуючий рівень захисту даних у Apache Spark та забезпечити надійний рівень захисту інформації в розподілених середовищах обробки великих даних.

Результати дослідження

Шифрування даних при підключенні картки як знімного носія

Spark складається з кількох основних компонентів, які забезпечують його функціональність. Це Spark Core, який відповідає за основні операції обробки даних, Spark SQL для обробки структурованих даних і взаємодії з базами даних, Spark Streaming для обробки потокових даних, Spark MLlib для машинного навчання, Spark GraphX для обробки графів і SparkR, що дозволяє використовувати Spark з мовою програмування R. Ці компоненти разом створюють потужну екосистему для обробки великих даних [14].

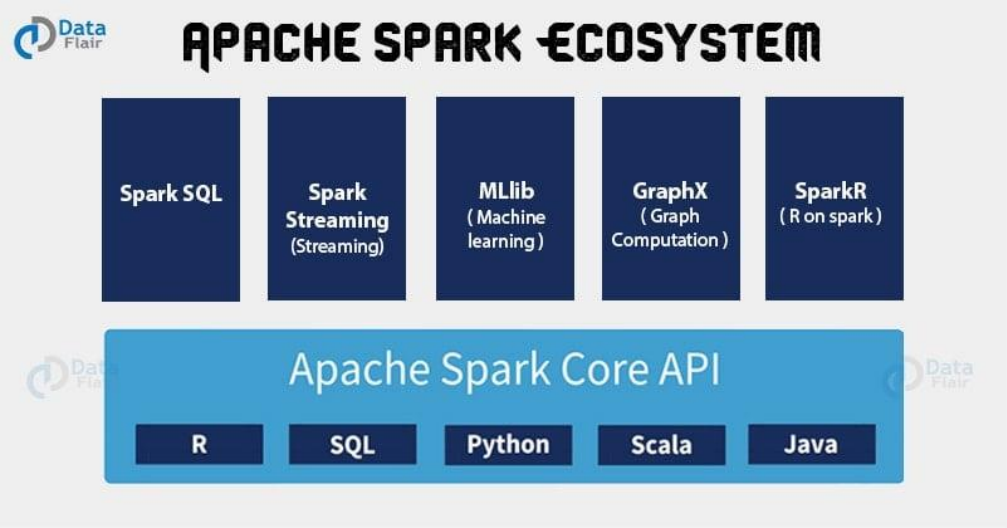

Рис. 1. Екосистема Apache Spark та її компоненти

Однією з ключових особливостей Apache Spark є Resilient Distributed Dataset (RDD) — розподілена абстракція даних, яка дозволяє ефективно обробляти великі обсяги інформації, використовуючи переваги розподілених обчислень [15].

Apache Spark є частиною екосистеми Hadoop і забезпечує тісну інтеграцію з її інструментами. Завдяки можливості виконання обчислень у пам’яті Spark значно перевершує Hadoop за швидкістю обробки даних, що робить його привабливим вибором для аналітики великих даних. Крім того, Spark надає широкий набір інструментів для машинного навчання,

© Коваль О. Р., Гарасимчук О. І. (2025). Організація захисту даних під час їх обробки у додатку Apache Spark. Сучасний захист інформації, 1(61), 197–205.
https://doi.org/10.31673/2409-7292.2025.014076.

потокової обробки, SQL-запитів та аналізу графів, що робить його універсальним рішенням для аналізу даних у кластерних середовищах

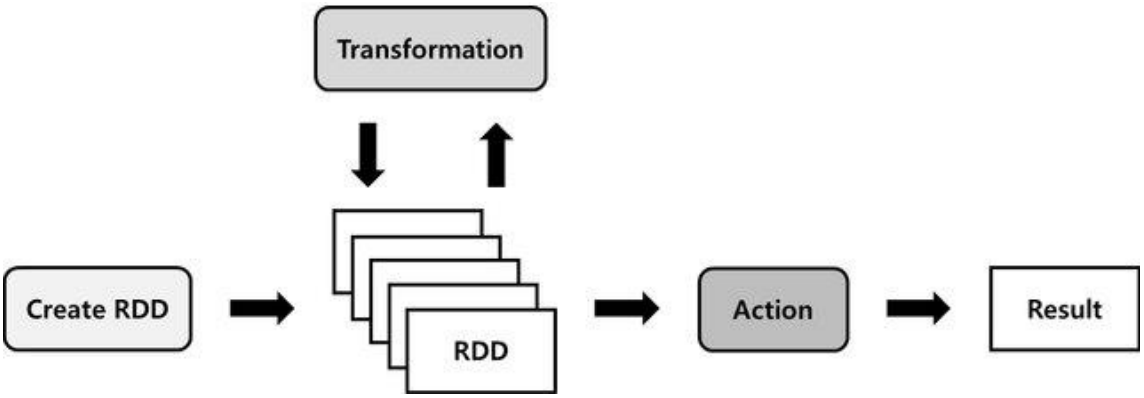

Рис. 2. Модель відмовостійких розподілених наборів даних (RDD) на Apache Spark

Оскільки Apache Spark обробляє величезні обсяги даних, часто виявляються різноманітні уразливості, які можуть бути використані зловмисниками. Можна виділити наступні *приклади критичних уразливостей*, що демонструють основні ризики безпеки, з якими стикається ця платформа [16]:

1. CVE-2023-22946. Дана уразливість була виявлена у версіях Apache Spark до 3.4.0 та класифікована як середнього рівня небезпеки. Вона дозволяє підвищення привілеїв шляхом використання шкідливого конфігураційного класу у функціоналі proxy-user під час виконання команд через spark-submit. Зловмисники можуть скористатися цією уразливістю, включивши шкідливі класи в classpath, що дає їм змогу виконувати код із привілеями користувача, який подає запит. Особливо ризикованою ця уразливість є для середовищ, які покладаються на конфігурацію proxy-user, наприклад, при використанні Apache Livy для керування додатками.

Можна виділити наступні способи усунення та запобігання даної уразливості:

а). Оновлення до останньої версії Apache Spark. Найефективнішим методом захисту є оновлення Apache Spark до версії 3.4.0 або новішої, оскільки ця уразливість була усунена у даному випуску. Оновлення забезпечить:

– Впровадження додаткових перевірок безпеки при використанні механізму proxy-user.

– Загальне покращення безпеки та стабільності Spark-кластера.

– Закриття можливості використання шкідливих конфігурацій класів.

б). Обмеження використання proxy-user. Функціональність proxy-user дозволяє одному користувачеві виконувати завдання від імені іншого, що створює потенційну загрозу підвищення привілеїв. Для мінімізації ризиків необхідно:

– Вимкнути цю опцію, якщо вона не є критично необхідною для вашого середовища.

– Використовувати чітку політику дозволених та заборонених користувачів.

– Переконатися, що тільки довірені облікові записи можуть користуватися proxy-user.

– Жорстке обмеження Classpath.

Одним із механізмів атаки є внесення зловмисних класів у classpath, що дозволяє виконання коду з підвищеними привілеями. Щоб запобігти цьому, необхідно переконатися, що spark.submit.proxyUser.allowCustomClasspathInClusterMode залишається у значенні false. Це запобіжить користувачам можливість додавати довільні класи до classpath при виконанні запитів.

2. CVE-2022-31777 – це вразливість типу зберігання кроссайтного сценарію (XSS), яка була виявлена в Apache Spark версій 3.2.1 та попередніх, включаючи 3.3.0. Вона дозволяє зловмисникам вставляти шкідливі JavaScript-скрипти в журнали, що потім відображаються в інтерфейсі користувача Spark. Користувачі, які переглядають ці логи через веб-інтерфейс,

© Коваль О. Р., Гарасимчук О. І. (2025). Організація захисту даних під час їх обробки у додатку Apache Spark. Сучасний захист інформації, 1(61), 197–205.
https://doi.org/10.31673/2409-7292.2025.014076.

можуть ненавмисно виконати небезпечний код на своєму браузері. Це може призвести до крадіжки сеансів користувачів, отримання конфіденційних даних, таких як куки або дані авторизації, а також до потенційного втручання в роботу системи. Вразливість створює серйозну загрозу безпеці користувачів і організацій, що використовують Apache Spark для обробки даних.

Можна виділити наступні способи усунення та запобігання даної уразливості:

а). Оновлення Apache Spark до версії 3.2.2 або новішої (зокрема 3.3.1 для гілки 3.3.x) дозволяє усунути вразливість шляхом очищення виводу журналів, що запобігає виконанню шкідливих JavaScript-скриптів в інтерфейсі користувача.

б). Впровадження політик безпеки на стороні клієнта, таких як Content Security Policy (CSP), допомагає обмежити виконання несанкціонованих скриптів у браузерах, а регулярний аудит журналів та обмеження доступу до них зменшують ризик введення шкідливих даних.

3. CVE-2022-33891 стосується вразливості ін'єкції команд оболонки в інтерфейсі користувача Spark (Spark UI). Вона впливає на версії до 3.1.3 та з 3.2.0 до 3.2.1, дозволяючи зловмисникам виконувати довільні команди оболонки на сервері, де працює Spark. Проблема виникає, коли в конфігурації увімкнено списки керування доступом (ACLs) через параметр конфігурації spark.acls.enable. Зловмисники можуть скористатися цією вразливістю, видаючи себе за автентифікованого користувача та інжектуючи шкідливі команди, які виконує сервер.

Можна виділити наступні способи усунення та запобігання даної уразливості:

а). Оновлення Apache Spark до версії 3.2.2 або новіших (наприклад, 3.3.0), де вразливість ін'єкції команд оболонки була виправлена та для неї був випущений (патч).

б). Перегляд та налаштування списків керування доступом (ACLs) в конфігурації Spark, зокрема вимкнути параметр `spark.acls.enable` або застосувати додаткові заходи безпеки для обмеження доступу лише авторизованим користувачам.

4. CVE-2021-38296 –ця вразливість виникає в процесі узгодження ключів для RPC-з'єднань, захищених параметрами spark.authenticate та spark.network.crypto.enabled в Apache Spark. Проблема виникає у версіях до 3.1.2, де використовується власний протокол взаємної автентифікації для обміну ключами, що може бути вразливим до атак. Зловмисники можуть перехопити та розшифрувати трафік, використовуючи цю вразливість. Це ставить під загрозу конфіденційність переданих даних, оскільки зловмисник може отримати доступ до чутливої інформації, що передається через захищене з'єднання. Ця проблема може призвести до серйозних наслідків, таких як витік даних або несанкціонований доступ до конфіденційної інформації, переданої між клієнтами та серверами. Зловмисники можуть використовувати вразливість для атак на інфраструктуру, де застосовуються захищені з'єднання для передачі критично важливих даних. Для усунення цієї вразливості важливо оновити Apache Spark до версії 3.1.3 або новішої, де було впроваджено виправлення для цього протоколу.

Можна виділити наступні способи усунення та запобігання даної уразливості:

а). Оновлення Apache Spark до версії 3.1.3 або новішої, де виправлено вразливість у процесі узгодження ключів та забезпечено безпечніший протокол для захисту RPC-з'єднань.

б). Перегляд та оновлення конфігурації безпеки для RPC-з'єднань, застосовуючи сучасні методи шифрування, такі як TLS/SSL та алгоритми AES-256 і ECDSA, а також інтегрування двофакторної автентифікації для мінімізації ризику перехоплення та розшифрування трафіку.

5. CVE-2020-9480 – вразливість віддаленого виконання коду (RCE) в Apache Spark, яка стосується менеджера кластерів у режимі "standalone" при увімкненій автентифікації (параметр spark.authenticate). Вразливість дозволяє зловмисникам обійти механізм автентифікації за допомогою спільного секрету і виконувати довільні команди оболонки на хост-машині, де працює Spark master. Ця проблема впливає на версії Apache Spark 2.4.5 та попередні, і може призвести до несанкціонованого виконання команд та компрометації хост-системи. Зловмисники можуть скористатися даною вразливістю для отримання доступу до системи і виконання шкідливих команд, що може призвести до серйозних наслідків, таких як

крадіжка даних, зміна конфігурацій або навіть повний контроль над хост-системою. Вразливість виникає через недосконаліть механізму автентифікації, що дозволяє обійти захист при неправильному налаштуванні. Для усунення даної вразливості необхідно вжити заходів, що включають оновлення до новіших версій Apache Spark і обмеження мережевого доступу.

Можна виділити наступні способи усунення та запобігання даної уразливості:

а). Оновлення Apache Spark до версії 2.4.6 або 3.0.0, в яких було виправлено цю вразливість та покращено механізми автентифікації.

б). Обмеження мережевого доступу до кластерних машин тільки для довірених хостів, яке можна здійснити за допомогою фаєрволів або мережевих правил, які дозволяють доступ лише з IP-адрес, що належать авторизованим користувачам або інфраструктурі. Це запобігає здійснення підключення з ненадійних джерел, мінімізуючи ймовірність того, що зловмисники зможуть отримати доступ до кластера та виконати шкідливі команди або здійснити атаку на систему.

Важливість аналізу логів безпеки та їх роль у Apache Spark

Логи – це своєрідний «цифровий слід» усіх подій у системі, які дають змогу відстежувати активність користувачів, ідентифікувати потенційні загрози та забезпечувати відповідність нормативним вимогам, таким як GDPR або ISO 27001. Вони містять важливу інформацію про несанкціоновані спроби доступу, сканування портів, мережеві аномалії та інші показники, які можуть свідчити про загрози безпеці. Завдяки грамотному аналізу логів організації можуть не лише оперативно реагувати на інциденти, а й запобігати потенційним атакам, мінімізуючи ризики для своїх даних та інфраструктури [17].

Однак класичні методи аналізу логів не здатні ефективно справлятися з їхнім стрімким зростанням та різноманітністю форматів. Тут на допомогу приходять сучасні Big Data технології, зокрема Apache Spark. Завдяки своїй високій продуктивності та можливості розподіленої обробки Spark дозволяє аналізувати величезні масиви даних у реальному часі, використовуючи Hadoop Distributed File System (HDFS) для зберігання та обробки логів у кластерному середовищі. Основний підхід для аналізу включає фільтрацію релевантних записів, їхню трансформацію в структуровані дані за допомогою MapReduce та застосування методів машинного навчання для виявлення аномалій і загроз. Такий підхід дає змогу організаціям швидко виявляти потенційно небезпечну активність, покращувати моніторинг та забезпечувати надійний рівень кібербезпеки.

Після фільтрації застосовується MapReduce, що дозволяє трансформувати дані у пари (джерело → призначення), об'єднувати їх та підраховувати частоту з'єднань. Це дає змогу швидко визначити найбільш активні IP-адреси та потенційні загрози. Заключним етапом є генерація статистики та візуалізація отриманих результатів. Наприклад, завдяки Spark Streaming можна аналізувати логи у реальному часі, виявляючи аномальні активності ще до того, як вони призведуть до серйозних наслідків. Простий приклад фільтрації логів у Spark виглядає наступним чином:

```
lines = sc.textFile("hdfs://path/to/logfile")
filtered_logs = lines.filter(lambda line: "src=" in line and "dst=" in line)
```

Рис. 3. Приклад коду, для початку аналізу логів Apache Spark додатка

Використовуючи таку фільтрацію ми практично отримуємо структуровані дані, що містять інформацію про активність користувачів, підозрілі запити та потенційні загрози. Наприклад, після аналізу логів можна виявити найбільш активні IP-адреси, які здійснюють

© Коваль О. Р., Гарасимчук О. І. (2025). Організація захисту даних під час їх обробки у додатку Apache Spark. Сучасний захист інформації, 1(61), 197–205.
https://doi.org/10.31673/2409-7292.2025.014076.

велику кількість підключень за короткий проміжок часу, що може свідчити про DDoS-атаку або спроби брутфорс-автентифікації.

Приклад отриманих даних:

IP-адреса	Кількість з'єднань	Тип загрози
192.168.1.10	12,543	Підозріла активність (DDoS)
203.0.113.5	8,932	Брутфорс-атака на SSH
185.220.101.1	7,125	Підозрілі запити до API

Рис. 4. Приклад таблиці, після аналізу логів Apache Spark додатка

Після обробки даних можна візуалізувати отримані результати у вигляді графіків або діаграм. Наприклад, використовуючи бібліотеку `matplotlib` у Python, можна побудувати графік розподілу атак за IP-адресами:

```python
import matplotlib.pyplot as plt

ips = ["192.168.1.10", "203.0.113.5", "185.220.101.1"]
connections = [12543, 8932, 7125]

plt.figure(figsize=(8,5))
plt.bar(ips, connections, color=['red', 'blue', 'green'])
plt.xlabel("IP-адреса")
plt.ylabel("Кількість з'єднань")
plt.title("Найбільш активні IP-адреси за кількістю підключень")
plt.show()
```

Рис. 5. Приклад коду, для візуалізації даних після парсингу логів Apache Spark додатка

Рис. 6. Таблиця частоти активності IP-адреси за кількістю підключень до Apache Spark додатка

Ця візуалізація є потужним інструментом для аналізу мережевої активності та виявлення потенційних загроз. Графік демонструє найбільш активні IP-адреси за кількістю з'єднань, що дозволяє адміністраторам безпеки швидко визначити аномальну поведінку. Наприклад, IP-адреса 192.168.1.10, яка має понад 12 500 з'єднань, може бути індикатором DDoS-атаки, сканування портів або інших зловмисних дій.

Такий аналіз дає змогу виявити підозрілу активність у реальному часі, що особливо важливо для запобігання витоку даних та інших кіберзагроз. Використовуючи Apache Spark, можна не тільки визначити найбільш активні IP-адреси, а й провести кореляційний аналіз, щоб встановити взаємозв’язки між підозрілими подіями, а також інтегрувати отримані дані з іншими системами моніторингу безпеки.

Ця візуалізація допомагає адміністраторам безпеки швидко оцінити рівень загрози та вжити необхідних заходів реагування. Наприклад, вони можуть заблокувати підозрілі IP-адреси, обмежити доступ до ресурсів або впровадити додаткові рівні аутентифікації для користувачів із нетиповою активністю. Завдяки таким інструментам компанії можуть ефективно зміцнювати свій кіберзахист, запобігаючи потенційним атакам ще на ранніх стадіях.

Висновки

В результаті проведених досліджень можна зробити висновок, що Apache Spark є потужною платформою для обробки великих даних, яка забезпечує високу швидкість та ефективність обчислень завдяки можливості роботи в пам’яті та розподіленій архітектурі. Однак, платформа має низку вразливостей, які можуть бути використані зловмисниками для отримання несанкціонованого доступу або виконання шкідливого коду. Виявлені проблеми, такі як підвищення привілеїв, вразливості XSS, ін’єкції команд оболонки та компрометація автентифікації, свідчать про необхідність постійного моніторингу безпеки та оновлення програмного забезпечення.

Запропоновані варіанти вирішення знайдених проблем включають оновлення Apache Spark до останніх безпечних версій, обмеження використання proxy-user, жорстке налаштування параметрів конфігурації безпеки, таких як обмеження classpath, впровадження політик безпеки на рівні веб-інтерфейсу, використання сучасних методів шифрування та розмежування доступу до логів. Важливим аспектом також є контроль мережевого доступу до кластерів та регулярний аудит налаштувань, що зменшує ризик компрометації системи.

Крім того, впровадження методів аналізу логів за допомогою Apache Spark та машинного навчання дозволяє виявляти аномалії у реальному часі, що підвищує рівень безпеки даних. Використання Spark Streaming для моніторингу активності, а також MapReduce для структуризації логів допоможе швидше виявляти загрози та реагувати на них. Таким чином, постійна оптимізація процесів безпеки та впровадження передових технологій аналізу даних сприятиме підвищенню захищеності Apache Spark у корпоративному середовищі.

Перелік посилань

1. Дейнека О.Р., Гарасимчук О. І. Виклики та стратегії зберігання великих обсягів даних у сучасному світі // Захист інформації. – 2024. – Т. 25, № 4. – С. 197–207. DOI: https://doi.org/10.18372/2410-7840.25.18225.

2. Deineka, O., Harasymchuk, O., Partyka, A., Obshta, A., Korshun, N. Designing Data Classification and Secure Store Policy According to SOC 2 Type II // CEUR Workshop Proceedings, 2024, 3654, pp. 398–409.

3. Apache Spark Unified engine for large-scale data analytics. URL: http://spark.apache.org.

4. C. S. Karthikeya Sahith, S. Muppidi and S. Merugula, "Apache Spark Big data Analysis, Performance Tuning, and Spark Application Optimization," 2023 International Conference on Evolutionary Algorithms and Soft Computing Techniques (EASCT), Bengaluru, India, 2023, pp. 1-8, doi: 10.1109/EASCT59475.2023.10393086.

5. Y. Tian, Q. Shen, Z. Zhu, Y. Yang and Z. Wu, "Non-Authentication Based Checkpoint Fault-tolerant Vulnerability in Spark Streaming," 2018 IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 2018, pp. 00783-00786, doi: 10.1109/ISCC.2018.8538745.

6. S. Shah, Y. Amannejad and D. Krishnamurthy, "Diaspore: Diagnosing Performance Interference in Apache Spark," in IEEE Access, vol. 9, pp. 103230-103243, 2021, doi: 10.1109/ACCESS.2021.3098426.

7. Spark Security. URL: https://downloads.apache.org/spark/docs/2.4.4/security.html.

8. Introduction to Transparent Data Encryption. URL: https://docs.oracle.com/en/database/oracle/oracle-database/19/asoag/introduction-to-transparent-data-encryption.html.

9. Apache Ranger. URL:https://ranger.apache.org/.

10. Amazon GuardDuty. URL: https://aws.amazon.com/guardduty/.

11. What is data loss prevention (DLP). URL: https://www.kingston.com/en/blog/data-security/data-loss-prevention-dlp.

12. Spark security. URL: https://docs.cloudera.com/runtime/7.3.1/configuring-spark/topics/spark-security.html.

13. Spark custom data sources and sinks for cybersecurity use cases. URL: https://medium.com/@alexott_en/spark-custom-data-sources-and-sinks-for-cybersecurity-use-cases-9623abb94574.

14. Apache Spark Ecosystem – Complete Spark Components Guide. URL: https://data-flair.training/blogs/apache-spark-ecosystem-components/.

15. Park, G., Heo, Y.S., Lee, K. et al. A parallel and accurate method for large-scale image segmentation on a cloud environment. J Supercomput 78, 4330–4357 (2022). https://doi.org/10.1007/s11227-021-04027-5.

16. How Do You Secure Apache Spark? URL: https://granulate.io/blog/spark-security-top-vulnerabilities-6-ways-to-secure-your-spark/.

17. Oktay, T., Sayar, A. (2017). Analyzing Big Security Logs in Cluster with Apache Spark. In: Angelov, P., Manolopoulos, Y., Iliadis, L., Roy, A., Vellasco, M. (eds) Advances in Big Data. INNS 2016. Advances in Intelligent Systems and Computing, vol 529. Springer, Cham. https://doi.org/10.1007/978-3-319-47898-2_14.

Надійшла 14.02.2025