УДК 004.75/519.687.5

Л. Рибачук, С. Жевакін

АНАЛІЗ МЕТОДІВ ДЛЯ ЗАХИСТУ ДАНИХ ПРИ ВИКОРИСТАННІ МУЛЬТИХМАРНОГО ПІДХОДУ

Анотація: В даній статті представлений результат аналізу наявних методів для захисту даних при використанні мультихмарного підходу. Також був запропонований алгоритм порціювання даних застосунку з подальшим шифруванням для збільшення захищеності даних.

Ключові слова: кібербезпека, хмарні розрахунки, мультихмара, хмарні обчислення, захист даних

Опис проблеми

У наші дні захищеність даних, відмовостійкість та захищеність бізнес-процесів стоїть на першому місці серед IT підприємств. Щороку для забезпечення описаних потреб компанії витрачають приблизно 12% від свого бюджету [1]. У середньому, після кібератаки бізнесу потрібно 8 місяців, щоб відновити свою роботу до рівня, який був до неї. Так само 25% дрібних бізнесів, що пережили кібератаку, перестають існувати. Для протидії можливим кібератакам компанії залучають більше фахівців у сфері кібербезпеки, цей висновок можна зробити з аналізу ринку фахівців з кібербезпеки, що нескінченно зростає. В період з 2021 по 2022 рр. попит на фахівців з кібербезпеки серед державного сектору виріс на 25%, а серед приватного сектору – на 21 % [2].

Для покращення кібербезпеки все більше компаній вибирають хмарний підхід до розташування ресурсів. Так дослідження Cynet's 2021 [3] показало, що 57% опитаних директорів з інформаційної безпеки пріоритезує хмарний підхід до розташування сервісів, у той час як 21% опитаних віддають перевагу локальному підходу для розгортання сервісів.

Хмарні розрахунки

Хмарні розрахунки можна охарактеризувати наступним чином – це виділення розрахункових ресурсів за запитом через мережу інтернет по факту їх використання, це парадигма, в рамках якої інформація постійно зберігається на серверах у мережі інтернет і тимчасово кешується на клієнтській стороні,

наприклад, на персональних комп'ютерах, ігрових приставках, ноутбуках, смартфонах тощо [4]. Наявні чотири підходи розташування ресурсів в хмарі на даний момент:

1) приватна хмара;
2) публічна хмара;
3) гібридна хмара;
4) мультихмара.

Кожен з цих підходів має як свої переваги так і недоліки. Приватна хмара представляє собою набір серверного забезпечення, яке розміщується в дата-центрі замовника, даний підхід передбачає те, що організація сама відповідальна за управління, налаштування та оновлення компонентів. Публічна хмара в свою чергу надає свої серверні ресурси користувачам на основі оплати тільки за використані ресурси [5]. В даному випадку компанії не потрібно облаштовувати й підтримувати дата-центри та комплектуючі, замість користувачів це робить провайдер хмарних сервісів. Гібридна хмара представляє собою симбіоз приватної та публічної хмари. При подібному підході всі критичні елементи інфраструктури розміщені на серверах та дата-центрах замовників. В публічну ж хмару винесені сервіси, вихід із строю яких не понесе за собою значних втрат. Мультихмара представляє собою підхід, при якому кілька постачальників хмарних сервісів використовуються однією компанією, задля отримання певних переваг від різних постачальників хмарних сервісів. Загалом, використання мультихмари приносить зменшення загальної вартості обслуговування інфраструктури, також даний підхід дозволяє вибирати найліпший варіант серед наявних рішень у різних провайдерів.

Переваги використання мультихмарного підходу

Зі швидкісним розвитком технологій, компаніям все частіше доводиться шукати альтернативні підходи до розробки програмного забезпечення для зберігання і обробки великих потоків даних. В цьому їм може допомогти мультихмарний підхід розробки програмного забезпечення – підхід, при якому частини програмних застосунків розосереджені серед декількох хмарних постачальників. Цей підхід покращує користувальницький досвід використання сервісів. Також даний підхід забезпечує ефективність зберігання та обробки даних, як і збільшує безпеку інфраструктури в цілому.

Виходячи з потреби передачі і зберігання великої кількості Ентерпрайз-даних, компанії використовують мультихмарний підхід. Як показало дослідження, в середньому 4,9 постачальників хмарних послуг використовують для розміщення своїх сервісів Ентерпрайз застосунки і ця цифра продовжує зростати [6].

Одним з найголовніших плюсів використання мультихмарної архітектури є її автономність. Вона полягає в здатності розгортати програми у різних постачальників хмарних послуг незалежно від можливостей та зон дії постачальників хмарних послуг. Так само автономність від них, дозволяє компанії диктувати умови надання послуг постачальниками, оскільки в кінцевому рахунку замовник має альтернативні варіанти розміщення.

Також даний підхід надає гнучкість використання сервісів, тобто система не залежить від постачальника хмарних послуг, є можливість вибору між кількома постачальниками та регіонами надання послуг постачальниками. Таким чином, є змога переадресовувати запит на найближчий дата-центр, який є в оренді у замовника, що дозволяє зменшити час відгуку програми для всіх користувачів, незважаючи на їх розміщення. Також використання мультихмари надає широкий діапазон для росту інфраструктури.

Огляд наявних підходів до захисту даних при використанні мультихмари

Найбільшою проблемою при використанні хмари для кінцевих користувачів є те, що користувачі не можуть знати, як їх дані зберігаються і використовуються, так само через закритість хмарних систем користувачі не можуть бути впевнені до кінця в тому, що вони спілкуються з хмарою, а не із зловмисниками, які підлаштували атаку та відсилають відповіді замість хмари.

Головною перевагою використання мультихмарного підходу є підвищена захищеність застосунку, а саме мультихмарний підхід відкриває нові можливості для зберігання та засекречування даних застосунку.

Існує кілька підходів, як користувачі можуть посилити безпеку своїх додатків при залученні мультихмари.

Серед таких підходів можна виділити наступні.

Реплікація застосунку – при використанні даного підходу додатки розміщуються у різних хмарних провайдерів, таким чином при надсиланні запиту від користувача на сервер застосунку, запит буде надісланий всім його реплікам. Після отримання відповідей від усіх доступних реплік, відповіді порівнюються і в

результаті вибирається відповідь, яка була отримана більшу кількість разів. Так само за допомогою реплікації додатку можна створити сервіс, який контролюватиме роботу інших сервісів, в даному випадку виступаючи в ролі master в зв’язку master-slave. Таким чином сервер буде верифікувати роботу інших сервісів. При використанні даного підходу можна буде визначити істинний результат програми за допомогою голосування більшості.

Розподіл застосунку – даний тип розміщення сервісів захищає від потенційного витоку даних, так як дані розміщені у різних хмарних провайдерів. В даному випадку захищеність даних не залежать від несправностей або ж від поломок на стороні постачальника хмарних послуг, на якому розміщений сервіс програмного застосунку. Крім додаткових витрат на захищеність хмари, на якій зберігаються дані, ця архітектура також потребує стандартного інтерфейсу для кореляції між даними та додатками, що знаходяться у різних хмарах. Для вирішення цієї проблеми можна верифікувати запити через користувача за наступною схемою, якщо додатку, що знаходяться в хмарі А, потрібні дані, які знаходяться в хмарі В, то сервіс з хмари А формує запит і надсилає користувачу, користувач підтверджує запит і дані вирушать на хмару А з хмари В. У цьому випадку користувач отримує повний контроль над потоком даних і найголовніше, таким чином можна уникнути витоків даних, навіть якщо в програмному коді, який розміщується в хмарі А, були допущені недоліки, які можуть спровокувати витік даних.

Порціювання логіки застосунку – при використанні даного патерну логіка застосунку розділяється поміж декількома постачальниками хмарних послуг. Розбиття застосунку між постачальниками проходить наступним чином – критичне обчислюване ядро системи виноситься до більш захищеного клауду, а інші частини застосунку можуть розміщатись у менш захищених хмарних провайдерів. Не критичні частини застосунку можуть бути розміщені на менш захищених хмарах, так як подібні хмари, в цілому, дешевші в використані. Оскільки критичне ядро системи являє собою невелику частину застосунку, то є можливість розмістити його на більш захищених хмарах, таким чином можна заощадити на розгортанні застосунка в цілому. Головною проблемою даного підходу є те, що немає уніфікованого способу розбиття програми на частини. Під час реалізації подібного підходу потрібно залучити велику кількість спеціалістів, тому даний процес не є швидким.

Порціювання даних застосунку з подальшим шифруванням

Розглянуті методи, в основному, сконцентровані на відмовостійкості сервісів, проте вони не приділяють увагу захисту даних користувачів. Для покращення захищеності даних був розроблений алгоритм порціювання даних застосунку з подальшим шифруванням.

Підхід з порціюванням даних застосунку з подальшим шифруванням полягає в поділі даних між кількома клауд-провайдерами. Порціювання проводиться над структурованими даними. Алгоритм порціювання даних серед декількох постачальників хмарних послуг зображений на рисунку 1 й включає в себе наступні етапи:

1) сервер застосунку передає інформацію до захищеної хмари;

2) на захищеній хмарі сервіс, який проводить вертикальну фрагментацію отримує дані й проводить їх фрагментацію;

3) сервіс фрагментації записує інформації стосовно відфрагментованих даних в базу даних для зворотної синхронізації даних, яка розміщена в тій же захищеній хмарі;

4) відфрагментовані дані відправляються на сервер, на якому відбувається шифрування даних;

5) зашифровані дані розосереджуються серед баз даних, які розміщені серед публічних постачальників хмарних послуг.

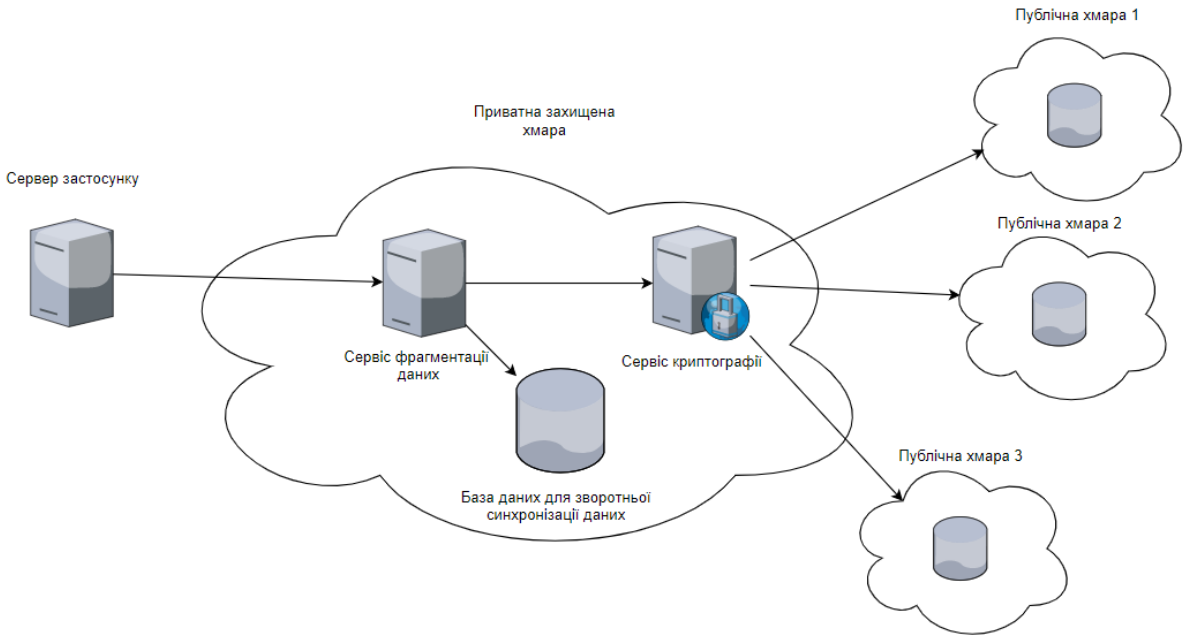

Рисунок 1. Схема порціювання даних застосунку з подальшим шифруванням

Для порціювання даних використовуються алгоритми вертикальної фрагментації. Даний вид фрагментації являє собою поділ однієї таблиці бази даних на кілька фрагментів або секцій на основі стовпців або атрибутів таблиці бази даних [7]. Кожен фрагмент містить підмножину стовпців вихідної таблиці, і поділ зазвичай базується на попередньо визначеному наборі правил або критеріїв. Для даного сценарію використаємо алгоритм BEA [8].

Логіка даного алгоритму полягає в кластеризації сутностей, в даному випадку – атрибутів таблиці. Кластеризація атрибутів відбувається на основі частоти їх використання в запитах до бази даних або ж до наявності їх в індексах початкової бази даних. Фактична міра підраховує кількість разів, коли пара атрибутів використовується разом за певний період часу. Для визначення даної міри описуються наявні типові запити до бази даних. Ідея алгоритму полягає в тому, що атрибути, які використовуються разом, утворюють кластер і повинні зберігатися разом.

Нехай таблиця в базі даних має n атрибутів

$$A = (A_1, A_2, \dots, A_n),$$

$$AM = \sum_{i=1}^{n} (bond(A_i, A_{i-1}) + bond(A_i, A_{i+1})). \quad (1)$$

Міра подібності визначена в формулі (1). Дана формула використовується для оцінки результатів проведеної кластеризації. Функція $bond$ визначає частоту використання елементів A_i, A_{i+1} та A_{i-1} разом в запитах. Алгоритм кластеризує елементи таким чином, щоб вихідні кластери мали елементи, які частіше зустрічаються разом в запитах. На рисунку 2 наведено приклад результуючої кластеризації.

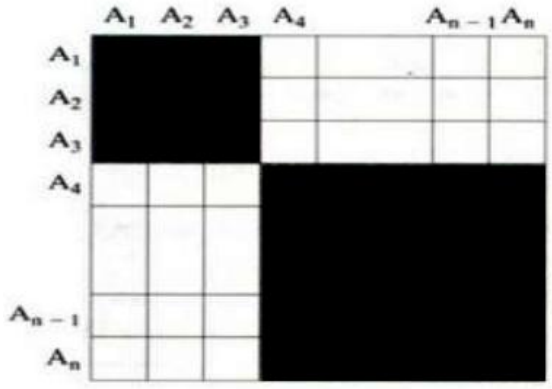

Рисунок 2. Результат проведення кластеризації

У розподіленій базі даних кожен результуючий кластер називається вертикальним фрагментом. В подальшому ці фрагменти шифруються в сервісі криптографії й розосереджуються серед баз даних, які розміщені в публічних хмарах. Таким чином при запиті до бази даних можна буде залучити тільки підмножину атрибутів, наявних в фрагментах, а не запитувати весь набір атрибутів щоразу.

Використання даного підходу забезпечує захищеність інформації застосунку від витоку даних, адже якщо зловмисникам вдасться отримати доступ до даних, які розміщені в базах даних в публічних хмарах, то вони нічого з нею не зможуть зробити без приватних ключів, які зберігаються в приватній захищеній хмарі. Також жоден із провайдерів не може мати представлення про загальну структуру даних, адже кожен зберігає лише частину зашифрованих даних.

Висновки

В результаті проведеного аналізу було розглянуто декілька підходів до захисту даних й логіки бізнес-процесів при використанні мультихмари. Описані підходи демонструють, як використання мультихмари покращує захист даних застосунків. Кожен з розглянутих підходів має свої переваги та недоліки, тому неможливо виконати загальну їх оцінку. Порівняльне дослідження основних характеристик таких, як цілісність даних, захищеність даних, простота в використанні, відмовостійкість представлено в Таблиці 1.

Таблиця 1.

Порівняння розглянутих підходів за визначеними характеристиками

Розглянуті підходи	**Характеристики**			
	Цілісність даних	**Захищеність даних**	**Складність імплементації**	**Відмовостійкість**
Реплікація застосунку	Середня	Низька	Низька	Висока
Розподіл застосунку	Середня	Низька	Низька	Висока
Порціювання логіки застосунку	Середня	Середня	Висока	Низька
Порціювання даних застосунку з подальшим шифруванням	Висока	Висока	Середня	Низька

На сьогоднішній день не існує підходу, який би повністю покривав представлені характеристики. Але серед розглянутих підходів виділяється запропонований підхід порціювання даних застосунку з подальшим шифруванням, даний підхід забезпечує цілісність даних з високою захищеністю даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Thorpe N. Cybersecurity spending - here is how much your company need to spend. *Data centric security solutions - SecureAge Technology*. URL: https://www.secureage.com/blog/cybersecurity-spending-a-necessary-evil.

2. 2023 cyber-attack statistics, data, and trends | parachute. *Parachute | Managed IT Services in the San Francisco Bay Area and Sacramento Valley*. URL: https://parachute.cloud/cyber-attack-statistics-data-and-trends/.

3. Kolakowski N. Market for cybersecurity specialists remains strong. *Dice Insights*. URL: https://www.dice.com/career-advice/market-for-cybersecurity-specialists-remains-strong.

4. Pay-as-you-use - Wikipedia. *Wikipedia, the free encyclopedia*. URL: https://en.wikipedia.org/wiki/Pay-as-you-use (date of access: 10.01.2023).

5. Why you should care about multicloud. *InfoWorld*. URL: https://www.infoworld.com/article/2611544/why-you-should-care-about-multicloud.html.

6. Cheng W., Feng H., Liang G. Design of IT infrastructure multicloud management platform based on hybrid cloud. Wireless communications and mobile computing. 2022. Vol. 2022. P. 1–12. URL: https://doi.org/10.1155/2022/9227948 (date of access: 10.01.2023).

7. Fragmentation in Distributed DBMS - GeeksforGeeks. URL: https://www.geeksforgeeks.org/fragmentation-in-distributed-dbms/.

8. Gaurav N. A vertical partitioning algorithm for distributed object-oriented databases. *International journal of computer applications*. 2015. Vol. 119, no. 2. P. 19–24. URL: https://doi.org/10.5120/21040-3491 (date of access: 29.03.2023).