НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут прикладного системного аналізу
Кафедра системного проектування

До захисту допущено:
Завідувач кафедри
_____ Вадим МУХІН
«___»_____2022 р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою
“Інтелектуальні сервіс-орієнтовані розподілені обчислювання”
зі спеціальності 122 "Комп'ютерні науки"
на тему: «Аналіз програмних засобів шифрування даних на стороні клієнта при використанні хмарних технологій»

Виконала:
студентка IV курсу, групи ДА-81
Козінцева Анна Олексіївна _____
Керівник:
Доцент, к.т.н.,
Капшук Олег Олексійович _____
Консультант з економічного розділу:
Доцент, к.е.н.,
Рощина Надія Василівна _____
Рецензент:
Професор, доктор технічних наук
Бідюк Петро Іванович _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2022

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра системного проектування

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 122 "Комп'ютерні науки"
Освітньо-професійна програма – "Інтелектуальні сервіс-орієнтовані розподілені обчислювання"

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Вадим МУХІН
«___»_____2022 р.

ЗАВДАННЯ
на дипломну роботу студенту
Козінцевій Анні Олексіївні

1. Тема роботи «Аналіз програмних засобів шифрування даних на стороні клієнта при використанні хмарних технологій», керівник роботи Капшук Олег Олексійович, кандидат технічних наук, доцент, затверджені наказом по університету від «___»_____ 20__ р. №_____
2. Термін подання студентом роботи – 17 червня 2022 р.
3. Вихідні дані до роботи: сучасні програмні засоби шифрування даних на стороні клієнта.
4. Зміст роботи
   4.1. Розгляд особливостей захисту даних у хмарних технологіях;
   4.2. Аналіз атак на хмарні сховища;
   4.3. Розгляд особливостей шифрування даних на стороні клієнта;
   4.4. Дослідження програмних засобів шифрування даних на стороні клієнта.
   4.5. Формування рекомендацій щодо вибору програмних засобів шифрування даних на стороні клієнта.
5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): Електронна презентація
6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В., кандидат наук, доцент		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	10.05.2022	
2	Аналіз літератури по шифруванню даних на стороні клієнта	10.05.2022	
3	Розгляд особливостей зберігання даних у хмарних технологіях, аналіз можливих атак на хмарні сховища	16.05.2022	
4	Дослідження особливостей використання шифрування на стороні клієнта	24.05.2022	
5	Дослідження програмних засобів шифрування даних на стороні клієнта, формування рекомендацій щодо вибору програмного засобу	30.05.2022	
6	Функціонально-вартісний аналіз програмного засобу	06.06.2022	
7	Оформлення дипломної роботи	10.06.2022	
8	Отримання допуску до захисту	19.06.2022	

Студент Анна Козінцева

Керівник Олег Капшук

АНОТАЦІЯ

бакалаврської дипломної роботи Козінцевої Анни Олексіївни на тему «Аналіз програмних засобів шифрування даних на стороні клієнта при використанні хмарних технологій»

Загальний обсяг роботи 86 с., 19 рис., 8 табл., 0 додатків, 24 джерела.

Метою даної роботи є дослідження засобів захисту даних при використанні хмарних технологій, а саме засобів шифрування даних на стороні клієнта.

У першому розділі розглядаються особливості зберігання даних у хмарних технологіях та методи захисту, які використовуються для підвищення надійності зберігання даних у хмарних сервісах.

Другий розділ присвячений дослідженню можливих вразливостей хмарних технологій, які можуть призвести до несанкціонованого доступу до даних користувача сервісу третіми сторонами. У даному розділі проводиться аналіз атак, які здійснюються на хмарні технології, та визначаються способи підвищення захисту даних користувача.

У третьому розділі здійснюється аналіз впливу розміщення ключів шифрування на надійність зберігання даних. Здійснено дослідження можливих способів зберігання ключів на стороні сервера та на стороні клієнта.

Четвертий розділ присвячений аналізу особливостей шифрування даних на стороні клієнта. У даному розділі здійснюється дослідження способів підвищення надійності зберігання даних за рахунок поєднання різних методів шифрування, а саме шифрування на стороні клієнта із шифруванням на стороні сервера, та використання наскрізного шифрування для розширення можливостей шифрування на стороні клієнта.

П’ятий розділ присвячений дослідженню програмних засобів шифрування даних на стороні клієнта, визначенню їхніх переваг та недоліків,

формуванню рекомендацій для вибору того чи іншого програмного засобу в залежності від потреб користувача.

У шостому розділі досліджуються хмарні сервіси, які використовують шифрування на стороні клієнта. Здійснено розгляд їхніх характеристик, визначено переваги та недоліки та сформовано рекомендації щодо вибору сервісу.

Результатом дипломної роботи є визначення переваг та недоліків існуючих програмних засобів шифрування даних на стороні клієнта та формування рекомендацій щодо вибору програмного засобу.

Ключові слова: KMS, платформа з нульовим рівнем знань, AES, ключі шифрування.

ABSTRACT

to the bachelor's thesis of Anna Kozintseva on the topic «Analysis of client-side data encryption software means for cloud technologies»

The total volume of work is 86 p., 19 fig., 8 tabl., 0 append., 24 sources

The aim of this work is to do research on client-side encryption utilities for the cloud.

The first chapter is devoted to investigation of cloud services' storage features and data protection methods which increase the reliability of data stored in these services.

In the second chapter we do research on possible vulnerabilities in cloud technologies, analyze attacks on cloud technologies which can be caused by these vulnerabilities and define ways on improving data protection.

The Third chapter is dedicated to analysis of the impact of place where encryption key is stored, on storage security. Here we check how storing keys impacts server side and client side security.

The fourth chapter is devoted to the analysis of client side encryption features. Here we do research on how data storage can be secured in a better way with a use of combination of server side and client side encryptions, and use of end-to-end encryption to enhance client-side encryption.

The fifth chapter is devoted to investigation of client-side encryption tools, their advantages and disadvantages. Here we form recommendations on choosing an encryption tool in accordance with the user's needs.

The seventh chapter explores cloud services that use client-side encryption, their characteristics, advantages and disadvantages. Recommendations were formed on what cloud service to choose depending on user's needs.

The result of the thesis is recommendations regarding which client-side encryption utility to choose.

Keywords: KMS, Zero Knowledge Platform, AES, encryption keys.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

- KMS – (Key Management Service) – сервіс керування ключами, який створює ключі шифрування та організовує керування цими ключами з використанням засобів безпеки для надійності зберігання ключів.
- ZKP – (Zero Knowledge Platform) – платформа з нульовим рівнем знань, яка забезпечує зберігання даних на своїй стороні (серверах), і при цьому не може переглядати ці дані, адже вони є зашифрованими.

ВСТУП

З кожним роком люди все більше застосовують хмарні технології, адже це дозволяє їм отримувати доступ до даних з будь-якого іншого свого пристрою, і навіть при його виході з ладу дані не будуть втрачені, а продовжать зберігатись на сервері й користувач матиме до них доступ. Постає питання забезпечення надійності їхнього зберігання у хмарному сервісі. Хоча хмарні провайдери забезпечують певний захист зі своєї сторони, цього може бути недостатньо, адже будь-яка система має свої вразливості, якими можуть скористатись кіберзлочинці та отримати доступ до даних користувача. Також у випадку звичного використання хмарного сервісу доступ до даних є як у користувача, так і у провайдера сховища. Провайдер, у свою чергу, може непомітно використовувати дані користувача у своїх цілях. Для того, щоб забезпечити максимальну конфіденційність даних, при якій лише власник даних матиме до них доступ, почала використовуватись технологія шифрування даних на стороні клієнта.

Метою даної роботи є аналіз технології шифрування даних на стороні клієнта та дослідження програмних засобів, які її використовують.

У даній роботі розглянуто наступні питання:

1. Особливості зберігання даних у хмарних технологіях.
2. Аналіз атак на хмарні технології. Розгляд способів підвищення захисту.
3. Дослідження особливостей використання шифрування на стороні клієнта.
4. Аналіз програмних засобів шифрування даних на стороні клієнта.
5. Формування рекомендацій щодо вибору програмних засобів шифрування даних на стороні клієнта.

1 ОСОБЛИВОСТІ ЗБЕРІГАННЯ ДАНИХ У ХМАРНИХ ТЕХНОЛОГІЯХ

Хмарні технології – це технології для обробки даних на віддалених серверах, при використанні яких обчислювальні ресурси надаються користувачеві в якості онлайн-сервісу.

Хмарна система загалом являє собою єдиний сервер даних, під'єднаний до мережі. Центр обробки даних можна розглядати як об'єкт, на якому розміщується багато серверів. Також центри обробки даних мають певну кількість серверів для виконання складних операцій для бізнесу.

1.1 Розміщення даних у хмарних технологіях

Раніше фізичний сервер представляв собою сховище для даних комп'ютера. З розвитком хмарних технологій дані почали зберігатися у центрах обробки даних, а потреба у використанні фізичного сервера суттєво зменшилась. Якщо користувач хоче зберігати свої дані на сервері, він «орендує» місце на цьому сервері. В залежності від потреб користувача кількість задіяних ресурсів сервера може бути збільшена або зменшена, а рахунок для клієнта формується у відповідності до їх використання. Це принцип роботи хмари в комерційних цілях.

З технічної точки зору, хмарне сховище розміщує дані користувача шляхом надсилання їх копії через мережу Інтернет на сервер, який записує інформацію. Якщо виникає необхідність – ця інформація отримується через веб-інтерфейс. Подібно до використання жорсткого диску комп'ютера, користувач може отримати доступ до даних, виконувати різні операції над даними або стирати дані, в залежності від потреби.

Резервування – зберігання кількох копій одних і тих самих даних на багатьох хмарних серверах у якості резервної копії – гарантує, що користувач

може отримати доступ до своїх даних будь-коли і з будь-якого місця, навіть при оновленні хмари.

Резервування можна використовувати різними способами, наприклад, створити резервну копію даних або архів, щоб звільнити місце для нових даних. Це гарантує, що при виході з ладу системи – користувач продовжить мати доступ до власних даних.

Подібно до антивіруса або захисту паролем на комп'ютері, хмара також використовує методи захисту, які переважно є складними і мають різні рівні захисту від загроз. Таким чином забезпечується безпека даних клієнта у хмарному сховищі.

1.2 Захист даних у хмарних технологіях

Існує два стани даних:

- дані в стані зберігання (дані у сховищі)
- дані, що обробляються або передаються

В обох цих станах дані повинні бути захищені.

Для захисту даних, які знаходяться у будь-якому з цих двох станів, хмарні обчислення використовують спеціальні інструменти та методи.

1.2.1 Захист даних у стані зберігання

Дані у стані зберігання (спокою) зазвичай зберігаються у програмі, яку використовує постачальник послуг у центрі обробки даних, а безпека цих даних досягається за допомогою процесу шифрування.

Хмарне шифрування – це послуга, яка надається постачальником сховища, при якій за рахунок кодування даних, яке використовує математичні алгоритми, досягається неможливість їхнього прочитання у випадку перехоплення, тобто інформація перетворюється у зашифрований текст.

Існує два методи шифрування: симетричний і асиметричний.

Симетричне шифрування полягає у використанні однакових ключів як для шифрування, так і для дешифрування даних.

Асиметричне шифрування використовує пару ключів: приватний та публічний. Публічні ключі можна використовувати спільно, а приватні потрібно зберігати в секреті. За рахунок математичної формули обидва ключі пов’язані між собою, тому пара ключів діє разом, що забезпечує конфіденційність та цілісність даних.

1.2.2 Захист даних у стані обробки або передачі

Для захисту даних у стані обробки або передачі використовуються наступні протоколи:

- **TLS (безпека транспортного рівня)**

TLS – протокол безпеки транспортного рівня – має вбудовані компоненти, які дозволяють безпечно спілкуватись через хмару хмарним програмам, у всіх станах даних над ними зберігається контроль. Наприклад, при використанні TLS надісланий до хмарної служби Google запит захищається протягом усього процесу передачі за допомогою протоколу HTTPs, а від публічного центру сертифікації використовується захищений сертифікат.

Наприклад, для отримання доступу до облікових записів Gmail із хмари потрібно здійснити процес входу. Вхід відбувається через захищений віртуальний сервер, після чого користувачеві видається унікальний цифровий сертифікат, ідентифікатор якого має наступні елементи: IP-адресу, MAC-адресу, характеристики пристрою користувача, характеристики браузера та ресурс, доступ до якого здійснюється [2]. Якщо користувач заходить з раніше не відомої системі точки, йому надсилається сповіщення про те, що була здійснена спроба увійти до його електронної пошти. У повідомленні

вказуються такі параметри, як IP-адреса, країна і тип пристрою, з якого була здійснена спроба входу. Дані параметри запобігають зловмисним зломам за рахунок використання автентичного сертифікату.

- **Протокол Secure Shell**

Secure Shell Protocol (SSH) — це зашифрований протокол, який надає хмарам і мережевим службам можливість безпечно працювати у «небезпечному» середовищі в рамках клієнтських фреймворків. Клієнт SSH підключається до сервера SSH, що є зручним у випадку віддаленого доступу. Існує два типи протоколів SSH: SSH1 і SSH2: за допомогою SSH1 будь-яка мережа може отримувати зашифровані дані; SSH2, у свою чергу, є важливим у хмарних обчисленнях, бо він забезпечує безпечний зв’язок навіть коли клієнт знаходиться в середовищі з прямим впливом загроз [2].

- **Internet Protocol Security**

IPSec має на меті захист IP-зв'язку у хмарі. Оскільки у сеансах зв’язку містяться різні пакети даних, IPSec використовується для шифрування та аутентифікації для кожного переданого пакету, а також він забезпечує аутентифікацію щоразу при узгодженні криптографічного ключа. Для функцій хмарних програм є важливими такі середовища як HTTP, SMTP та IMAP, підтримку безпеки яких здійснює IPSec [2].

1.3 Керування ключами шифрування

Керування ключами шифрування полягає у зберіганні, захисті, адмініструванні та резервному копіюванні ключів шифрування. Усі ключі повинні бути збережені та захищені надійно, адже у випадку пошкодження, викрадення чи втрати ключів з’являються значні організаційні наслідки, а саме відбувається втрата доступу до критичних систем і даних, система може стати непридатною до використання.

Життєвий цикл зашифрованого ключа складається із створення, використання та видалення ключа, а за сам цикл відповідає керування ключем шифрування. Життєвий цикл ключа шифрування є тим коротшим, чим чутливіші дані, тобто високочутливі дані шифруються ключами, які мають дуже недовготривалий термін служби.

Система керування ключами (Key Management System – KMS) відповідає як за генерацію ключів, так і за їхнє зберігання та керування. Велика кількість сервісів розробила власні системи керування ключами, які або безпосередньо пов’язані з тим чи іншим сервісом, або просто надають можливість обрати необхідний з наявних сервісів та інтегруватися з ним.

ВИСНОВКИ ДО РОЗДІЛУ 1

Дані у хмарному сховищі зберігаються на серверах, якими керує центр обробки даних. Існує два стани, у яких можуть знаходитись дані: стан зберігання (спокою) і стан передачі (руху).

Для безпеки даних під час використання хмарного сервісу використовуються додаткові методи захисту. Для даних, які знаходяться у стані передачі від користувача до сервера або навпаки, для надійності та безпечності їхньої відправки використовуються спеціальні протоколи, такі як TLS, Secure Shell та IPSec. Для даних, які вже зберігаються у хмарному сховищі, додатково використовуються методи шифрування з метою запобігти втраті конфіденційної інформації користувача у випадку зламу системи зловмисниками.

Для шифрування даних, які знаходяться у стані зберігання, необхідно надійним чином згенерувати та зберегти ключі шифрування, за що відповідають системи керування ключами.

2 АНАЛІЗ АТАК НА ХМАРНІ СХОВИЩА

Користувачі хмарних сервісів зберігають у хмарних сховищах різні типи даних, зокрема такі, які містять конфіденційну інформацію. Однак ці дані можуть бути втрачені або пошкоджені, що може бути пов’язано як із вразливістю самих програм, так і дій людини. Оскільки хмарні технології досі активно розвиваються, у них є багато вразливостей, внаслідок яких зловмисники можуть проникнути в систему.

Розглянемо, якими можуть бути вразливості хмарних сервісів.

2.1 Вразливості хмарних сервісів

- **Вразливості хмарного API**

Для взаємодії з хмарними сервісами використовуються інтерфейси прикладного програмування, API, однак їхні вразливості можуть впливати на безпечність використання сервісу, саме тому для надійності роботи сервісу розробникам потрібно застосовувати потужний контроль над API.

- **Зловмисні інсайдери**

Деякі користувачі хмари мають злочинні наміри щодо сервісу та даних, які він зберігає, та можуть влаштувати витік даних зі сховища. Для запобігання таким видам атак розробники застосовують технології для ідентифікації та керування доступом.

- **Слабка криптографія**

Попри використання криптографічних алгоритмів для захисту даних, хмарні провайдери часто використовують обмежені джерела ентропії, наприклад час, для генерування випадкових чисел, які використовуються для шифрування даних, автоматично. У випадку віртуальної машини Linux ключі генеруються за мілісекунду, але для надійного шифрування цього може бути недостатньо, адже для злому кіберзлочинці використовують складні

механізми. Тому важливим є продумати, яким чином захистити дані перед їхньою відправкою до хмарного сховища.

- **Вразливості хмарних сервісів**

Платформи хмарних обчислень представляють собою розподілені системи хмарних сервісів, які пов’язані між собою і мають слабкий захист один від одного. Тому кіберзлочинець може використати вразливість конкретного хмарного сервісу щоб проникнути до інших і отримати доступ до даних користувачів. Для запобігання потрібно створити потужну архітектуру системи.

2.2 Найпоширеніші типи атак на хмарні обчислення

Для здійснення атак на хмарні сервіси кіберзлочинці постійно вдосконалюють або придумують нові способи атак. Розглянемо найпоширеніші типи атак, щоб розуміти яким чином можна покращити безпеку зберігання даних у хмарних сервісах.

1) Хмарні атаки зловмисного програмного забезпечення

Для того, щоб взяти під контроль інформацію користувача в хмарі, здійснюються атаки з ін’єкцією шкідливого програмного забезпечення, завдяки яким кіберзлочинці додають заражений модуль сервісу до рішення. Якщо хакерам вдалося успішно заплутати хмарну систему, вона перенаправляє запити користувача хмари на екземпляр зловмисника, спровоковуючи виконання шкідливого коду, внаслідок чого хакер може почати маніпулювати, викрадати або підслуховувати дані.

Найчастіше здійснюються міжсайтові атаки сценаріїв та атаки ін’єкції SQL. Наприклад, шкідливі сценарії додаються на цільову веб-сторінку, наприклад, через JavaScript. За таким принципом у 2011 році німецькі дослідники здійснили XSS-атаку на платформу хмарних обчислень Amazon Web Services [3].

У випадку ін’єкції SQL ціллю зловмисників є сервери SQL із вразливими програмами для баз даних, як от у 2008 році веб-сайт Sony PlayStation став жертвою атаки SQL-ін'єкції [3].

2) Зловживання хмарними сервісами

Цілями для здійснення DoS-атак або атак грубої сили зазвичай є дешеві хмарні сервіси та їхні користувачі. Наприклад, у 2010-му році експерти з безпеки Брайан і Андерсон влаштували DoS-атаку, використовуючи можливості хмарної інфраструктури Amazon EC2, а результаті чого їм вдалося зробити свого клієнта недоступним в Інтернеті, витративши лише 6 доларів на оренду віртуальних послуг [3].

Також у 2011 році атака грубої сили була продемонстрована Томасом Ротом на конференції з технічної безпеки Black Hat, подібно до того, як орендуючи сервери у хмарних провайдерів, кіберзлочинці можуть використовувати потужні хмарні можливості для надсилання тисяч можливих паролів до облікового запису цільового користувача [3].

3) Атаки відмови в обслуговуванні

DoS-атаки перевантажують систему з метою зробити її недоступною для користувачів. Навіть вихід з ладу одного сервера може стати серйозною проблемою, наслідок якої відчує багато користувачів. Це відбувається за рахунок того, що під час великого навантаження хмарні системи починають надавати більше обчислювальної потужності і залучаючи все більше віртуальних машин. Потім швидкість роботи хмарного сервісу сповільнюється, і користувачі не можуть отримати доступ до послуг хмари. Чим більше машин хакери використовують для DDoS-атаки – тим небезпечніше хмарному середовищі DDoS-атаки.

4) Атака «людини в хмарі»

При здійсненні такої атаки хакери перехоплюють і переналаштовують хмарні служби. Вони використовують вразливі місця в системі маркерів синхронізації, щоб під час наступної синхронізації з хмарою маркер синхронізації був замінений на новий, який надасть доступ хакерам. Також зловмисник може повернути оригінальні маркери синхронізації в будь-який момент, тому користувачі можуть і не дізнатися, що їхні облікові записи зламано, а крім цього існує ризик того, що зламані облікові записи не можна буде відновити.

5) Атаки бічного каналу

За рахунок розміщення хакерами шкідливої віртуальної машини на тому ж хості, що й цільова віртуальна машина, здійснюється атака бічного каналу. Під час атаки на бічний канал хакери намагаються зламати системні реалізації криптографічних алгоритмів, але таких атак можна уникнути за допомогою використання безпечної системи.

6) Обгортання атак

Атака обгортання – це один з різновидів атаки «людина посередині» у хмарному середовищі, яка є особливо небезпечною для хмарних сервісів, адже користувачі хмари зазвичай підключаються до провайдерів використовуючи вею-браузер. Для захисту облікових даних користувачів від несанкціонованого доступу використовується підпис XML, однак даний підпис не захищає позиції в документі, тому обгортання елементів за допомогою підпису XML дозволяє зловмисникам маніпулювати XML-документом [3].

У 2009 році в інтерфейсі SOAP від Amazon Elastic Cloud Computing (EC2) було виявлено вразливість, яка в результаті успішної атаки обгортання

підпису дозволила зловмисникам змінити повідомлення, яке було підслухано [3].

7) Інсайдерські атаки

Інсайдерську атаку здійснює законний користувач, який цілеспрямовано порушує політику безпеки, наприклад, адміністратор сервісу або співробітник компанії-клієнта. Надійна архітектура з різними рівнями доступу до хмарних сервісів допомагає запобігти зловмисній діяльності такого типу.

8) Розширені постійні загрози

APT – це атаки, які дозволяють хакерам викрадати конфіденційні дані з хмарного сховища на постійній основі, не помічаючись звичайними користувачами. Оскільки атака здійснюється тривалий час, хакери призвичаюються до заходів безпеки проти них, а після того як хакери встановлюють несанкціонований доступ до сервісу, вони можуть переміщатися по мережах центрів обробки даних і використовувати для своїх цілей мережевий трафік.

9) Викрадення облікового запису

Після отримання доступу до облікових даних користувача хакери можуть викрасти цей обліковий запис. Це може відбуватись за рахунок фішингових атак, використання шпигунського програмного забезпечення, а також зараження cookie-файлів. Після зламу хмарного облікового запису зловмисники можуть отримати конфіденційну інформацію користувача або корпоративні дані та зламати служби хмарних обчислень, як от у 2007 році співробітник Salesforce, став жертвою фішингової афери, яка призвела до викриття всіх облікових записів клієнтів компанії [3].

10) Spectre і Meltdown

Атаки Spectre та Meltdown здійснюються за рахунок використовуючи шкідливого коду JavaScript та вразливості дизайну процесорів. Внаслідок цих атак зловмисники можуть зчитувати зашифровані дані зі сховища. Як Spectre, так і Meltdown порушують ізоляцію між програмами та операційною системою, дозволяючи зловмисникам читати інформацію з ядра [3]. Забезпечення захисту у цьому випадку є складним завданням для розробників, бо не всі користувачі хмари встановлюють найновіші оновлення безпеки.

2.3 Способи покращення безпеки хмарних рішень

Постачальник хмарних послуг не може забезпечити повну безпеку в хмарі – частина відповідальності лежить і на користувачах хмари. Найкращим способом для захисту даних користувачів у хмарі зі сторони хмарного провайдера є забезпечення багаторівневого підходу до безпеки. та Розглянемо способи, якими хмарні розробники можуть забезпечити безпеку своїх хмарних рішень.

1) Поліпшити політику безпеки

Постачальники хмарних послуг повинні обмежити сферу своєї відповідальності за захист даних користувачів і операцій у хмарі, а саме повідомляти користувачів про заходи для забезпечення безпеки у хмарі, які вони здійснюють, а також надавати поради користувачам про те, які заходи безпеки їм варто застосувати зі свого боку.

2) Використання надійної автентифікації

Одним із найпоширеніших способів доступу до даних і сервісів користувачів у хмарному середовищі є викрадення паролів, тому є необхідність у використанні надійної аутентифікації, наприклад, шляхом

встановлення багатофакторної аутентифікації із використанням як статичних, так і динамічних (одноразових) паролів.

3) Захист хмарних сервісів

Для запобігання отримання хакерами несанкціонованого доступу до операцій і даних користувача через слабкі місця в хмарних сервісах, потрібно вводити обмеження доступу до цих хмарних сервісів. Розробникам варто зменшити дозволи і залишити лише необхідні для виконання конкретних операцій або залишити лише ті хмарні служби, яким користувачі найбільше довіряють.

4) Впровадження управління доступом

Розробники хмари повинні надавати користувачам можливість призначати ролі різним користувачам, а привілейованим користувачам встановлювати обсяг дозволів інших користувачів в залежності від їхніх робочих обов’язків.

5) Використання безпечних API

Тільки захищені API повинні надавати клієнтам доступ до програми, що можна забезпечити шляхом обмеження діапазону IP-адрес або надання доступу лише через корпоративні мережі. Однак цей підхід може бути важко реалізувати для загальнодоступних програм.

8) Виявлення вторгнення

Система виявлення вторгнень має керувати хмарою повністю з метою знаходження вторгнень та повідомлень про використання хмарних служб зловмисниками. Тобто система повинна забезпечувати моніторинг мережі та сповіщати про підозрілу поведінку.

7) Захист даних

На всіх етапах передачі та зберігання дані у хмарному сховищі повинні бути зашифровані:

- у стані зберігання на стороні користувача;
- у стані передачі (під час передачі від користувача до хмарного сервера);
- у стані зберігання при збереженні в хмарній базі даних.

Ще до надходження до хмари дані повинні бути зашифровані надійними алгоритмами шифрування. Для захисту від викрадення облікового запису найкраще підходять технології шифрування та токенізації, а для даних, які передаються, варто використовувати наскрізне шифрування щоб уникнути атаки «людина посередині». Оскільки дані у хмарі можуть також бути пошкоджені ненавмисно, доцільно застосувати резервне копіювання даних, яке дозволить у разі необхідності ці дані відновити.

ВИСНОВКИ ДО РОЗДІЛУ 2

Загалом, основними цілями кібератак на хмарні технології є отримання даних користувачів. У більшості випадків для здійснення атак зловмисники втручаються у комунікації між користувачами хмари та іншими службами. Це здійснюється за рахунок використання вразливостей у хмарних технологіях.

Багаторівневий підхід до безпеки – це найкращий спосіб захистити дані користувачів у хмарі. Одним з найнадійніших способів захисту даних є використання шифрування на всіх етапах передачі та зберігання даних: на стороні користувача, під час передачі даних від користувача до хмарного сервера та при збереженні у хмарному сховищі.

Однак навіть система найкращого хмарного провайдера має свої вразливості і потенційно може бути зламана зловмисником, що може

вплинути на безпеку зберігання даних користувача. Тому для надійного захисту даних варто використовувати шифрування на стороні клієнта, адже навіть у випадку зламу сховища зловмисники отримають лише зашифрований текст, який не несе у собі інформаційної цінності, бо розшифрувати дані може лише користувач.

3 ОСОБЛИВОСТІ ШИФРУВАННЯ ДАНИХ У СТАНІ ЗБЕРІГАННЯ

3.1 Шифрування на стороні сервера

Шифрування на стороні сервера – це шифрування даних безпосередньо у програмі або сервісі, яка їх отримує (у даному випадку – у хмарному сервісі). Сервіс шифрує дані та записує їх у своїх центрах обробки даних. Якщо користувач хоче отримати до них доступ – даний сервіс їх розшифровує для нього.

3.1.1 Шифрування на стороні сервера за допомогою ключів, керованих хмарним сервісом

При шифруванні на стороні сервера за допомогою ключів, керованих хмарним сервісом, кожен об’єкт користувача шифрується на стороні сервера унікальним ключем. Для забезпечення додаткового захисту ключ також шифрується кореневим ключем, який сервіс регулярно змінює. У більшості випадків для шифрування на стороні сервера застосовується один із найнадійніших блочних шифрів – 256-бітний стандарт шифрування (AES-256) GCM.

3.1.2 Шифрування на стороні сервера за допомогою ключів, які зберігаються у службі керування ключами

Служба керування ключами (KMS) надає додатковий захист для уникнення несанкціонованого доступу до об’єктів у хмарному сервісі. Це здійснюється за рахунок того, що KMS надає контрольний журнал, який показує, коли і ким використовувався ключ KMS. Користувач також може створювати ключі, якими він буде керувати, або використовувати ключі, керовані хмарним сервісом (рисунок 3.1).

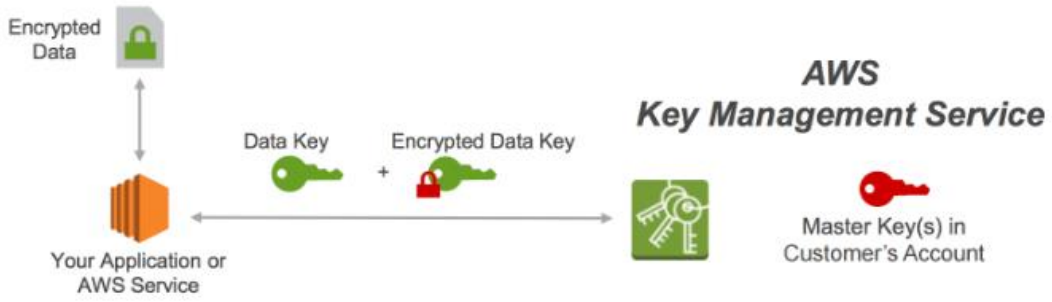

Рисунок 3.1 – Шифрування на стороні сервера за допомогою ключів, які зберігаються у службі керування ключами [5]

3.1.3 Шифрування на стороні сервера за допомогою ключів, наданими клієнтом

При використанні шифрування за допомогою ключів, наданих клієнтом, користувач керує ключами шифрування, а хмарний сервіс керує самим шифруванням під час запису даних до своїх сховищ та розшифруванням цих даних для надання до них доступу користувачу (рисунок 3.2).

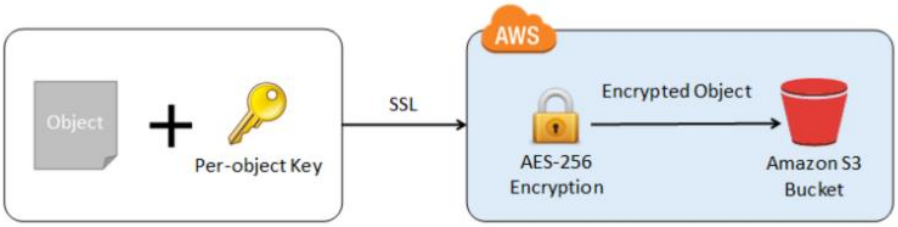

Рисунок 3.2 – Шифрування на стороні сервера за допомогою ключів, наданих клієнтом [5]

3.2 Шифрування на стороні клієнта

Шифрування на стороні клієнта – це локальне шифрування даних для забезпечення їхньої безпеки під час передачі цих даних до хмарного сервісу.

Сервіс отримує зашифровані дані та не відіграє ролі в їх шифруванні чи дешифруванні.

Для шифрування на стороні клієнта використовуються наступні методи:

- Використовувати ключ, що зберігається у службі керування ключами.
- Використовувати ключ, керований клієнтом.

3.2.1 Використання ключа, збереженого у службі керування ключами

Ключ KMS використовується для шифрування даних на стороні клієнта перед завантаженням їх до хмарного сервісу, або для дешифрування даних при їх отриманні із сервісу.

Під час завантаження об’єкту використовується ідентифікатор ключа KMS. Клієнт спочатку надсилає запит до KMS на новий симетричний ключ шифрування, який він хоче застосувати для шифрування даних свого об’єкту. Потім KMS повертає дві версії випадково згенерованого ключа:

- Відкрита текстова версія ключа, яку клієнт використовує для шифрування даних об’єкта.
- Блок шифру цього ж ключа, який клієнт завантажує у хмарний сервіс як метадані об’єкта.

Для кожного завантаженого об’єкта клієнт отримує унікальний ключ. Для того, щоб отримати свої дані із сервісу, клієнт завантажує зашифрований об’єкт із хмарного сервісу разом із версією шифру ключа, який зберігається як метадані об’єкта, а потім клієнт надсилає блок шифру в KMS, щоб отримати відкриту текстову версію ключа даних для подальшого розшифрування даних об’єкта [6].

3.2.2 Використання ключа, керованого клієнтом

Для шифрування даних ключем, керованим клієнтом, використовується кореневий ключ, який зберігається у застосунку для шифрування даних на стороні клієнта безпосередньо у користувача.

Ключі клієнта та незашифровані дані не надсилаються до хмарного сервісу, тому важливим є вміння користувача безпечно керувати своїми ключами шифрування, адже при втраті ключа він не зможе розшифрувати свої дані.

Під час завантаження об’єкта користувач використовує кореневий ключ, який генерується випадковим чином, щоб зашифрувати ключ шифрування.

Це відбувається наступним чином:

1) Користувач генерує одноразовий симетричний ключ шифрування і використовує його для шифрування даних одного об’єкту. Для кожного об’єкту клієнт створює окремий ключ.

2) Клієнт шифрує ключ шифрування за допомогою кореневого ключа і завантажує зашифрований ключ і його опис як частину метаданих об’єкта. Далі він використовує цей опис для визначення, який кореневий ключ на стороні клієнта використовувати для розшифрування.

3) Користувач завантажує зашифровані дані до хмарного сховища і зберігає зашифрований ключ як метадані у хмарному сховищі.

Для отримання даних клієнт завантажує зашифрований об’єкт зі сховища, а за допомогою опису метаданих об’єкта клієнт визначає, який кореневий ключ використовувати для розшифрування ключа даних [6]. Спочатку використовується кореневий ключ для розшифрування ключа даних, потім – ключ даних для розшифрування об’єкта.

Кореневий ключ на стороні клієнта, який надається користувачем, може бути або симетричним ключем шифрування, або парою публічним і приватним ключами.

ВИСНОВКИ ДО РОЗДІЛУ 3

Для надійності шифрування даних, які знаходяться у стані зберігання, окрім використання надійного алгоритму шифрування, не менш важливим є надійне генерування та зберігання ключа. У випадку шифрування на стороні сервера ключ може бути згенерований безпосередньо сервісом, службою керування ключами або ж наданий клієнтом. При використанні ключа, згенерованого та збереженого у хмарному сервісі, якщо буде здійснено злам системи та отримано доступ до ключа шифрування зловмисником, дані користувача можуть бути ним розшифровані. Для запобігання такої ситуації надійнішим є використовувати ключ, який зберігається у системі керування ключами або надається клієнтом, адже він таким чином буде відокремлений від самого сервісу, що ускладнить злочиннику отримання доступу до даних користувача.

Шифрування на стороні клієнта, у свою чергу, для генерування та зберігання ключа використовує або службу керування ключами, або ж використовує ключ, наданий користувачем. У другому варіанті небезпека полягає в тому, що користувач може загубити ключ, що зробить неможливим розшифрування його даних. Служба керування ключами, у свою чергу, має необхідні засоби для керування ключами.

4 ОСОБЛИВОСТІ ВИКОРИСТАННЯ ШИФРУВАННЯ НА СТОРОНІ КЛІЄНТА

Шифрування на стороні клієнта — це криптографічна техніка шифрування даних на стороні відправника перед тим як вони будуть відправлені на сервер до служби хмарного сховища. Шифрування на стороні клієнта полягає у використанні ключа шифрування, який є недоступним для постачальника послуг, що ускладнює або робить неможливим для постачальників послуг розшифрування розміщених даних. Таким чином забезпечується високий рівень конфіденційності.

У своїй основі шифрування на стороні клієнта використовує симетричні алгоритми шифрування. На даний момент стандартом і найнадійнішим із таких алгоритмів визнано AES-256, який і використовується у більшості відповідних програмних засобах. Якщо необхідно забезпечити спільний доступ до даних у сховищі із забезпеченням конфіденційності додатково використовуються асиметричні методи шифрування.

Для прикладу, на рисунках 4.1, 4.2 зображено принцип шифрування та дешифрування даних на стороні клієнта у сервісі Amazon.

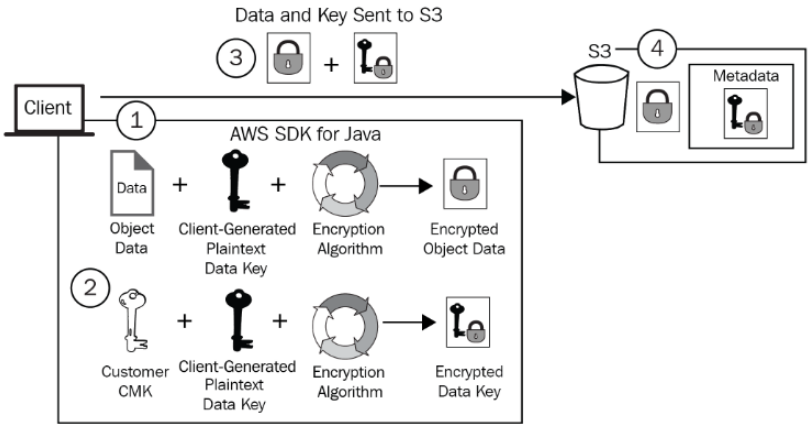

Рисунок 4.1 – Принцип шифрування на стороні клієнта у сервісі Amazon [7]

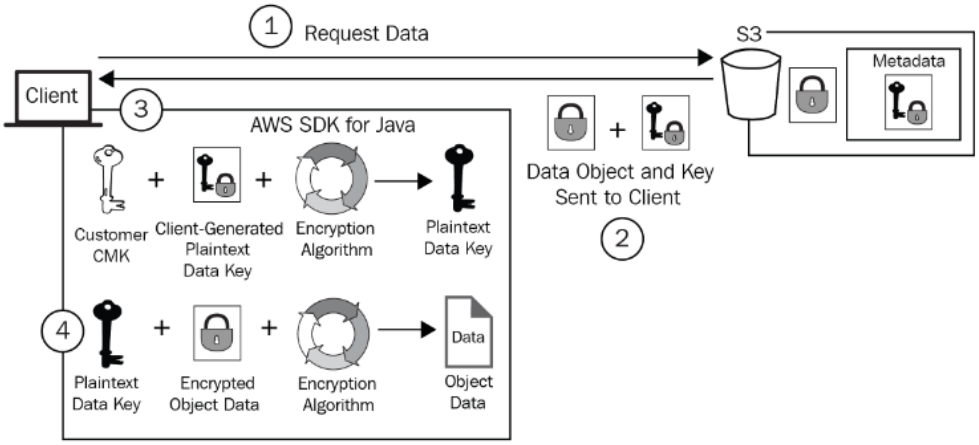

Рисунок 4.2 – Принцип дешифрування на стороні клієнта у сервісі Amazon [7]

Шифрування на стороні клієнта гарантує, що перегляд даних та файлів, які зберігаються в хмарі, може відбуватися лише на стороні клієнта. Конфіденційність даних додатково забезпечується за рахунок шифрування на кожному сервері-посереднику між початковим сервером та сервером призначення. Внаслідок цього запобігається втрата даних і розголошення файлів користувача.

На даний момент забезпечення конфіденційності даних є актуальною проблемою, тому різні наукові установи надають розробникам велику підтримку щодо застосування шифрування на стороні клієнта. Прикладами хмарних служб, які використовують шифрування на стороні клієнта, є Mega, pCloud, Icedrive.

Шифрування на стороні клієнта можна використовувати для створення платформ з нульовим знанням, які використовують деякі хмарні сервіси. Такі платформи на даний момент є хорошим рішенням безпеки. Також для безпеки файлів користувача важливим є яке саме хмарне сховище використовується.

4.1 Застосування шифрування на стороні клієнта у поєднанні із шифруванням на стороні сервера

Шифрування на стороні сервера є корисним в плані захисту даних клієнта від третіх сторін. Однак, оскільки при використанні шифрування на стороні сервера дані шифруються лише після відправки їх на сервер, є загроза, що дані у незахищеному вигляді будуть перехоплені в процесі передачі на сервер. Іншим недоліком є той факт, що захист даних зберігається за провайдером хмарних послуг, а не самим клієнтом, що може впливати на використання даних клієнта самим провайдером.

Шифрування на стороні клієнта в цьому плані запобігає обом із перелічених загроз, адже дані будуть відправлені уже в зашифрованому вигляді, тож навіть при перехопленні зловмисник отримає лише зашифроване повідомлення й не матиме можливості його розшифрувати, адже ключ зберігається лише у клієнта. З цієї ж причини повідомлення не зможе розшифрувати й сам провайдер, що сприяє забезпеченню доступу до своєї інформації лише користувачем.

Якщо використовувати шифрування на стороні клієнта у поєднанні із шифруванням на стороні сервера, це забезпечить більшу надійність зберігання даних, адже дані будуть зашифровані двічі, що унеможливить як використання даних провайдером, так і отримання доступу до даних третіми сторонами.

4.2 Шифрування на стороні клієнта на основі наскрізного шифрування

Наскрізне шифрування (End-To-End Encryption) – це криптографічна техніка шифрування даних та системи комунікації, при якій доступ до файлів можуть отримати лише дві сторони, які встановили між собою зв’язок. Дані шифруються на пристрої відправника, і лише конкретний отримувач може їх

розшифрувати. Таке шифрування гарантує безпечний обмін інформацією між кінцевими точками та запобігає отриманню доступу до файлів чи розмови телекомунікаційними та інтернет-провайдерами, адже вони не мають ключів, необхідних для дешифрування файлів.

Особливості реалізації відрізняються в залежності від сервісу, однак в загальному стратегія полягає у використанні пари ключів.

Принцип шифрування полягає у тому, що ключі для шифрування та дешифрування даних знаходяться на кінцевих точках. Публічні ключі використовуються для шифрування інформації, приватні ключі – для розшифрування. Кожне повідомлення шифрується на пристрої відправника публічним ключем, який надав отримувач, і може бути розшифроване лише приватним ключем на пристрої отримувача (рисунок 4.3). Кількість серверів та мереж на шляху проходження повідомлення не впливає на безпеку відправлених даних, адже лише отримувач може його розшифрувати.

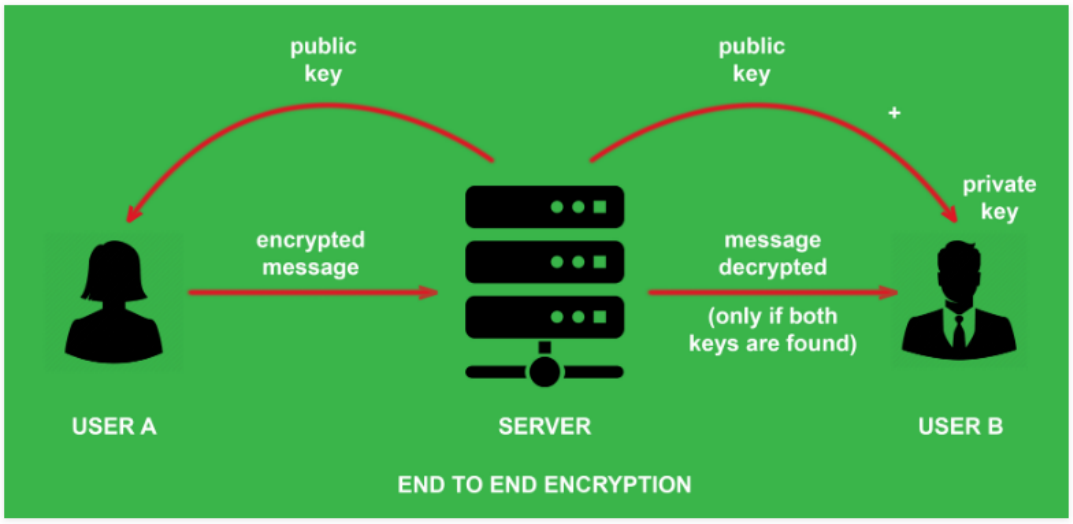

Рисунок 4.3 – Принцип наскрізного шифрування [8]

У хмарних технологіях даний метод захисту інформації доцільно використовувати якщо потрібно ділитись файлами із іншими користувачами, але при цьому запобігти прочитанню файлів провайдером.

Однак цей тип захисту може бути зламаний кількома способами:

1) Зловмисник може імітувати одержувача повідомлення (між торгівлею ключами або шляхом заміни його відкритим ключем одержувача), тому повідомлення шифруються ключем, відомим нападнику [8]. Даний спосіб атаки відомий як «людина посередині».

2) Може бути зламаний безпосередньо комп'ютер кожного клієнта, щоб отримати його або її криптографічний ключ та прочитати розшифровані повідомлення.

3) Бекдори – засоби обходу шифрування, які навіть самі компанії часто впроваджують (навмисно чи ненавмисно). У 2013 році Едвард Сноуден оприлюднив інформацію, яка показала, що Skype мав бекдор, який дозволяв Microsoft передавати повідомлення своїх користувачів АНБ попри те, що повідомлення були офіційно наскрізно шифровані [8].

Загалом, наскрізне шифрування можна розглядати як особливе використання шифрування на стороні клієнта з метою обміну повідомленнями. У стандартному випадку при шифруванні на стороні клієнта дані шифруються локально перед тим як передаватись іншій стороні, тобто користувач – єдиний, хто має доступ до вмісту своїх файлів. У випадку наскрізного шифрування доступ до вмісту файлів можуть отримати кінцеві точки.

ВИСНОВКИ ДО РОЗДІЛУ 4

Шифрування на стороні сервера є хорошим рішенням для запобігання отримання даних третіми сторонами, однак повне забезпечення конфіденційності досягається тільки при його поєднанні із шифруванням на стороні клієнта, що дає високий захист даних.

Використання класичного шифрування на стороні клієнта гарантує отримання доступу до даних лише користувачем, однак якщо необхідно

надавати доступ до даних, збережених у хмарному сервісі, іншим користувачам, варто використовувати шифрування на стороні клієнта із використанням наскрізного шифрування, при якому застосовуються асиметричні методи шифрування. Це дає змогу використовувати хмарний сервіс як сховище, однак з неможливістю ним отримувати доступ до інформації, адже розшифрувати вміст можуть тільки кінцеві точки, а ключі для шифрування та дешифрування є різними.

5 АНАЛІЗ ПРОГРАМНИХ ІНСТРУМЕНТІВ ДЛЯ ШИФРУВАННЯ ДАНИХ НА СТОРОНІ КЛІЄНТА

Захист, забезпечення конфіденційності та доступ до даних безпосередньо пов’язані із програмними засобами, якими здійснюється шифрування. Розглянемо особливості програмних інструментів, що відповідають за шифрування даних на стороні клієнта.

5.1 Огляд програмного інструменту Cryptomator

Cryptomator - це міжплатформний програмний інструмент з відкритим кодом, який забезпечує шифрування даних на стороні клієнта для хмарного сервісу (рисунок 5.1).

Призначення:

Забезпечення конфіденційності при збереженні файлів до хмарних сховищ.

Даний програмний засіб зменшує ризик того, що доступ до збережених у хмарі даних отримає постачальник хмарних послуг або треті сторони. Для того, щоб прочитати або редагувати вміст файлу, потрібно знати ключ шифрування, який провайдеру невідомий. Це стосується як вмісту файлів, так і імен файлів.

Чим Cryptomator не є:

- Cryptomator не захищає файли на локальному комп’ютері.
- Cryptomator не є повною заміною іншим інструментам шифрування, адже мета-інформація ним не шифрується.
- Cryptomator не забезпечує захист, якщо під час роботи із зашифрованими файлами програми створюють їхні резервні копії, адже такі файли Cryptomator не розпізнає, тому вони можуть залишатися у сховищі [9].

- Cryptomator не забезпечує захист, якщо локальний комп’ютер уражений шкідливим програмним забезпеченням, яке зчитує введені паролі та вміст файлів (наприклад, файли у розблокованому сховищі) [9].

Для того, щоб уникнути таких ризиків, потрібне повне шифрування диску, вчасна інсталяція оновлень програмного забезпечення та системи в цілому і використання антивірусного програмного забезпечення.

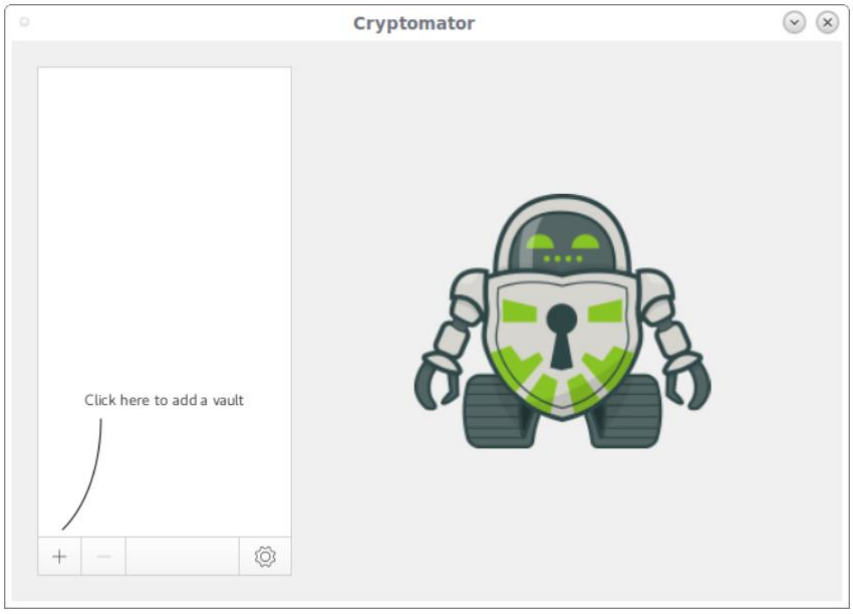

Рисунок 5.1 – Інтерфейс програмного засобу Cryptomator [9]

Принцип роботи:

Cryptomator зберігає конфіденційні дані користувача на віртуальному диску під назвою Vault («підземне сховище»). Цей диск можна використовувати як звичайний USB-флеш-накопичувач або жорсткий диск. Все, що вноситься до сховища, буде зашифровано та захищено паролем (рисунок 5.2). Cryptomator шифрує всі дані, які поміщаються на диск, «на льоту». Можна зберігати ці «підземні сховища» (vaults) на своєму локальному жорсткому диску, у хмарі або на будь-якому зовнішньому жорсткому диску чи USB-накопичувачі [11]. Якщо vault був розміщений у

хмарі, як, наприклад, Google Drive або Dropbox, тоді можна отримати доступ до даних з усіх пристроїв з підтримкою Інтернету за допомогою настільного додатку Cryptomator або мобільного клієнта.

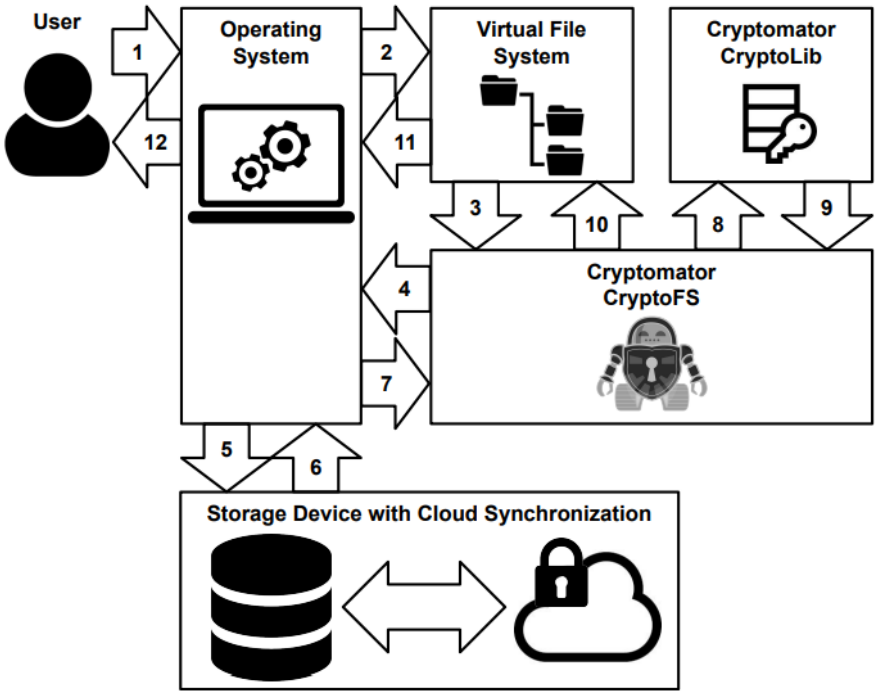

Рисунок 5.2 – Принцип читання та дешифрування даних, збережених у Cryptomator [11]

Існує деяка мета-інформація, яку Cryptomator не шифрує для забезпечення синхронізації, а саме:

- доступ, зміна та час створення файлів і папок;
- кількість файлів і папок у vault та в папках;
- розмір збережених файлів.

Особливості Cryptomator

Cryptomator використовує шифрування на стороні клієнта. Дані спочатку шифровуються, а потім синхронізуються з онлайн-хмарною службою зберігання, щоб на комп’ютері не залишалося незашифрованих

даних. Даний програмний засіб шифрує як файли, так і назви файлів використовуючи алгоритм AES-256.

Порівняно з іншими утилітами для шифрування диска, Cryptomator шифрує кожен файл окремо. Отже, якщо користувач відредагував файл, не потрібно перешифровувати та синхронізувати весь вміст, а лише файл, який був змінений (рисунок 5.3). Багато інших програмних засобів шифрування не мають такої можливості.

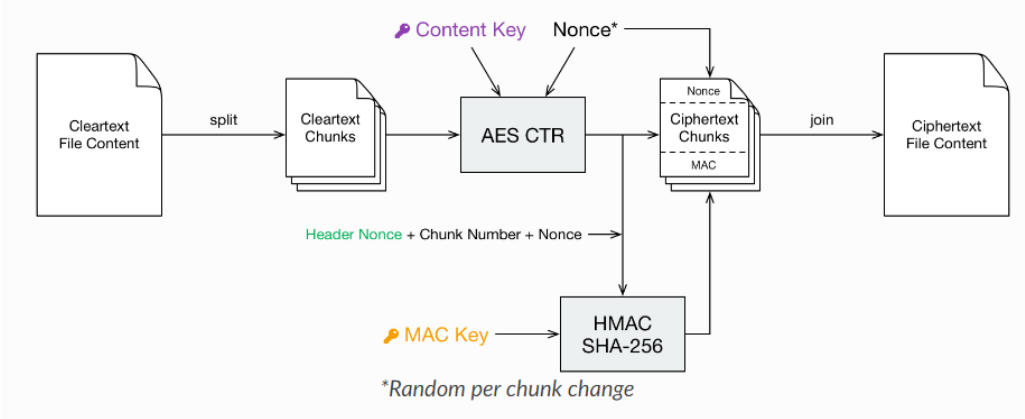

Рисунок 5.3 – Шифрування вмісту файлу у Cryptomator [11]

Переваги:

1) Даний інструмент є простим у використанні.

2) Підтримує будь-якого постачальника хмарних сховищ, який синхронізується з локальним каталогом.

3) Дозволяє отримати доступ до файлів з усіх пристроїв користувача.

4) Є безкоштовним.

Недоліки:

Cryptomator не підключається безпосередньо до служби хмарного сховища. Якщо необхідно використовувати Cryptomator для шифрування

хмарного сховища, потрібно встановити клієнт синхронізації (наприклад, Dropbox, Google Drive тощо) або підключити сервер як спільну мережу.

ОС, для яких доступна програма: Windows, Mac, Linux, iOS та Android.

5.2 Огляд програмного інструменту Boxcryptor

Boxcryptor – це програмний інструмент, який захищає дані користувачів та компаній при їхньому розміщенні у хмарних сховищах. Він шифрує дані безпосередньо на пристрої, і вже потім ці дані завантажуються до хмарного сховища (рисунок 5.4).

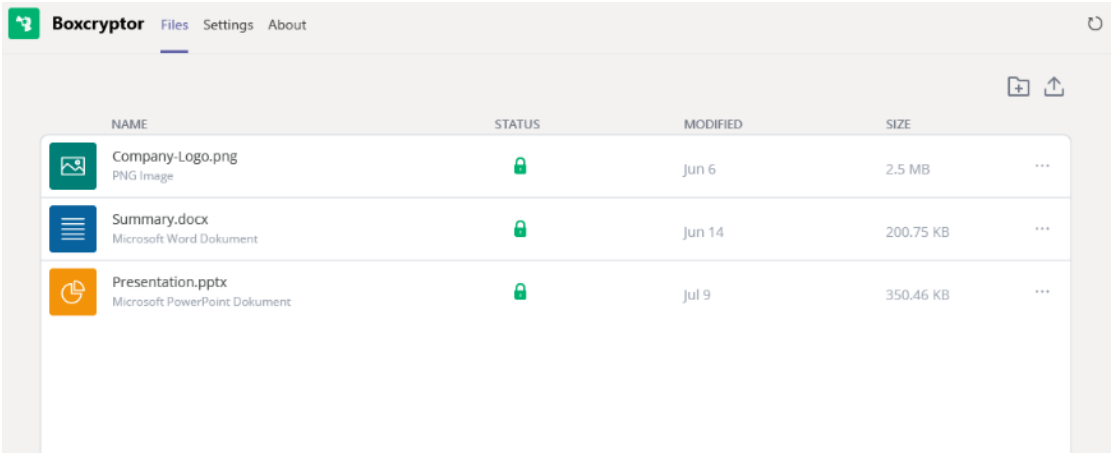

Рисунок 5.4 – Інтерфейс програмного засобу Boxcryptor

Призначення:

Boxcryptor надає додатковий рівень безпеки для хмарних сховищ за рахунок шифрування файлів локально на пристрої. Оскільки Boxcryptor з самого початку був оптимізований для хмари, доступ до файлів може бути спільним, а шифрування здійснюється для кожного файлу, тобто кожен файл шифрується незалежно від інших [12]. Крім того, підтримуються такі функції як історія файлів або вибіркова синхронізація.

Чим Boxcryptor не є:

- Boxcryptor не є хмарним сховищем. Це програмне забезпечення відповідає за надання додаткового рівня безпеки до необхідного хмарного сховища, але не зберігає дані користувача. За зберігання та керування файлами відповідає хмарний провайдер.
- Boxcryptor не займається синхронізацією файлів із хмарою. За це відповідає хмарний провайдер, тому користувачу потрібно встановити на свій пристрій програмне забезпечення свого хмарного провайдера.
- Boxcryptor не призначений для захисту довільних хмарних сервісів. Він може шифрувати лише файли, які зберігаються у хмарі. Наприклад, такі служби, як Google Docs або Evernote, зберігають дані у базах даних на своїх серверах, але із локально збереженими файлами не працюють [12].

Принцип роботи:

Будь-яка конфіденційна та чутлива інформація шифрується за допомогою ключа, невідомого провайдеру. Тільки публічні ключі є доступними для провайдера.

Паролі, ключі паролів і ключі файлів знаходяться лише на пристроях користувачів і не передаються. Ключі користувача, ключі групи та ключі компанії зберігаються у зашифрованому вигляді на сервері Boxcryptor, а операції шифрування відбуваються на пристрої користувача.

Для дешифрування використовується ключ пароля користувача, який потрібен для розблокування приватного ключа та ключа обгортання, які, у свою чергу, потрібні для розблокування всіх інших ключів у системі, таких як ключі AES, ключі файлів, групові ключі та ключі членства. Ключ пароля зберігається тільки на пристрої користувача. Попри те, що сервер Boxcryptor зберігає ключі для всіх користувачів, він не має доступу до вмісту файлів,

адже конфіденційні ключі вже отримані в зашифрованому вигляді. Єдиними типами ключів, які сервер зберігає у відкритому тексті, є публічні ключі.

Як і у випадку Cryptomator, усі файли шифруються окремо, тому при зміні якогось файлу буде перешифровано та синхронізовано не весь вміст, а тільки даний файл.

Особливості Boxcryptor

Для шифрування Boxcryptor використовує поєднання асиметричного шифрування RSA і симетричного шифрування AES (рисунок 5.5). При створенні файлу випадковим чином генерується спеціальний унікальний ключ файлу, який використовується для шифрування та розшифрування вмісту файлу.

Спочатку на пристрої користувача пароль декілька разів хешується і встановлюється по-різному з метою отримання хеша пароля і ключа пароля, далі хеш пароля передається до серверів Boxcryptor, хешується там вдруге, після чого зберігається результат [12]. Внаслідок цього тільки пристрій користувача містить пароль, і Boxcryptor не може отримати доступ до хмарного вмісту. Тому як Boxcryptor, так і Cryptomator є постачальниками з нульовим рівнем знань.

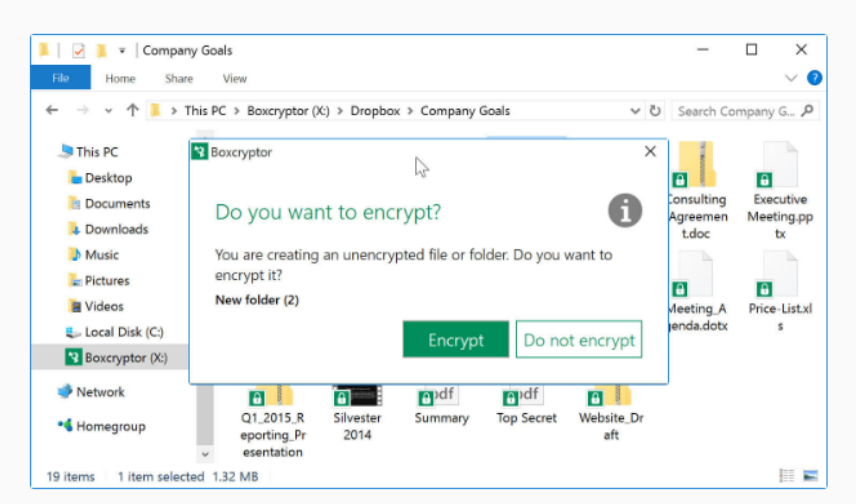

Рисунок 5.5 – Виконання шифрування у Boxcryptor

Шифрування:

Спочатку генерується випадковий ключ файлу, після чого необхідні дані зашифровуються цим ключем. Далі ключ файлу шифрується публічним ключем користувача, а ключ зашифрованого файлу зберігається разом із зашифрованими даними у файлі, який зашифрували. У випадку якщо одразу декілька користувачів мають доступ до файлу, то ключ файлу шифрується декілька разів публічними ключами різних користувачів, після чого у зашифрованому файлі зберігається кожен результат [12].

Дешифрування:

Для дешифрування зашифрованого ключа файлу використовується приватний ключ користувача, а самі дані розшифровуються цим ключем файлу.

Застосовані алгоритми:

- AES з довжиною ключа 256 біт, CBC (Cipher Block Chaining) і заповненням PKCS7.
- RSA з довжиною ключа 4096 біт і заповненням OAEP.

Переваги:

- Шифрує дані з можливістю їхнього розшифрування одразу декількома користувачами.
- Дозволяє отримати доступ до файлів з усіх пристроїв користувача.

Недоліки:

Самостійно не синхронізується із хмарним сервісом.

ОС, для яких доступна програма: Windows, macOS, iOS, Android (Linux з Boxcryptor Portable і обмеженою функціональністю).

5.3 Порівняння програмних інструментів Cryptomator та Boxcryptor

У цьому розділі порівняємо такі критерії програмних засобів Cryptomator та Boxcryptor як підтримувані платформи, функції відповідних мобільних додатків, а також загальні функції, такі як спільний доступ до файлів.

Підтримувані постачальники хмарних послуг:

Cryptomator: хмарні сховища на основі Dropbox, Google Drive, OneDrive і WebDAV.

Boxcryptor: Dropbox, Google Drive, OneDrive, iCloud, SharePoint, хмарні сховища на основі WebDAV.

Підтримувані платформи:

Cryptomator: Windows, Android, macOS, iOS, Linux.

Boxcryptor: Windows, Android macOS, iOS, (Linux з Boxcryptor Portable і обмеженою функціональністю).

Спільний доступ до файлів:

Cryptomator: Єдиним способом, щоб поділитися файлами та папками з іншими користувачами, є надати їм доступ до сховища, тобто свій пароль. При цьому сховище не може бути відкрите одночасно для двох людей. Люди, які не використовують Cryptomator, неможливий.

Boxcryptor: Доступ до файлів може бути отриманий користувачами, у яких не встановлений Boxcryptor. Ця функція з’явилась у 2016-му році, коли був розроблений наскрізний зашифрований сервіс передачі файлів Whispy, який можна використовувати у браузері і який є інтегрованим у Boxcryptor [13]. Також не обов’язково повідомляти користувачам свій пароль, адже для того, щоб безпечно ділитися окремими файлами, можна використовувати електронну адресу.

Акаунт:

Cryptomator: Працює без облікового запису. Оскільки даний програмний засіб не використовує спільний доступ до файлів, дана опція не є необхідною.

Boxcryptor: Необхідно створити обліковий запис. Це пов’язано із тим, що для спільного використання файлів і папок необхідно використовувати електронну адресу. Для того, щоб поділитися певним файлом, потрібно зашифрувати ключ цього файлу публічним ключем іншого користувача. При реєстрації у Boxcryptor реєструється публічний ключ користувача, який потім зберігається разом з усіма іншими публічними ключами на сервері Boxcryptor [13]. Таким чином інші користувачі можуть отримати публічний ключ необхідного користувача, з яким хочуть поділитись файлом.

Додаткова безпека облікового запису:

Cryptomator: немає.

Boxcryptor: двофакторна аутентифікація (2FA) на всіх платформах використовуючи програму Authenticator (TOTP). У Windows та macOS підтримуються ключі безпеки стандарту WebAuthN – наприклад, YubiKeys [13].

Доступні мови:

Cryptomator: англійська, німецька, російська, голландська, китайська.

Boxcryptor: англійська, німецька, російська, італійська, іспанська, французька.

Довіра:

Cryptomator: Є програмним забезпеченням із відкритим вихідним кодом. Оскільки код знаходиться у вільному доступі, будь-які зміни видно спільноті.

Boxcryptor: Є власністю, тому спільнота не може побачити код. Тому, наприклад, якщо постачальник вбудує бекдори, спільнота їх не зможе виявити.

Цінова модель:

Cryptomator: Настільний додаток є безкоштовним. За мобільний додаток стягується одноразова фіксована сума 9,99 євро (iOS або Android).

Boxcryptor: Існують різні підписки:

- Безкоштовна версія: шифрування Dropbox за допомогою Boxcryptor для Windows та Android;
- Повна версія для приватного користування: 36 євро на рік або 27 євро на рік за 3-річну підписку;
- Бізнес-версія для самозайнятих осіб: 72 євро на рік або 54 євро на рік з 3-річною підпискою;
- Знижка для студентів: 25%.

Хмарне шифрування для команд і компаній:

Cryptomator: Є можливість для компаній шифрувати власні сервери використовуючи Cryptomator Server. Доступні функції:

- керування правами користувачів;
- журнали аудиту;
- зашифровані хмарні резервні копії.

Boxcryptor: Доступні функції:

- керування правами користувачів;
- функції аудиту;
- налаштування політики.

5.4 Рекомендації щодо вибору програмного інструменту

Наступна таблиця наводить порівняльні характеристики розглянутих програмних засобів (табл.2).

Таблиця 5.1 – Порівняння програмних інструментів Cryptomator та Boxcryptor

	Cryptomator	Boxcryptor
Підтримувані постачальники хмарних послуг	Dropbox, Google Drive, OneDrive, WebDAV	Dropbox, Google Drive, OneDrive, iCloud, SharePoint, WebDAV
Підтримувані платформи	Windows, Android, macOS, IOS, Linux	Windows, Android, macOS, IOS (Linux з Boxcryptor Portable і обмеженою функціональністю)
Спільний доступ до файлів	–	+
Необхідність створення акаунта	–	+
Додаткова безпека	–	Двофакторна аутентифікація (2FA)
Доступні мови	Англійська, німецька, російська, голландська, китайська	Англійська, німецька, російська, італійська, іспанська, французька
Довіра, відкритий вихідний код	+	–
Вартість	Безкоштовно	Існують різні підписки. Безкоштовна – з обмеженою функціональністю
Можливості для команд	▪ Керування правами користувачів;	▪ Керування правами користувачів;

	▪ Журнал аудиту; ▪ Зашифровані хмарні резервні копії.	▪ Функції аудиту; ▪ Налаштування політики.

Порівнюючи Cryptomator та Boxcryptor та беручи до уваги відкритий вихідний код, Cryptomator є надійнішим. Можна переглянути логіку програми та переконатися у безпечності її використання. Boxcryptor, у свою чергу, не надає у загальний доступ вихідний код, що не дає користувачам змоги переконатися у принципі роботи. Однак Boxcryptor, на відміну від Cryptomator, надає можливість спільного доступу до файлів, тому для роботи в команді доцільніше використовувати Boxcryptor.

Беручи до уваги вартість, Boxcryptor у безкоштовній версії забезпечує лише базову функціональність, а для отримання більших можливостей потребує платної підписки. Якщо користувач хоче мати власне надійне сховище і не збирається працювати в команді, тоді для цієї цілі доцільніше використовувати Cryptomator, адже він має всі необхідні функції та при використанні десктопної версії є безкоштовним.

ВИСНОВКИ ДО РОЗДІЛУ 5

У даному розділі було розглянуто програмні засоби шифрування даних на стороні клієнта Cryptomator та Boxcryptor. Було визначено переваги та недоліки використання кожного з них. Головна відмінність полягає у методах шифрування. Boxcryptor надає можливість спільного доступу до даних за рахунок використання асиметричних методів шифрування на основі приватних та публічних ключів. У Cryptomator такої можливості немає, тому єдиним способом для спільного доступу є поділитись симетричним ключем шифрування. Однак Cryptomator, на відміну від Boxcryptor, має відкритий вихідний код, тому до даного засобу у користувачів є довіра.

6 АНАЛІЗ ХМАРНИХ СЕРВІСІВ, ЯКІ ВИКОРИСТОВУЮТЬ ШИФРУВАННЯ НА СТОРОНІ КЛІЄНТА

Існують також хмарні сховища, мають функціональність для шифрування даних на стороні клієнта. Розглянемо їх.

6.1 Огляд хмарного сервісу pCloud

pCloud – це хмарний міжплатформний сервіс зберігання даних, до якого можна отримати доступ через будь-який веб-браузер (рисунок 6.1).

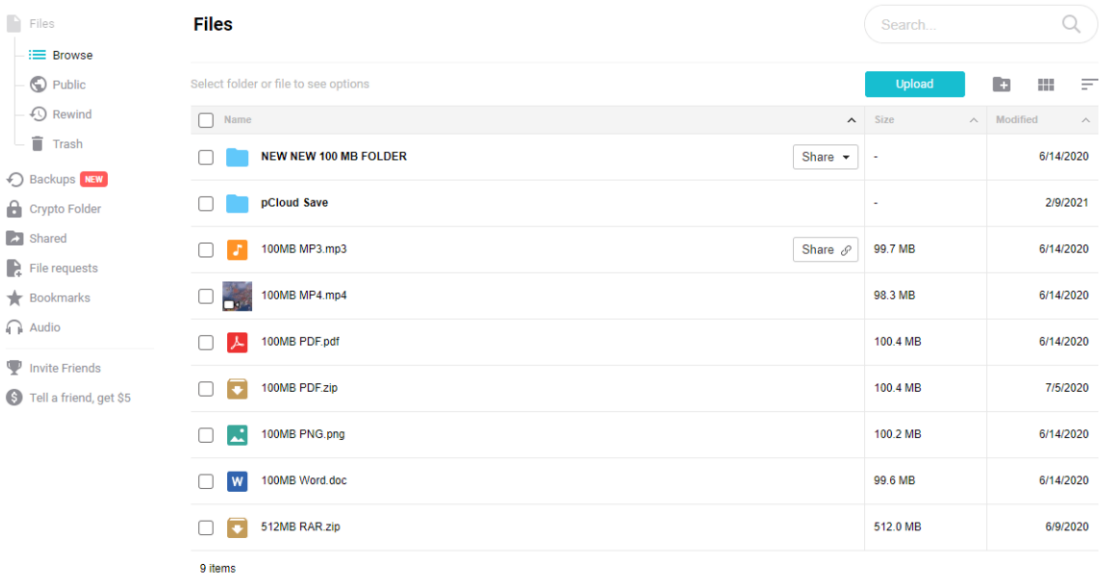

Рисунок 6.1 – Інтерфейс сервісу pCloud

Шифрування: pCloud використовує як шифрування на стороні клієнта, так і шифрування на стороні сервера. Оскільки дані, зашифровані перед відправкою на сервер, не можна безпосередньо відтворювати при використанні сервісу, було вирішено зробити наступним чином:

- Шифрування на стороні клієнта використовується для захисту важливих файлів.

- Шифрування на стороні сервера використовується для зберігання музики, відео і подібних файлів, щоб їх можна було швидко відтворити.

pCloud використовує спеціальну папку Crypto Folder, до якої можна завантажувати свої файли. Ці файли потім шифруються користувачем, і тільки він має до них доступ. Якщо користувач забуде пароль до папки Crypto, всі файли будуть втрачені, оскільки без нього неможливо відкрити папку.

Для захисту даних діє спеціальна опція pCloud Crypto – дані шифруються на пристрої користувача, а на сервери завантажуються лише зашифровані дані. Ключ шифрування зберігається лише в користувача.

Приватні ключі користувачів: 4096-бітний RSA.

Ключі для кожного файлу та кожної папки: 256-розрядний AES-256.

Захист файлів під час їх передачі з комп’ютера на сервери: шифрування TLS / SSL.

Кроки для здійснення шифрування даних на стороні клієнта:

1) Встановити pCloud Drive і активувати Crypto Folder – безкоштовний варіант на пробний період 14 днів або підписка 3,99$ на місяць;

2) pCloud Crypto запитає пароль і дасть підказку, яким чином його краще обрати для надійності;

3) Створиться папка Crypto Folder, у яку можна додати файли, які потрібно зашифрувати;

4) Файли у зашифрованому вигляді збережуться на сервер, потрібно їх заблокувати, оскільки інакше їх буде видно на пристрої;

5) Переконатися, що файли заблоковані. Відкрити pCloud Drive – папки Crypto Folder не повинно бути видно, що означає, що файли у безпеці навіть якщо щось станеться із пристроєм.

Для того, щоб отримати доступ до файлів, потрібно натиснути «Unlock Crypto», ввести пароль і підтвердити. Після цього файли знову у доступі користувача.

Синхронізація:

Файли синхронізуються на комп’ютері, мобільному пристрої та через веб-додаток. Програма для комп'ютера pCloud - pCloud Drive - має додаткову опцію синхронізації файлів, яка об'єднує локальні файли на комп'ютері з pCloud Drive [16]. pCloud підтримує блокову синхронізацію, тому файли оновлюються швидко.

Зберігання файлів: мінімум 3 сервери з різним місцезнаходженням.

Операційні системи: Windows, Mac, Linux.

Переваги:

- Блокова синхронізація файлів;
- Необмежені розміри файлів;
- Резервне копіювання папки на робочому столі;
- Надійне шифрування на стороні клієнта (дослідники з Бостонського університету та багатьох інших організацій не змогли його зламати);
- Вбудований відеоплеєр для безпечного перегляду відеофайлів;
- Зберігає 5 копій файлів на різних серверах для більшої надійності.

Недоліки:

- Не рекомендується вирізати та вставляти файли, щоб завантажити їх на pCloud Drive. Drag And Drop файлів підходить для невеликих об’ємів даних. Це не рекомендується для великих файлів або великої кількості файлів. Якщо потрібно завантажити великий обсяг даних або великий файл, рекомендується додати його для

синхронізації і тільки після того, як файл буде завантажено успішно – зупинити синхронізацію [16].

- У безкоштовній версії немає немає Crypto Folder та опції для поширення файлів між командами та групами.

Ціна: Доступні наступні плани:

- Щорічна підписка: 47.88$ на користувача;
- Lifetime Premium Plus: 350$.

6.2 Огляд хмарного сервісу Sync.com

Sync.com – це хмарне сховище, яке використовує технологію «нульового знання» (рисунок 6.2).

Sync.com використовує платформу із нульовим рівнем знань. Шифрування та розшифрування даних здійснюється на стороні клієнта, що забезпечує високу конфіденційність. Ключі шифрування, які використовуються для шифрування файлів, знаходяться лише у користувача, Sync.com не має до них доступу. Пароль до облікового запису сервісу також невідомий.

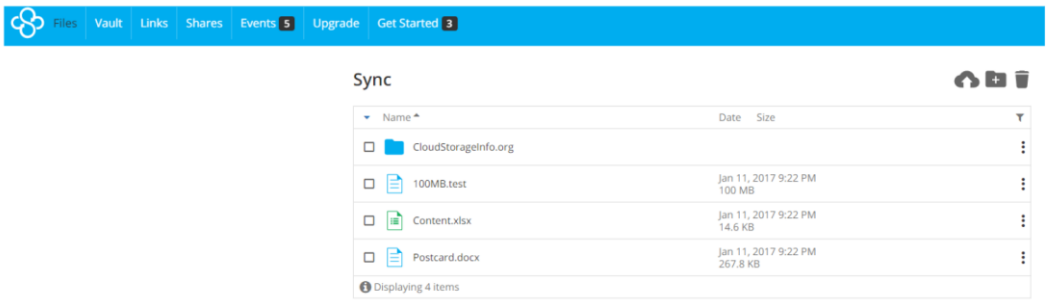

Рисунок 6.2 – Інтерфейс сервісу Sync

Sync.com також відповідає вимогам HIPAA. Тобто він підходить для компаній і фірм, які працюють із конфіденційною інформацією про пацієнтів.

Кожна установа, яка працює з даними такого типу, має використовувати мережу або хмару, яка відповідає HIPAA [17].

Шифрування:

- Sync використовує шифрування RSA-2048, AES-256, SSL і TLS.
- Шифруються як файл, так і його мета дані;
- Паролі для шифрування залишаються тільки у користувача;
- Зашифровані приватні ключі зберігаються на серверах Sync.

Синхронізація:

Sync підтримує дедуплікацію файлів, але не може підтримувати синхронізацію на рівні блоків, тому файли завантажуються довго [18]. Через шифрування на стороні клієнта синхронізація на рівні блоків є неможливою. Для того, щоб синхронізація працювала, хмарна служба повинна отримати доступ до файлів, що є неможливим в умовах забезпечення конфіденційності.

Доступна вибіркова синхронізація, для якої потрібно вибрати папки, які потрібно синхронізувати з певним пристроєм.

ОС: Windows, Mac.

Переваги:

- Є одним з найнадійніших хмарних сервісів в плані забезпечення конфіденційності.
- Хороші варіанти обміну та отримання файлів.

Недоліки:

- Робочий стіл не дозволяє синхронізувати будь-яку папку на комп’ютері;
- Не підтримує блокову синхронізацію.

Ціна: Доступні наступні плани:

- Безкоштовна версія: 5 Гб у сховищі, можливість ділитись файлами (до 5 Гб);
- Solo Basic: 8$ за місяць, 2000 Гб у сховищі, покращені можливості для спільного використання;
- Solo Professional: 20$ за місяць, 6000 Гб у сховищі, зручніша організація, покращені можливості спільного використання.

6.3 Огляд хмарного сервісу Mega

MEGA — новозеландський файлообмінник та хмарне сховище даних [20]. Вміст файлу шифрується за допомогою алгоритму AES. Шифрування відбувається у браузері. Є можливість для користувачів передавати у зашифрованому вигляді файли один одному. Для того, щоб поширити ключі до файлів, використовується схема Friend-to-Friend – між тими користувачами сервісу, які мають довіру один до одного [21]. Самі ключі у відкритому доступі не публікуються.

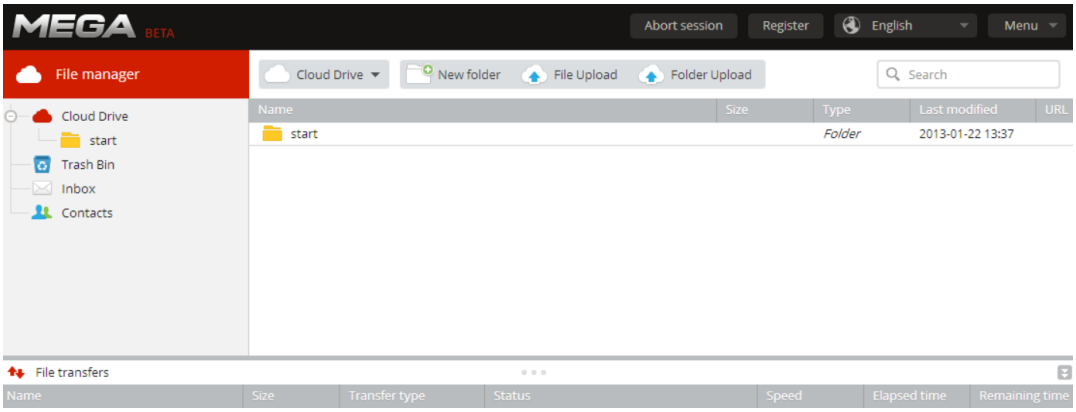

Рисунок 6.3 – Інтерфейс сервісу Mega

Шифрування:

- Для шифрування файлів використовується симетричний алгоритм AES-128;
- Сервіс не має доступу до ключів шифрування файлів.

- Mega використовує кероване користувачем шифрування, яке забезпечує автоматичне шифрування всіх даних, які передаються і зберігаються у Mega. Ключами шифрування керує користувач, тому він може контролювати, хто має доступ до його файлів за рахунок застосованого при шифруванні алгоритму RSA.

Синхронізація: Mega дає змогу створити папку синхронізації, у якій потім можна зберігати свої файли.

ОС: Windows, macOS та Linux.

Переваги:

Хороша синхронізація.

Недоліки:

- Ключ зберігається у зашифрованому вигляді на серверах Mega, але розблоковується після введення користувачем пароля. Однак немає гарантії, що сам пароль не буде приховано зчитано.
- Попри те, що шифрування відбувається на стороні клієнта за допомогою javascript, код щоразу завантажується з сервера повторно і у будь-який момент може бути підмінений без попередження, наприклад в результаті злому сервера третьою стороною [22]. Через це сервіс критикувався експертами з криптографії.

Ціна: Доступні наступні плани:

- Безкоштовна версія: 15 ГБ пам’яті.
- Платні версії: від 5 євро до 30 євро за місяць – від 400 ГБ до 16 ТБ пам’яті.

6.4 Огляд хмарного сервісу Icedrive

Icedrive – хмарний сервіс, який використовує шифрування на стороні клієнта Twofish, на відміну від більшості сервісів, які використовують протокол AES, що робить його унікальним на ринку [23]. У поєднанні з сучасним інтерфейсом користувача це є дуже корисним і безпечним рішенням для хмарного зберігання.

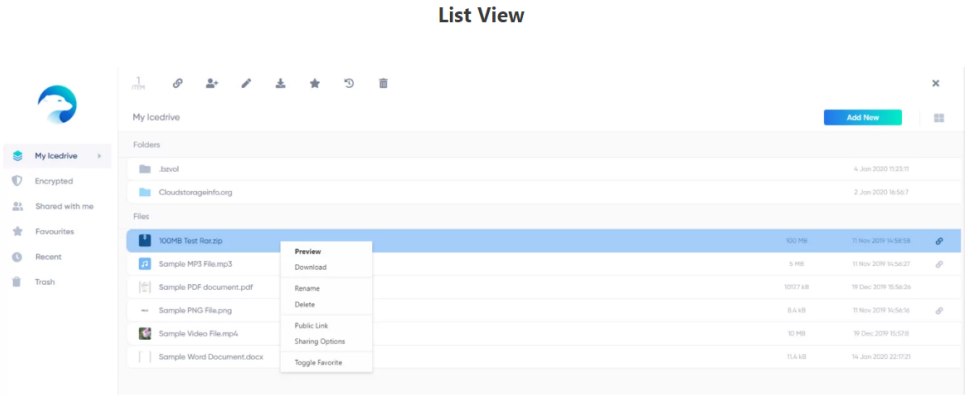

Рисунок 6.4 – Інтерфейс сервісу Icedrive

Шифрування:

Сервіс використовує шифрування Twofish, яке вважається безпечнішим та більш складним, однак для цього використовується більше ресурсів. Icedrive генерує захищені 256-бітні ключі за допомогою парольної фрази шифрування. Ключі генеруються на стороні клієнта і не передаються на сервери, тільки користувач має до них доступ. Спочатку шифрується назва файлу, потім файл розбивається на маленькі частинки. Кожна частинка шифрується за 16 раундів використовуючи збережений на стороні клієнта 256-бітний ключ. Потім ці частинки завантажуються один за одним на сервери за допомогою HTTPS, об’єднуються, і результуючий фрагмент доповнюється додатковими даними, щоб заповнити розмір блоку Twofish [24]. Внаслідок цього утворюється добре зашифрований файл і назви файлів без витоку даних, які доступні лише користувачу.

Файли шифруються як під час передачі (TLS/SSL), так і під час зберігання.

Icedrive також пропонує двофакторну аутентифікацію використовуючи Google Authenticator, FIDO/U2F та SMS-повідомлення.

ОС: Windows, macOS, Linux.

Синхронізація:

Клієнт віртуального диска встановлює на комп'ютері уявний диск, який не займає жодного місця в сховищі (крім кешу). Синхронізація можлива для будь-якої папки з комп'ютера в хмару.

Переваги:

- Надійний алгоритм шифрування

Недоліки:

- Шифрування на стороні клієнта потребує більше ресурсів, аніж аналоги.

Ціна: Доступні такі плани:

- Безкоштовний: 10 ГБ, пропускна здатність 3 ГБ/день;
- Lite: 1.83 євро, 150 ГБ, пропускна здатність 250 ГБ/місяць;
- Pro: 4.17 євро, 1 ТБ, пропускна здатність 2 ТБ/місяць;
- Pro +: 15 євро, 5 ТБ, пропускна здатність 8 ТБ/день.

6.5 Рекомендації щодо вибору хмарного сервісу

Для загального порівняння хмарних сховищ наведемо таблицю з характеристиками різних сервісів. У таблиці 6.1 наведені характеристики, на основі яких користувач в залежності від потреб може обрати необхідний програмний засіб.

Таблиця 6.1 – Порівняння характеристик програмних засобів [14]

	pCloud	Sync.com	Mega	Icedrive
Вільна пам’ять	5 GB (До 10 GB)	5 GB	15 GB	10 GB
Безкоштовний пробний період	-	-	-	-
Життєві плани	+	-	-	+
Способи оплати	PayPal, Credit card, Bitcoin	PayPal, Credit card, Bitcoin	Paysafecard, Credit card, Wire transfer, Bitcoin	PayPal, Credit card, Crypto
Відкриття у браузері	Office, PDF, Music, Images, Videos	Office, PDF, Images	PDF, Music, Images, Videos	PDF, Office, Videos, Images, Music
Редагування у браузері	-	Вимагає ліцензію Office 365	-	-
Мобільні застосунки	Android, iOS	Android, iOS	Android, iOS, Windows	Android, iOS
WebDAV	+	-	Only via third-party integration	+
Підтримувані ОС	Windows, macOS, Linux	Windows, macOS	Windows, macOS, Linux	Windows, macOS, Linux
Віртуальний диск	+	-	Тільки для команд	+
Папка синхронізації	-	+	+	+
Блочне шифрування файлів	+	-	-	-
Вибіркова синхронізація	+	+	+	-
Синхронізація довільної папки	+	-	+	+

Можливість ділитись посиланнями для завантаження даних у сховище	+	+	+	+
Захист посилання паролем	+	+	+	+
Термін вичерпання дії посилання	+	+	+	+
Брендинг посилань	+	+	-	-
Обмеження завантаження посилань	-	+	-	-
Статистика посилань	+	-	-	-
Можливість ділитись посиланнями для завантаження даних зі сховища	+	+	+	+
Спільний доступ до папок	+	+	+	+
Встановлення дозволів на використання папки	+	+	+	-
Система версій файлів	+	+	+	+
Кошик для непотрібних файлів	+	+	+	+
Rewind feature	+	+	-	-
Розміщення сервера	US, EU	Canada	EU	UK, Germany, US
Відповідність стандарту	-	+	-	-

HIPAA				
Протокол шифрування	256-bit AES, 4096-bit RSA	256-bit AES, 2048-bit RSA	128-bit AES, 2048-bit RSA	256-bit Twofish
Шифрування у стані зберігання	+	+	+	+
Шифрування у стані передачі	+	+	+	+
Шифрування на стороні клієнта	+	+	+	+
Технологія нульового знання	+	+	+	+
Двофакторна аутентифікація	+	+	+	+
База поширених запитань	+	+	+	+
Форуми	-	-	-	-
Підтримка через email	+	+	+	+
Підтримка через чат	-	-	-	-
Підтримка через телефон	-	Тільки для бізнесу	-	-

Для порівняння хмарних сервісів, які використовують шифрування даних на стороні клієнта, потрібно зважати на такі характеристики, як:

- операційні системи, з якими сумісна програма;
- ціна продукту;
- застосовані алгоритми шифрування даних;
- зручна синхронізація;
- зрозумілий інтерфейс;
- доступні функції.

Розглянемо основні переваги та недоліки чотирьох розглянутих програмних засобів.

pCloud:

pCloud має надійне шифрування на стороні клієнта, навіть дослідники з Бостонського університету не змогли його зламати. З основних переваг – можливість зберігання разом як зашифрованих даних, так і не зашифрованих. Сервіс також підтримує блочну синхронізацію, завдяки чому лише змінений фрагмент буде перешифровано, що значно спрощує процедуру та економить час. Не всі хмарні сервіси мають таку функцію. pCloud має багато додаткових функцій, однак не всім користувачам вони потрібні. Недоліки – відсутність шифрування на стороні клієнта та спільного доступу до файлів у безкоштовній версії.

Сервіс підійде тим, хто хоче захистити конфіденційні дані, і при цьому мати доступ до швидкого перегляду фото та відео або інших файлів, які шифрувати на стороні клієнта не потрібно. Він є хорошим варіантом, якщо потрібна швидка робота сервісу, спільний доступ до файлів на надійна синхронізація, однак для цього потрібно обрати платну версію.

Sync.com:

Sync.com відповідає стандарту HIPAA, який дозволяє використовувати його навіть медичним установам, які зберігають максимально конфіденційні дані. З порівняних сервісів тільки Sync відповідає такому стандарту. Також він є хорошим варіантом якщо потрібно швидко редагувати файли, адже він дозволяє це зробити одразу у браузері. До його недоліків відноситься те, що він не може синхронізувати будь-яку папку та не підтримує блочну синхронізацію.

Він підійде тим, кому потрібно забезпечити високу надійність зберігання даних навіть у безкоштовній версії. На відміну від pCloud, сервіс підтримує особливості платформи із нульовим рівнем знань навіть у безкоштовній версії, однак має менше різноманітних функцій ніж pCloud у платних версіях.

Mega:

Mega має переваги в плані зручної синхронізації. На відміну від інших сервісів файли шифруються у браузері. Mega є зручним в плані використання, але не зовсім надійним. Хоч він і використовує шифрування на стороні клієнта, є ймовірність зламу третьою стороною, оскільки всі дані шифруються безпосередньо у браузері і можуть бути перехоплені. Також, хешована версія ключа зберігається на серверах сервісу, але немає гарантії, що для його розблокування Mega таємно не прочитає безпосередньо пароль для розшифрування ключа.

Icedrive:

Icedrive, на відміну від інших розглянутих сервісів, використовує для шифрування алгоритм не AES, а Twofish. Хоча AES прийнятий стандартом, Twofish добре працює для випадків, коли часто змінюються ключі, а також є гнучким. Тому даний сервіс є надійним. Однак шифрування потребує більше ресурсів, що може вплинути на час генерації ключа. Сервіс підійде тим, кому потрібне надійне шифрування, але високої швидкості шифрування даних.

ВИСНОВКИ ДО РОЗДІЛУ 6

У даному розділі було розглянуто особливості хмарних сервісів, які використовують шифрування на стороні клієнта. У розділі рекомендацій по вибору хмарного сервісу було визначено, у яких випадках який сервіс варто застосовувати. Також для вибору того чи іншого сервісу варто зважати на ціну та функціональність, яку сервіс надає за цю ціну.

7 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі здійснюється оцінка головних характеристик інструменту шифрування даних на стороні клієнта Cryptomator.

Візьмемо до уваги декілька варіантів реалізації програмного засобу та проведемо їхній аналіз для вибору оптимального, беручи до уваги економічні фактори, характеристики засобу, який розробляється, а саме такі, які мають вплив на продуктивність. Врахуємо сумісність продукту із апаратним забезпеченням. Використаємо для цього функціонально-вартісний аналіз.

Функціонально-вартісний аналіз – це технологія для незалежної оцінки справжньої вартості продукту або послуги. На кожному етапі реалізації, в залежності від необхідних обсягів ресурсів, здійснюється розподіл витрат – прямих та побічних – по продуктах та послугах. Дії, які виконуються на даних етапах, називаються функціями.

Метою ФВА є забезпечення найоптимальнішого розподілу ресурсів. Ресурси можуть бути виділені як на прямі, так і на непрямі витрати, тобто на виробництво продукції або надання послуг. У даному випадку відбувається аналіз функцій продукту та виявлення всіх витрат пов’язаних із реалізацією таких функцій.

Для початку, метод ФВА визначає послідовності функцій, які є необхідними для реалізації програмного засобу, а саме – усі можливі функції, які розподіляються на дві групи: такі, які не мають впливу на вартість продукту, і такі, що мають вплив. На цьому етапі також оптимізується скороченням кроків, що не впливають на витрати. Також на даному етапі скорочується сама послідовність.

Для кожної функції потрібно визначити повний обсяг річних витрат та необхідні робочі години, на основі чого формується кількісна характеристика джерел витрат. Далі, після визначення джерел витрат, проводиться розрахунок витрат на виробництво продукту.

7.1 Постановка задачі техніко-економічного аналізу

Застосуємо метод ФВА для проведення техніко-економічний аналізу розробки програмного засобу шифрування даних на стороні клієнта. Так як рішення з приводу побудови компонентів мають вплив на всю систему, кожна підсистема також повинна їм задовольняти. Фактичний аналіз у даному випадку представляє собою аналіз функцій програмного продукту, призначеного для шифрування даних на стороні клієнта з можливістю синхронізації із хмарним сховищем.

Наступні технічні вимоги були визначені до програмного засобу:

1) можливість роботи на персональних комп'ютерах та мобільних пристроях;
2) зручність та зрозумілість при використанні;
3) велика кількість підтримуваних мов;
4) можливість синхронізації із хмарними сервісами;
5) впровадження програмного засобу із мінімальними витратами.

7.2 Обґрунтування функцій програмного продукту

Головна функція F_0– розробка програмного засобу для шифрування даних на стороні клієнта із можливістю синхронізації із хмарними сервісами. Візьмемо дану функцію за основу та виділимо наступні:

F_1 – вибір мови програмування;

F_2 – вибір фреймворку для створення додатку;

F_3 – вибір середовища розробки.

Для кожної з цих функцій є декілька варіантів реалізації:

Функція F_1:

а) Python

б) Java

Функція F_2:

а) Spring Boot

б) Django

Функція F_3:

а) PyCharm

б) Intellij Idea

У морфологічній карті системи наведено варіанти реалізації основних функцій (Рисунок 7.1).

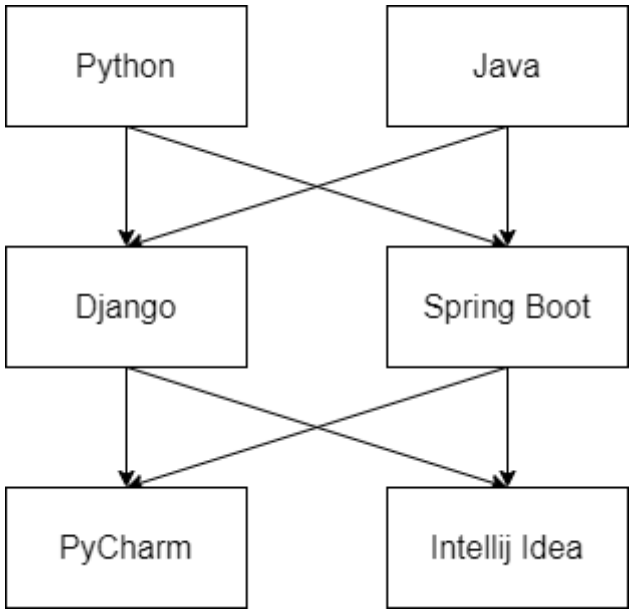

Рисунок 7.1 Морфологічна карта

Дана карта відображає множину всіх можливих варіантів основних функцій.

Таблиця 7.1 - Позитивно-негативна матриця

Функції	Варіанти реалізації	Переваги	Недоліки
F_1	*А*	Кросплатформність, швидка розробка програми	Низька швидкість роботи
	Б	Швидке виконання коду,	Повільна розробка

Функції	Варіанти реалізації	Переваги	Недоліки
		багато бібліотек	програми
F_2	*А*	Простий у використанні	Велика кількість зайвого коду
	Б	Хороша швидкість роботи при навантажені	Складність для розуміння
F_3	*А*	Швидкість роботи, зрозумілий інтерфейс	Підтримується виключно мова Python
	Б	Підтримка багатьох мов, багато додатків	Необхідно багато ресурсів

В результаті розгляду матриці формуємо висновок, що деякі варіанти реалізації функцій при розробці програмного засобу потрібно відкинути, адже вони не відповідають поставленим перед ПП задачам. Ці варіанти визначені у морфологічній карті.

Функція *F1*:

Перевагу віддаємо кросплатформності та швидкості розробки. Відкинемо варіант Б для спрощення роботи по написанню коду.

Функція *F2*:

У розробці можна застосовувати обидва варіанти.

Функція *F3*:

Віддаємо перевагу варіанту А у випадку вибору мови програмування Python.

Таким чином, будемо розглядати наступні варіанти реалізації програмного продукту:

1. F1а – F2а – F3а
2. F1а – F2б – F3а

Для оцінювання якості розглянутих функцій обрана система параметрів, яка описана нижче.

7.3 Обґрунтування системи параметрів програмного продукту

Визначимо основні параметри вибору, беручи до уваги дані про основні функції, які продукт повинен реалізувати, та вимоги до них. Ці параметри будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб описати програмний продукт, використаємо наступні параметри:

- *X1* – швидкодія мови програмування;
- *X2* – об’єм пам’яті для роботи програми;
- *X3* – час завантаження програми;
- *X4* – потенційний об’єм програмного коду.

На основі вимог замовника й умов, що характеризують експлуатацію програмного продукту, визначаються гірші, середні і кращі значення параметрів, як показано у таблиці 7.2.

Таблиця 7.2 - Основні параметри програмного продукту

Назва Параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	операцій/мс	10000	15000	18000
Об’єм пам’яті для роботи програми	X2	Мб	5000	3500	2000
Час завантаження програми	X3	МГц	15	10	8
Потенційний об’єм програмного коду	X4	рядки	6000	4000	2500

За даними таблиці 7.2 побудуємо графічні характеристики параметрів – (див. рис. 7.2 – 7.5).

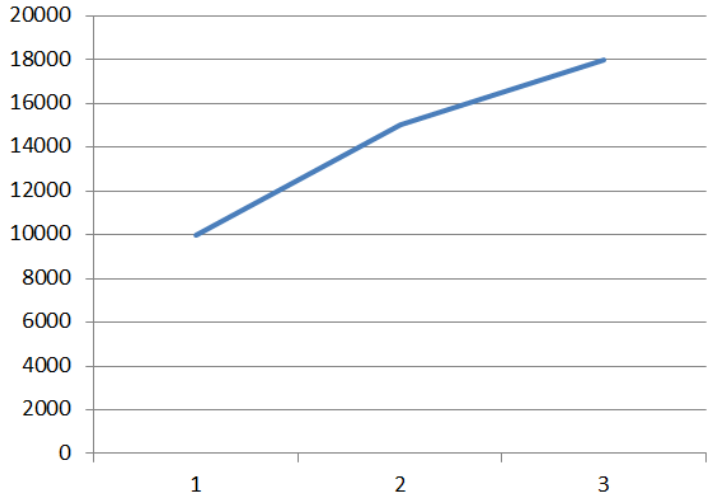

Рисунок 7.2 X1, швидкодія мови програмування

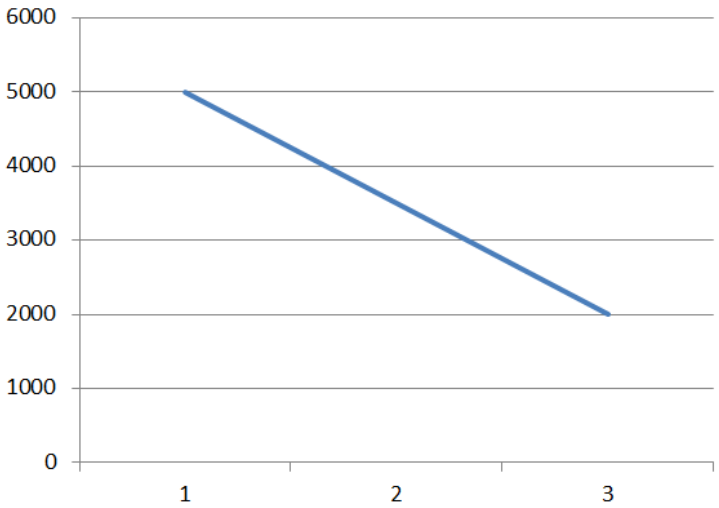

Рисунок 7.2 X2, об’єм пам’яті для збереження даних

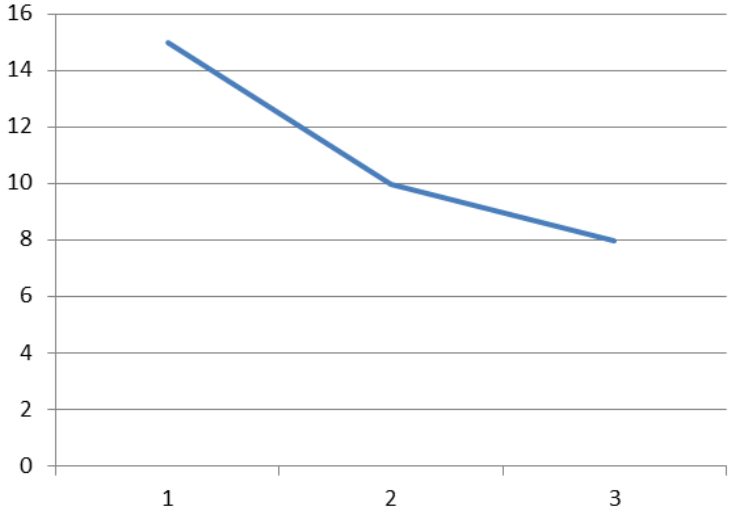

Рисунок 7.3 X3, час завантаження програми

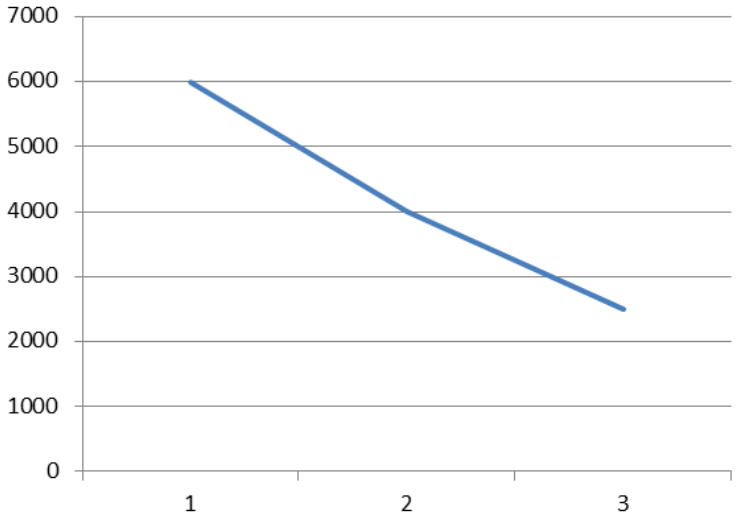

Рисунок 7.4 X4, об’єм програмного коду

7.4 Аналіз експертного оцінювання параметрів

Провівши обговорення та аналіз, кожен експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі, а саме розробки програмного засобу для шифрування даних на стороні клієнта.

За допомогою метода попарного порівняння визначається значимість кожного параметра. Експертна комісія, яка здійснює оцінку, складається із 7 людей. Визначення коефіцієнтів значимості передбачає:

1) визначення рівня значимості параметра шляхом присвоєння різних рангів;
2) перевірку придатності експертних оцінок для подальшого використання;
3) визначення оцінки попарного пріоритету параметрів;
4) обробку результатів та визначення коефіцієнту значимості.

У таблиці 7.4 наведені результати експертного ранжування.

Таблиця 7.3 - Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	операцій/мс	3	2	2	3	1	3	1	15	-7	49
X2	Об’єм пам’яті	Мб	4	4	5	5	3	4	4	29	7	49
X3	Час завантаження програми	МГц	4	5	4	3	4	5	3	28	6	36
X4	Об’єм програмного коду	рядки	2	3	3	1	2	3	2	16	-6	36
	Разом		13	14	14	12	10	15	10	88	0	170

Для того, щоб зробити перевірку степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 88 \quad (7.1)$$

де N – число експертів,

n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 22 \quad (7.2)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T \quad (7.3)$$

Сума відхилень за всіма параметрами повинна дорівнювати 0;

г) загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 170 \quad (7.4)$$

д) коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 170}{7^2(4^3 - 4)} = 0{,}69 > W_k = 0{,}67 \quad (7.5)$$

Знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67, тому ранжування можна вважати достовірним. Скориставшись результатами ранжування, проведемо попарне порівняння всіх параметрів і занесемо результати у таблицю 7.4.

Таблиця 7.4 - Попарне порівняння параметрів.

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	<	<	<	<	<	<	0,5
X1 і X3	<	<	<	<	<	<	<	<	0,5
X1 і X4	>	<	<	>	<	=	<	<	0,5
X2 і X3	=	<	>	>	<	<	>	=	1.0
X2 і X4	>	>	>	>	>	>	>	>	1,5
X3 і X4	>	>	>	>	>	>	>	>	1,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases} \quad (7.4)$$

З отриманих числових оцінок переваги складемо матрицю $A=\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{ві}$ за наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i} \quad (7.5)$$

$$b_i = \sum_{i=1}^{N} a_{ij} \quad (7.6)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'} \tag{7.7}$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j \tag{7.8}$$

Різниця значень коефіцієнтів вагомості, як видно з таблиці 7.5, не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 7.5 - Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1.0	0.5	0.5	0.5	2,5	0,16	25	0,33	50,125	0,16
X2	1.5	1.0	1.0	1.5	5	0,31	19	0,25	93,875	0,31
X3	1.5	1.0	1.0	1.5	5	0,31	19	0,25	93,875	0,31
X4	1.5	0.5	0.5	1.0	3,5	0,22	12,25	0,16	68,75	0,22
Всього:					16	1	75,25	1	306,625	1

7.5 Аналіз рівня якості варіантів реалізації функцій

Визначення рівення якості кожного варіанту виконання основних функцій здійснюється окремо.

Абсолютні значення параметрів *X2* (об’єм пам’яті), *X3* (швидкість завантаження програми), *X4* (об’єм програмного коду) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X2* (потенційний об’єм коду) обрано не найгіршим.

Коефіцієнт технічного рівня для кожного варіанта реалізації програмного продукту розраховується так (таблиця 6.6):

$$K_K(j) = \sum_{i=1}^{n} K_{ei,j} B_{i,j} \quad (7.9)$$

де n – кількість параметрів;

K_{ei}– коефіцієнт вагомості i–го параметра;

B_i – оцінка i–го параметра в балах.

Таблиця 7.6 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	X2	3500	8	0,31	2,48
F2	А	X3	10	7	0,31	2,17
	Б	X3	12	6	0,31	1,86
F3	А	X4	4000	3	0,22	0,66

За даними з таблиці 7.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}] \quad (7.10)$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 2{,}48 + 2{,}17 + 0{,}66 = 5{,}31$$

$$K_{K2} = 2{,}48 + 1{,}86 + 0{,}66 = 5$$

З розрахунків робимо висновок, що кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

7.6 Економічний аналіз варіантів розробки програмного продукту

Для визначення вартості розробки ПП проведемо спочатку розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю, алгоритми, які використовуються в завданні 1, належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як

$$T_О = T_Р \cdot К_П \cdot К_{СК} \cdot К_М \cdot К_{СТ} \cdot К_{СТ.М} \quad (7.11)$$

де $T_Р$ – трудомісткість розробки ПП;

$К_П$ – поправочний коефіцієнт;

$К_{СК}$ – коефіцієнт на складність вхідної інформації;

$К_М$ – коефіцієнт рівня мови програмування;

$К_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;

$К_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, беручи до уваги норми часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $T_Р$ = 30 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $К_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю

вхідної та вихідної інформації для всіх двох завдань рівний: $К_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це коефіцієнтом $К_{СТ}$ = 0.8. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$Т_1 = 30 \cdot 1.7 \cdot 0.8 = 40,8 \text{ людино-днів.}$$

Проведемо аналогічні розрахунки для другого завдання, де використовується алгоритм третьої групи складності, зі степенем новизни Б, тобто $Т_Р$ = 30 людино-днів, $К_П$ =0.9,$К_{СК}$ = 1,$К_{СТ}$ =0.8:

$$Т_2 = 30 \cdot 0.9 \cdot 0.8 = 21,6 \text{ людино-днів.}$$

Складемо трудомісткість завдань для кожного з обраних варіантів реалізації програми, щоб отримати загальну трудомісткість:

$$Т_I = (40,8 + 21,6 + 4.8 + 21,6) \cdot 8 = 710,4 \text{ людино-годин.}$$

$$Т_{II} = (40,8 + 21,6 + 28.8 + 21,6) \cdot 8 = 902,4 \text{ людино-годин.}$$

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 14000 та 20000 грн., один аналітик в області даних з окладом 25000. Визначимо середню зарплату за годину за формулою:

$$С_ч = \frac{М}{T_m \cdot t} \text{ грн.} \qquad (7.12)$$

де М – місячний оклад працівників;

T_m – кількість робочих днів тиждень;

t – кількість робочих годин в день.

$$С_ч = \frac{14000+20000+25000}{5 * 21 * 8} = 70 \text{ грн.} \quad (7.13)$$

Тоді, розрахуємо заробітну плату за формулою:

$$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д \quad (7.14)$$

де $С_ч$– величина погодинної оплати праці програміста;

$Т_i$ – трудомісткість відповідного завдання;

$К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП} = 70 \cdot 710,4 \cdot 1,2 = 59673,6$ грн.

II. $С_{ЗП} = 70 \cdot 902,4 \cdot 1,2 = 75801,6$ грн.

Відрахування на єдиний соціальний внесок становить 22%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 59673,6 \cdot 0,22 = 13128,192$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 75801,6 \cdot 0,22 = 16676,352$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 14000, з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 14000 \cdot 0,2 = 33600 \text{ грн.}$$

З урахуванням додаткової заробітної плати:

$$С_{ЗП} = С_Г \cdot (1 + К_З) = 33600 \cdot (1 + 0.2) = 40320 \text{ грн.}$$

Відрахування на соціальний внесок:

$$С_{ВІД}= С_{ЗП} \cdot 0.22 = 40320 \cdot 0,22 = 8870 \text{ грн.}$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 20000 грн.

$$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 20000 = 5750 \text{ грн.},$$

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача;

$К_А$– річна норма амортизації;

$Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1,15 \cdot 20000 \cdot 0,05 = 1150 \text{ грн.},$$

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$Т_{ЕФ} =(Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 10 – 8) \cdot 8 \cdot 0.9 =$$

$$= 1749,6 \text{ годин},$$

де $Д_К$ – календарна кількість днів у році;

$Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів;

$Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день;

$К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1749,6 \cdot 0,4 \cdot 0,4 \cdot 3,52 = 985,37 \text{ грн.},$$

де $N_С$ – середньо-споживча потужність приладу;

$К_З$– коефіцієнтом зайнятості приладу;

$Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$С_Н = Ц_{ПР} \cdot 0.67 = 20000 \cdot 0,67 = 13400 \text{ грн.}$$

Тоді, річні експлуатаційні витрати становитимуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н \quad (7.15)$$

$$С_{ЕКС} = 40320 + 8870 + 5750 + 1150 + 985,37 + 13400 = 70475,37 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 70475,37 / 985,37 = 71,52 \text{ грн/год.}$$

Так як всі роботи, пов’язані з розробкою програмного продукту, відбуваються на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складають:

$$С_М = С_{М-Г} \cdot Т \quad (7.16)$$

I. $С_М = 71,52 \cdot 710,4 = 50807,81$ грн.

II. $С_М = 71,52 \cdot 902,4 = 64539,65$ грн.

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67 \quad (7.17)$$

I. $С_Н = 59673,6 \cdot 0,67 = 39981,31$ грн.

II. $С_Н = 75801,6 \cdot 0,67 = 50787,07$ грн.

Отже, вартість розробки програмного продукту за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н \quad (7.18)$$

I. $С_{ПП} = 59673,6 + 13128,192 + 50807,81 + 39981,31 = 163590,912$ грн.

II. $С_{ПП} = 75801,6 + 16676,352 + 64539,65 + 50787,07 = 207804,672$ грн.

7.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕР j} = К_{Кj} / С_{Фj} \quad (7.19)$$

$К_{ТЕР1} = 5,31 / 163590,912 = 3,2459 \cdot 10^{-5}$,

$К_{ТЕР2} = 5 / 207804,672 = 2,40611 \cdot 10^{-5}$.

Отже, найефективнішим є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 3,2459 \cdot 10^{-5}$.

Цей варіант реалізації програмного продукту передбачає:

– мову програмування – Python;

– Використання Spring Boot

– Середовище розробки PyCharm

Даний варіант виконання програмного комплексу забезпечує швидке створення програмного продукту та високу швидкість роботи.

ВИСНОВКИ ДО РОЗДІЛУ 7

У даному розділі було здійснено функціонально-вартісний аналіз програмного продукту. Було розглянуто основні характеристики продукту, його функції та здійснено експертне оцінювання параметрів, на основі якого було визначено вибір кращого варіанту реалізації програмного продукту.

ВИСНОВКИ

У дипломній роботі було розглянуто методи захисту даних у хмарних технологіях, проаналізовано атаки на хмарні сховища, детально розглянуто застосування шифрування на стороні клієнта, досліджено сучасні програмні засоби шифрування даних на стороні клієнта, а саме програмні інструменти, які синхронізуються із вибраним хмарним сховищем, та хмарні сховища, які використовують шифрування на стороні клієнта.

Було проаналізовано способи захисту даних у хмарних технологіях. Було з’ясовано, що для надійності зберігання даних необхідно забезпечувати комплексний підхід із шифруванням даних як на стороні клієнта, так і під час їхньої передачі на сервер та на стороні сервера. Було проаналізовано можливі атаки на хмарні сервіси та способи підвищення захисту. Внаслідок цього було з’ясовано, що якщо засоби шифрування на стороні сервера підвищують захист даних від третіх сторін, то програмні засоби шифрування даних на стороні клієнта забезпечують повну конфіденційність даних, адже навіть провайдер хмарного сховища не здатен їх розшифрувати.

В результаті дослідження існуючих програмних засобів було розглянуто переваги, недоліки та особливості програмних засобів шифрування даних на стороні клієнта.

Програмні засоби Cryptomator та Boxcryptor дозволяють синхронізуватись із необхідним хмарним сховищем, яке підтримує таку синхронізацію. Основна відмінність між інструментами Cryptomator та Boxcryptor полягає у доступі до даних. В плані довіри до сервісу, а саме наявності вихідного коду, використаних алгоритмів шифрування, Cryptomator є хорошим варіантом. Він підходить для користувачів, які не планують використовувати хмарне сховище у режимі спільного доступу до файлів, адже шифрує дані симетричним алгоритмом, і щоб надати доступ до таких файлів іншим користувачам потрібно передавати безпосередньо ключ.

Boxcryptor, у свою чергу, шифрує дані асиметричним методом, тому за рахунок використання пари ключів забезпечується як конфіденційність зберігання даних у сховищі, так і можливість керування доступом до зашифрованих файлів користувачем. Тому він підійде командам та компаніям для спільної роботи із конфіденційними даними. Однак він не надає доступу до вихідного коду, що не може гарантувати невикористання ним засобів обходу, таких як бекдори.

Також було розглянуто хмарні сервіси, які використовують шифрування на стороні клієнта, а саме pCloud, Sync, Mega та Icedrive. Було розглянуто їхні особливості, переваги та недоліки, складено порівняльну таблицю та визначено, на які характеристики потрібно зважати при виборі, а саме: операційні системи, з якими сумісна програма; ціна продукту; застосовані алгоритми шифрування даних; зручна синхронізація; зрозумілий інтерфейс; доступні функції.

Загалом, pCloud найкраще підходить для зручного доступу як до зашифрованих, так і до незашифрованих даних, однак тільки у платній версії. Sync у безкоштовній версії має хороший функціонал платформи із нульовим рівнем знань і відповідає стандарту HIPAA. Mega використовує шифрування у браузері, тому є загроза зламу третіми сторонами. Icedrive має надійне шифрування, але повільнішу швидкодію. На основі розглянутих характеристик у роботі було сформовано рекомендації по вибору програмних засобів в залежності від потреб користувача.

ПЕРЕЛІК ПОСИЛАНЬ

1. How is Data stored in Cloud Computing? [Електронний ресурс]. Режим доступу https://coreitx.com/blogs/how-is-data-stored-in-cloud-computing#:~:text=With%20the%20cloud%2C%20the%20physical,according%20to%20the%20usage%20only.
2. Securing Data in the Cloud: Approaches and Importance [Електронний ресурс]. Режим доступу https://study.com/academy/lesson/securing-data-in-the-cloud-approaches-importance.html#:~:text=Security%20is%20achieved%20through%20a,non%2Dplain%20text)%20format.
3. Cloud Computing Attacks: A New Vector for Cyber Attacks [Електронний ресурс]. Режим доступу https://www.apriorit.com/dev-blog/523-cloud-computing-cyber-attacks
4. Protecting data using server-side encryption [Електронний ресурс]. Режим доступу https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html
5. AWS S3 Encryption [Електронний ресурс]. Режим доступу https://jayendrapatil.com/aws-s3-encryption/
6. Protecting data using client-side encryption [Електронний ресурс]. Режим доступу https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingClientSideEncryption.html
7. Client-Side Encryption with Customer-Managed Keys [Електронний ресурс]. Режим доступу https://subscription.packtpub.com/book/cloud-and-networking/9781789534474/23/ch23lvl1sec73/client-side-encryption-with-customer-managed-keys-cse-c

8. End-To-End Encryption [Електронний ресурс]. Режим доступу https://cloudstorageinfo.org/client-side-encryption-cloud-storage
9. Cryptomator [Електронний ресурс]. Режим доступу https://ostechnix.com/cryptomator-open-source-client-side-encryption-tool-cloud/#:~:text=to%20help%20you%3A-,How%20Cryptomator%20works%3F,and%20secured%20with%20a%20passphrase
10. Cloud Storage with Client-Side Encryption [Електронний ресурс]. Режим доступу https://www.scitepress.org/Papers/2020/91306/91306.pdf
11. Cryptomator Documentation [Електронний ресурс]. Режим доступу https://docs.cryptomator.org/en/latest/security/security-target/
12. Boxcryptor Documentation [Електронний ресурс]. Режим доступу https://www.boxcryptor.com/en/help/introduction/android/#what-is-boxcryptor
13. Cryptomator and Boxcryptor in Comparison Documentation [Електронний ресурс]. Режим доступу https://www.boxcryptor.com/en/blog/post/cryptomator-check-cloud-encryption/
14. pCloud Review [Електронний ресурс]. Режим доступу https://cloudstorageinfo.org/pcloud-review
15. What is pCloud Crypto and How It Works [Електронний ресурс]. Режим доступу https://www.cloudwards.net/what-is-pcloud-crypto/
16. How To Set Up Crypto Folder [Електронний ресурс]. Режим доступу https://blog.pcloud.com/how-to-set-up-your-crypto-folder/
17. Sync.com Review [Електронний ресурс]. Режим доступу https://cloudstorageinfo.org/pcloud-review

18. Sync.com Technical Overview [Електронний ресурс]. Режим доступу https://www.sync.com/pdf/sync-privacy-whitepaper.pdf
19. Sync.com and pCloud in Comparison [Електронний ресурс]. Режим доступу https://www.cloudwards.net/sync-com-vs-pcloud/
20. Mega review [Електронний ресурс]. Режим доступу https://cloudstorageinfo.org/mega-cloud-storage-review
21. Mega Service [Електронний ресурс]. Режим доступу https://mega.io/
22. Mega's Encryption [Електронний ресурс]. Режим доступу https://help.mega.io/files-folders/sharing/encrypted-links
23. Icedrive Review [Електронний ресурс]. Режим доступу https://cloudstorageinfo.org/icedrive-review
24. Icedrive Encryption Process [Електронний ресурс]. Режим доступу https://icedrive.net/help/encryption/the-icedrive-encryption-process-explained