Вісник ЛДУБЖД
Bulletin of Lviv State University of Life Safety
https://journal.ldubgd.edu.ua/index.php/Visnuk
ISSN 2078-4643 (print), ISSN 2708-1389 (online)

UDC 004.75
DOI: 10.32447/20784643.27.2023.06

С. В. Кулина
Західноукраїнський національний університет, м. Тернопіль, Україна
ORCID: https://orcid.org/0000-0002-6162-9457 – С. В. Кулина
sersks@gmail.com

СИСТЕМА РОЗПОДІЛЕНОГО ЗАХИЩЕНОГО ЗБЕРІГАННЯ ДАНИХ

Проблема. Основною метою кібербезпеки є гарантувати безпеку користувачів та їх даних. Зазвичай кібератаки спрямовані на пошкодження важливих даних у корпоративній і персональній мережі або отримання доступу до них. Такі атаки можуть здійснювати як окремими особами, так і цілими організаціями. Вимоги кібербезпеки часто замість того, щоб стимулювати проектування стійких мереж, призводять до додавання засобів контролю безпеки до архітектури існуючої системи, проте зазвичай неможливо перешкодити зловмиснику, який має намір проникнути в систему цілі. Тому на перше місце виходять технології резервного копіювання даних та збільшення стійкості систем у випадку атак. Одним із таких рішень, що дає можливість негайного відновлення даних, можуть слугувати децентралізовані системи зберігання файлів. Вони є привабливим та поширеним рішенням для керування великими та складними системами, такими як енергетичні підприємства, робототехніка, водопровідні мережі, бездротові сенсорні мережі, керування трафіком тощо.

Метою роботи є розробка структурної схеми системи розподіленого захищеного зберігання даних та основних її компонентів.

Методи. При дослідженні та розробці системи розподіленого захищеного зберігання даних використано методи модулярної арифметики, методи виявлення та виправлення помилок в системі залишкових класів, оцінювання та порівняння показників систем, що дало змогу реалізувати системний підхід до оцінки стійкості розроблюваної системи. Загальні принципи побудови системи дають можливість визначити її компоненти та ефективно їх поєднати.

Результати дослідження. Одним із методів побудови розподілених систем можна вважати систему залишкових класів. Завдяки розділенню файлів на блоки - залишки та збереженню їх в розподілені сховища досягається шифрування даних та можливість відновлення у випадку втрати.

Висновки. В роботі представлено структуру системи розподіленого захищеного зберігання даних. Розроблено блок-схеми модулів, а також алгоритми компонент. Запропоновано метод відновлення пошкоджених файлів при спотворенні частини файлів залишків. Для запропонованого методу розраховано залежність ймовірності виправлення помилок від кількості помилок із врахуванням кількості пошкоджених файлів залишків. Наведено приклад відновлення пошкодженого файлу при виникненні трьох помилок у двох файлах залишків.

Ключові слова: Система залишкових класів, розподілене зберігання даних, захищені системи зберігання даних, відновлення даних.

S. V. Kulyna
West Ukrainian National University, Ternopil, Ukraine

DISTRIBUTED PROTECTED DATA STORAGE SYSTEM

Introduction. The main goal of cyber security is to ensure the safety of users and their data. Usually, cyber attacks are aimed at damaging important data in corporate and personal networks or gaining access to them. Such attacks can be carried out by both individuals and entire organisations. Rather than driving the design of resilient networks, cyber security requirements often lead to the addition of security controls to existing system architectures, but it is usually impossible to prevent an attacker from penetrating a target's system. Therefore, data backup technology and increasing stability of the system in the event of attacks come first. Decentralised file storage systems can serve as one such solution that enables immediate data recovery. They are attractive and common solutions for managing large and complex systems such as power plants, robotics, water networks, wireless sensor networks, traffic management, etc.

The purpose of the work is to develop a structural diagram of the distributed secure data storage system and its main components.

Methods. During the research and development of distributed protected data storage systems, methods of modular arithmetic, methods of detecting and correcting errors in the redundant residue number system, evaluation and comparison of system indicators were used, which made it possible to implement a systematic approach to assessing the stability of the developed system. The general principles of system construction make it possible to determine its components and effectively combine them.

Results. One of the methods of building distributed systems can be considered the use of the redundant residue number system. Due to the division of files into blocks – remnants and their storage in distributed storage, data encryption and the possibility of recovery in case of loss are achieved.

Conclusion. The paper presents the structure of distributed protected data storage systems. Block diagrams of modules, as well as component algorithms, have been developed. A method of restoring damaged files when a part of the remnant file is distorted is proposed. For the proposed method, the dependence of the error correction probability on the number of errors, taking into account the number of damaged residual files, was calculated. An example of restoring a damaged file when three errors occur in two residual files is given.

Keywords: redundant residue number system, distributed data storage, protected data storage systems, data recovery.

Вступ. Відомі стратегії боротьби з програмами-вимагачами та іншим шкідливим ПЗ рекомендують певні алгоритми захисту, однак досвід показує, що надзвичайно складно перешкодити добре підготовленому та технічно забезпеченому нападнику, який має намір проникнути в систему жертви [1]. У свою чергу, посібник із найкращих практик програм-вимагачів від Агентства з кібербезпеки та безпеки інфраструктури (CISA) Міністерства національної безпеки починається з рекомендації «зберігати зашифровані резервні копії даних у автономному режимі та регулярно створювати свої резервні копії» [2]. Відповідно, наявні стратегії захисту інформації спрямовані на підвищення стійкості та потребують підтримки резервного копіювання системи, що може запобігти значній втраті даних, проте потребує додаткових ресурсів для обробки та зберігання інформації [3].

Поширені системи зберігання інформації зазвичай використовують централізовану структуру, де за обробку інформації відповідає сервер. Будь-який запит у мережі проходить через нього і у випадку кібератак саме він в першу чергу піддається загрозам. Атаки можуть відбуватися як із зовнішньої мережі, так і з робочих станцій, підключених безпосередньо до сервера. У випадку компрометації сервера інформація може бути зашифрована або видалена.

Одним із ефективних шляхів боротьби із кіберзагрозами є налаштування мереж таким чином, щоб збільшити їхню стійкість після атак. Зокрема, поєднання захисту пристроїв, таких як сервери та робочі станції, та наявності плану і можливості негайного відновлення даних на цих пристроях [4].

Рішенням, що дають можливість реагувати на такі кіберзагрози, можуть слугувати децентралізовані системи зберігання файлів, які замість зберігання файлів і даних на центральному сервері розбивають, хешують та шифрують файли, зберігаючи фрагменти в різних сховищах [5, 6].

Аналіз останніх досліджень та публікацій. У випадку кібератаки на децентралізовану систему користувачі можуть відмовитися від скомпрометованих пристроїв використовуючи нові машини, щоб повторно зібрати свої файли та відновити роботу у звичному режимі, а шифрування запобігає використанню їх для вимагання чи інших цілей.

Децентралізовані системи є привабливими та поширеними рішеннями для керування великими та складними системами, такими як енергетичні підприємства, робототехніка, водопровідні мережі, бездротові сенсорні мережі, керування трафіком тощо [7, 8].

У [9] досліджуються проблеми децентралізованого керування об’єктами та пропонується три підходи до керування великомасштабними взаємопов’язаними системами.

Для гарантування роботи стабільної великомасштабної енергетичної системи використовуються децентралізовані контролери із зворотні зв’язком [10].

У [11] досліджується використання автономної децентралізованої технології, як один із заходів забезпечення безпеки в системі управління, що інтегрує функції інформаційного обслуговування. Проте, збереження резервних копій системи і надалі вважається основним засобом обмеження шкоди та створення стійкості після атаки [2].

При резервному копіюванні системи забезпечується певна стійкість, проте може відбутися втрата даних у разі повернення до останнього відомого справного стану даних [4].

Системні резервні копії є кращими, ніж нічого, але їх недостатньо, тому NIST Privacy Framework [4] зазначає, що успіх місії та бізнес-функції залежать від «конфіденційності, цілісності та доступності інформації, яка обробляється, зберігається та передається», тобто захист даних є важливішим, ніж захист пристроїв або навчання користувачів захисту пристроїв.

Одним із підходів до резервного копіювання, які ускладнюють доступ зловмисників до інформації, є використання розподілених систем зберігання даних [12]. У випадку отримання зловмисником доступу до частини даних, або одного із носіїв з інформацією, використання блочного типу шифрування, наприклад AES 128, та збереження блоків на різних носіях дозволяє додатково захистити інформацію від несанкціонованого видалення та розшифровування.

Система залишкових класів (СЗК) є не позиційною системою числення, що дає можливість реалізувати ефективну систему

розподіленого зберігання даних. Це досягається шляхом розділення повідомлення на фрагменти та їх поділу на систему взаємно простих модулів. Отримані залишки зберігаються в розподілених сховищах, що робить неможливим їх криптоаналіз.

Тому, використання системи залишкових класів є одним із напрямків підвищення надійності та захищеності систем зберігання даних [13].

Методи досліджень. При дослідженні та розробці системи розподіленого захищеного зберігання даних (СРЗЗД) використано методи модулярної арифметики, методи виявлення та виправлення помилок в СЗК, оцінювання та порівняння показників систем, що дало змогу реалізувати системний підхід до оцінки стійкості розроблюваної системи. Загальні принципи побудови системи дають можливість визначити її компоненти та ефективно їх поєднати.

Результати досліджень. У попередній роботі [14] проводились дослідження методів відновлення пошкоджених даних використовуючи надлишкову систему залишкових класів. При проектуванні СРЗЗД обирається кількість та значення взаємопростих модулів, які будуть відповідати поставленим вимогам.

Розроблена структура СРЗЗД побудована на основі чотирьох пристроїв зберігання даних (ПЗД) містить наступні основні модулі: модуль розділення на фрагменти, модуль запису/зчитування, модуль вибору файлів залишків, модуль зворотного перетворення та модуль відновлення (рис. 1).

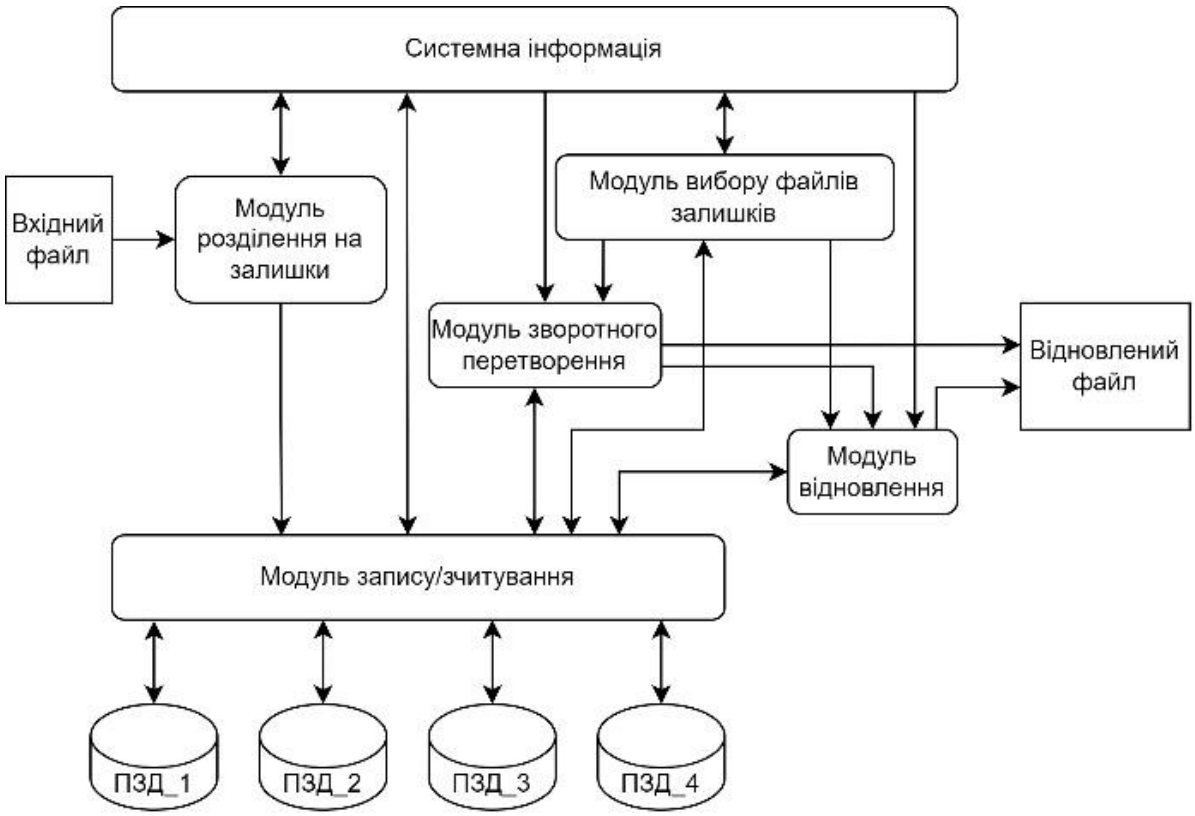

Рисунок 1 – Загальна структура СРЗЗД

Для оцінки ефективності такої системи необхідно розглянути кожен із модулів більш детально.

Модуль розділення на залишки використовується для розділення вхідного файлу на залишки по вибраній системі взаємно простих модулів та їх зберігання у відповідних файлах.

Алгоритм розділення даних з використанням надлишкової СЗК показано на рисунку 2.

Для початку роботи модуль розділення на фрагменти зчитує початкову інформацію таку як: взаємно прості модулі, тимчасові назви для файлів, лічильники, масиви та іншу допоміжну інформацію. Після зчитування вхідного файлу обчислюється хеш файлу, який надалі зберігається у файлі сесії.

Для розбиття на фрагменти визначається довжина вхідного файлу, при цьому розмір самих фрагментів залежить від робочого діапазону обраного набору модулів. Кожен із зазначених фрагментів надалі розглядається як окрема послідовність даних, від якої беруться залишки згідно з обраною системою модулів.

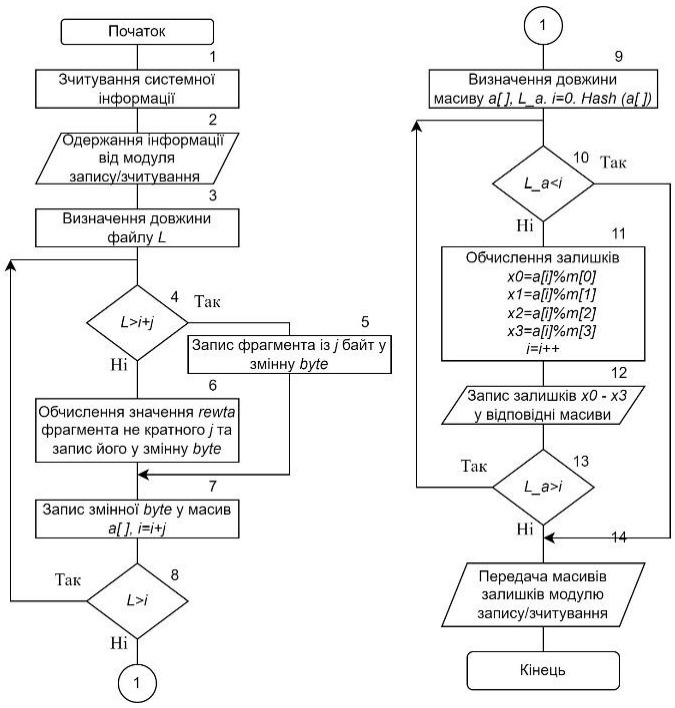

Рисунок 2 – Алгоритм розділення файлу даних на залишки в СЗК

Наступним кроком роботи системи є використання модуля запису/зчитування для зберігання даних та запис системної інформації, отриманої в результаті роботи.

Модуль запису/зчитування служить буфером обміну та забезпечує зв'язок інших модулів із підключеними ПЗД. Він також записує отримані файли залишків на відповідні ПЗД.

Окрім цього, даний модуль після отримання файлів залишків обчислює їх хеш та записує його в назву файлу, для подальшого визначення будь-яких модифікацій їх вмісту на пристроях ПЗД. Після успішного виконання цього кроку розділення даних на файли залишків є завершеним.

Використання надлишкової СЗК, як основи для побудови СРЗЗД, забезпечує можливість відновлення початкового файлу даних, зчитавши лише 3 із 4 збережених файлів залишків.

Модуль вибору файлів залишків дає можливість обрати, які саме із них будуть використовуватися при відновленні даних. Алгоритм вибору файлів залишків, які будуть використані для зворотного перетворення, показаний на рисунку 3.

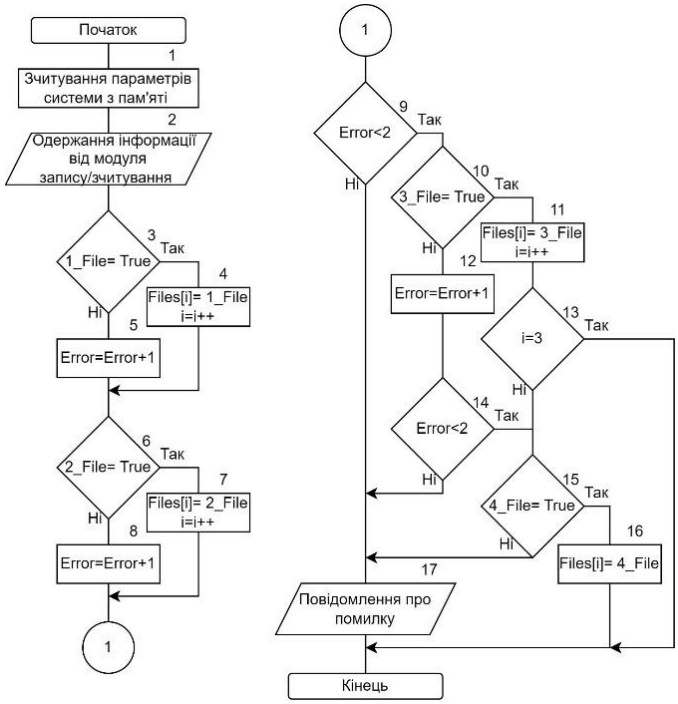

Рисунок 3 – Алгоритм вибору файлів залишків

Вибір файлів залишків, з допомогою яких буде відновлено початкове значення даних здійснюється шляхом порівняння хешу кожного із файлів залишків з хешем, записаним у назву файлу при їх зберіганні.

Саме модуль вибору файлів залишків визначає доступність файлів для зчитування, а пізніше почергову перевірку їх хешів. Якщо перші три файли залишків доступні і їх хеш-значення збігаються, то цього достатньо для відновлення початкового файлу.

У випадку, коли один із файлів не доступний для зчитування або його хеш відрізняється від збереженого, здійснюється перевірка доступності і цілісності наступного (четвертого) файлу залишків. У випадку пошкодження користувач отримає повідомлення про неможливість відновлення початкового файлу цим методом.

За умови, що перевірка доступності та цілісності будь-яких трьох файлів залишків пройшла успішно, підключається **модуль зворотного перетворення**. Алгоритм роботи модуля представлений на рисунку 4.

Рисунок 4 – Алгоритм зворотного перетворення даних

Алгоритм роботи модуля зворотного перетворення даних полягає у послідовному зчитуванні фрагментів даних із обраних файлів залишків та обчисленні значення початкового

фрагмента даних. Послідовність початкових фрагментів записується у файл із використанням модуля запису/зчитування. За умови, що хеш отриманого файлу дорівнює хешу вхідного файлу, який зберігається в файлі сесії, отримай файл вважається відновленим. Якщо фрагменти, що зчитуються з початкового файлу, мають розрядність один байт, то вони всі будуть однаковими і введення додаткових обчислень або змінних не потрібно.

Однак зчитування однобайтових фрагментів є неефективним. Відповідно виникатиме ситуація, коли останній фрагмент не буде кратний обраному розміру фрагмента.

При прямому перетворенні така проблема не враховується, оскільки в бітовому представлені додані на початку всіх розрядів нулі не змінять значення фрагменту файлу залишку (таблиця 1).

Таблиця 1

Залежність десяткового значення фрагменту від початкового двійкового представлення

Розрядність останнього фрагменту початкового файлу, байт	**Представлення значення в двійковому форматі**	**Представлення значення в десятковому форматі**	**Розрядність останнього фрагменту файлу залишків, байт**
1	00000010	2	1
2	00000000 00000010	2	1
3	00000000 00000000 00000010	2	1
4	00000000 00000000 00000000 00000010	2	1
5	00000000 00000000 00000000 00000000 00000010	2	1
6	00000000 00000000 00000000 00000000 00000000 00000010	2	1

Як видно з таблиці 1, фрагменти різної довжини можуть мати однакові значення в десятковому поданні і як результат будуть мати однакову кількість байтів у файлі залишків при цьому ніяк не змінюючи його. Однак, під час зворотного перетворення важливо, скільки байтів потрібно записати у файл при відновленні (таблиця 2).

Таблиця 2

Залежність двійкового представлення значення від кінцевої розрядності фрагменту

Розрядність файлу залишків, байт	**Значення в десятковому форматі**	**Розрядність останнього фрагменту запису в початковий файл, байт**	**Значення фрагменту в двійковому представленні**
1	2	1	00000010
1	2	2	00000000 00000010
1	2	3	00000000 00000000 00000010
1	2	4	00000000 00000000 00000000 00000010
1	2	5	00000000 00000000 00000000 00000000 00000010
1	2	6	00000000 00000000 00000000 00000000 00000000 00000010

Як бачимо з таблиці 2, під час зворотного перетворення автоматичне заповнення розрядів нулями змінює вміст файлу, щоб уникнути такої ситуації здійснюється попередній запис розміру останнього фрагменту при розділенні та перевірка його значення при зворотному перетворенні.

Якщо розмір початкового файлу був кратний розміру обраного фрагмента, то ніяких додаткових дій виконувати не потрібно. В іншому випадку значення останнього обчисленого фрагменту має бути записане у файл.

Для перевірки цілісності файлу обчислюється його хеш значення та подальший запис у файл із відповідним ім’ям за допомогою модуля читання/запису.

Розроблений метод. Використання надлишкової СЗК з одним перевірочним модулем дає змогу відновити значення по $(n-1)$ залишках. Для визначення пошкодженого файлу використовується хеш функція.

Суть розробленого методу полягає в тому, що модуль відновлення визначає позиції спотворених

залишків і відновлює їх з використанням методу проекцій та порівняння хеш суми файлів.

Отже цей метод дає можливість відновити файл при спотворенні двох і більше файлів залишків при певній комбінації помилок. Всі можливі варіанти помилок в СРЗЗД з трьома основними і одним перевірочним модулем приведені в таблиці 3

Таблиця 3

Варіанти пошкодження фрагментів у системі із n=4

№	1-й залишок	2-й залишок	3-й залишок	4-й залишок
1	*	x_2	x_3	x_4
2	x_1	*	x_3	x_4
3	x_1	x_2	*	x_4
4	x_1	x_2	x_3	*
5	*	*	x_3	x_4
6	*	x_2	*	x_4
7	*	x_2	x_3	*
8	x_1	*	*	x_4
9	x_1	*	x_3	*
10	x_1	x_2	*	*
11	*	*	*	x_4
12	*	*	x_3	*
13	*	x_2	*	*
14	x_1	*	*	*
15	*	*	*	*

де x_1- x_4 – цілий залишок, * – пошкоджений.

Як видно з таблиці 3, при загальній кількості модулів n=4 існує v=15 варіантів пошкодження фрагментів в файлах залишків. Відповідно, коли в масиві виникають 2 помилки, $v=15^2$, а загальна кількість можливих помилок дорівнює:

$$v=15^k,$$

де: k – кількість помилок, які виникли в масиві даних.

Метод проекцій забезпечує відновлення початкового значення X при пошкодженні одного із залишків за умови використання двох перевірочних модулів [14].

У цій системі кількість перевірочних модулів $r=1$, а отже, однозначно визначити, яка із проекцій правильна, використовуючи класичний метод проекцій, неможливо, лише завдяки пошуку всіх варіантів і порівнянню його хешу з хешем початкового файлу можна визначити, яка проекція відповідає початковому значенню. Вводячи додаткові умови та відбираючи лише ті випадки, коли файл підлягає відновленню, можна значно скоротити кількість варіантів, які необхідно обчислити.

Запропоновано використовувати порівняння хешів файлів залишків, щоб відкинути частину варіантів. За умови, що хеш одного із файлів залишків відповідає хешу, записаному в його назві, максимальна кількість пошкоджень з однією помилкою становить v=7 і не залежить від того, які фрагменти пошкоджено, а загальну кількість помилок можна представити як $v=7^k$ (таблиця 4).

Таблиця 4

Варіанти пошкодження залишків у системі із n=4 при цілому 4-му залишку

№	1-й залишок	2-й залишок	3-й залишок	4-й залишок
1	*	x_2	x_3	x_4
2	x_1	*	x_3	x_4
3	x_1	x_2	*	x_4
4	*	*	x_3	x_4
5	*	x_2	*	x_4
6	x_1	*	*	x_4
7	*	*	*	x_4

За умови, що хеші двох файлів залишків не збігаються, тобто пошкоджено два фрагменти з чотирьох, кількість варіантів пошкодження з однією помилкою в масиві даних становить $v=3$, а загальна кількість дорівнює $v=3^k$ (таблиця 5).

Таблиця 5

Варіанти пошкодження залишків у системі із n=4 при цілому 2-му та 4-му залишках

№	1-й залишок	2-й залишок	3-й залишок	4-й залишок
1	*	x_2	x_3	x_4
2	x_1	x_2	*	x_4
3	*	x_2	*	x_4

Якщо на етапі вибору файлів залишків визначено, що пошкоджено більше ніж один з них, а при відновленні обчислено, що вони відбулися в одному фрагменті, то значить виправити таке пошкодження неможливо. Тому необхідно зазначити, що модуль відновлення працює з помилкою в двох і більше фрагментах файлів залишків і всі обчислення будуть проводитись виходячи з цього твердження.

У таблиці 6 наведено загальну кількість помилок, які можуть виникнути, та кількість, яку можна виправити запропонованим методом.

Таблиця 6

Залежність кількості помилок від кількості цілих файлів залишків

№	Кількість непошкоджених файлів залишків	Загальна кількість помилок	Кількість помилок, які можна виправити
1	0	15^k	4^k
2	1	7^k	3^k
3	2	3^k	2^k

де: k – кількість пошкоджених фрагментів відновлювального файлу

Зазвичай при застосуванні методу проекцій шукане значення X, знаходиться в діапазоні $0 \le X \le R$, де R – робочий діапазон, а інші проекції X'' знаходяться в діапазоні $R \le X \le M$, де M – загальний діапазон.

У випадку, коли використовується лише один перевірочний модуль, значення проекцій не відповідає цим умовам, але одна з проекцій все одно буде очікуваним результатом. Після перегляду всіх варіантів і порівняння хеш-значення суми масиву даних, який містив втрачений вихідний файл, з хешем відновленого, є певна ймовірність отримати початковий файл.

Як згадувалося раніше, користувач звертається до модуля відновлення лише тоді, коли пошкоджено більше одного файлу, тому існує ймовірність, що два файли із залишками були пошкоджені в одному фрагменті, а це означає, що використання методу проекцій не матиме жодного ефекту. Можна також зауважити, що зі збільшенням кількості помилок у файлах залишків, ефективність цього методу знижується. Залежність ефективності відновлення даних від кількості помилок при n=4 та пошкодженні усіх файлів залишків розраховано в таблиці 7.

Таблиця 7

Залежність ефективності відновлення файлу від кількості помилок при пошкодженні усіх файлів залишків

k	Загальна кількість помилок	Кількість помилок, які можна виправити	Ефективність, %
2	225	16	7,111
3	3375	64	1,896
4	50625	256	0,506
5	759375	1024	0,135
10	5,77E+11	1,05E+06	>0,001
20	3,33E+23	1,10E+12	>0,001
50	6,38E+58	1,27E+30	>0,001
100	4,07E+117	1,61E+60	>0,001
200	1,65E+235	2,58E+120	>0,001

Як видно із таблиці 7 ймовірність виправити чотири помилки при пошкодженні всіх файлів залишків менше 1 %, а при 10 і більше – менше 0,001%.

Залежність ефективності модуля відновлення від кількості помилок при n=4 та трьох пошкоджених файлах залишків обчислено в таблиці 8.

Таблиця 8

Залежність ймовірності виправлення помилок від кількості помилок при трьох пошкоджених файлах залишків

k	Загальна кількість помилок	Кількість помилок, які можна виправити	Ефективність, %
2	49	9	18,367
3	343	27	7,872

Продовження таблиці 8

k	Загальна кількість помилок	Кількість помилок, які можна виправити	Ефективність, %
4	2401	81	3,374
5	16807	243	1,446
10	2,82E+08	59049	0,021
20	7,98E+16	3,49E+09	>0,001
50	1,80E+42	7,18E+23	>0,001
100	3,23E+84	5,15E+47	>0,001
200	1,05E+169	2,66E+95	>0,001

Як ми бачимо, при трьох пошкоджених файлах залишків ймовірність виправити дві помилки досить висока і становить 18%. Збільшення кількості помилок призводить до зменшення ефективності – ймовірність виправити 10 помилок становить 0,21%, а 20 і більше – менше 0,001%. Залежність ефективності відновлення файлу від кількості помилок при n=4 та двох пошкоджених фрагментах файлів залишків обчислено в таблиці 9.

Таблиця 9

Залежність ймовірності виправлення помилок від кількості помилок при двох пошкоджених файлах залишків

k	Загальна кількість помилок	Кількість помилок, які можна виправити	Ефективність, %
2	9	4	44,444
3	27	8	29,630
4	81	16	19,753
5	243	32	13,169
10	59049	1024	1,734
20	3,49E+09	1,05E+06	0,030
50	7,18E+23	1,13E+15	>0,001
100	5,15E+47	1,27E+30	>0,001
200	2,66E+95	1,61E+60	>0,001

У випадку, коли пошкоджено два з чотирьох файлів залишків (табл. 8) ймовірність виправити дві помилки становить 44%. Як і в попередніх розглянутих випадках, при збільшенні кількості помилок ймовірність їх виправлення зменшується, при цьому ймовірність виправлення 50 і більше помилок становить менше 0,001%.

Порівняння хешу відновленого та початкового файлу в кінці роботи модуля відновлення дозволяє уникнути помилкового повідомлення про успішне відновлення файлу після запису даних у кінцевий файл.

Якщо хеш збігається, можна зробити висновок, що файли залишків були пошкоджені в різних фрагментах і процес відновлення пройшов успішно. В іншому випадку відновлення даних не відбулося, про що користувач буде сповіщений відповідним повідомленням. Алгоритм зворотного перетворення із використанням удосконаленого методу проекцій, який реалізовано в модулі відновлення, представлено на рисунку 5.

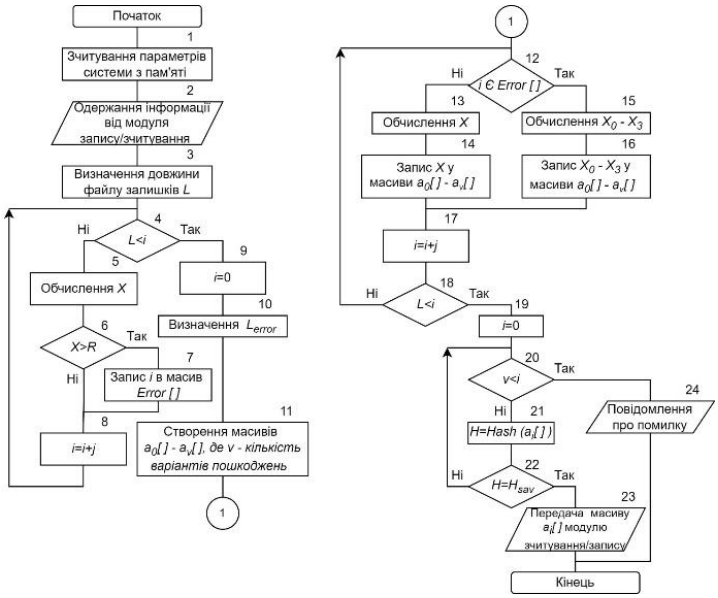

Рисунок 5 – Алгоритм зворотного перетворення на основі удосконаленого методу проекцій

Першим кроком при роботі модуля відновлення є визначення кількості пошкоджених фрагментів. Ми знаємо, що при пошкодженні одного із залишків шукане значення X вийде за межі робочого діапазону, тобто $R \le X \le M$, тому, розбиваючи весь масив даних на фрагменти, обчислюючи почергово значення X і перевіривши його належність до робочого діапазону, ми отримаємо всі номери фрагментів, в яких сталася одна помилка.

Ми також зможемо визначити фрагменти, де є пошкодження двох або більше залишків, але не зможемо їх виправити при використанні одного перевірочного модуля.

Перевіривши весь масив даних, ми отримаємо масив, який міститиме індекси всіх пошкоджених фрагментів, а визначивши його довжину, дізнаємося загальну кількість помилок. Використовуючи масив, який містить імена непошкоджених файлів залишків і формули з таблиці 6, можна обчислити кількість масивів, які нам потрібно створити та перевірити.

Розглянемо приклад формування таких масивів за умови, що при пошкодженні двох файлів залишків виникає три помилки. Відповідно до таблиці 6 загальна кількість можливих варіантів помилок становитиме $3^3=27$, а кількість помилок, які ми можемо виправити – $2^3=8$.

Варто зауважити, що вісім варіантів, які можна виправити, включають обидва випадки, які вже були відкинуті, тобто коли три помилки знаходяться в одному із файлів залишків.

Отже, необхідно створити 6 масивів і обчислити значення проекцій для них згідно з таблицею 10.

У подальшому фрагменти із файлів залишків необхідно прочитати повторно. З кожним фрагментом у випадку, коли він не пошкоджений, необхідно виконати операцію зворотного перетворення і записати отримане значення у всі створені масиви.

В іншому випадку, коли фрагмент пошкоджений, необхідно обчислити значення всіх проекцій і записати їх у порядку, згідно з таблицею 10, в новостворені масиви.

В результаті проведених обчислень, для наведеного прикладу, сформовано 6 масивів, один з яких, за умови відсутності пошкодження двох і більше залишків в одному фрагменті, є шуканим.

Таблиця 10

Варіант виникнення трьох помилок у системі із n=4 при цілому 3-му та 4-му фрагментах

№	1-й залишок	2-й залишок	3-й залишок	4-й залишок		№	1-й залишок	2-й залишок	3-й залишок	4-й залишок
1 варіант						**4 варіант**				
1.	…	…	…	…		1.	…	…	…	…
2.	*	x_2	x_3	x_4		2.	x_1	*	x_3	x_4
3.	…	…	…	…		3.	…	…	…	…
4.	*	x_2	x_3	x_4		4.	x_1	*	x_3	x_4
5.	…	…	…	…		5.	…	…	…	…
6.	x_1	*	x_3	x_4		6.	*	x_2	x_3	x_4
2 варіант						**5 варіант**				
1.	…	…	…	…		1.	…	…	…	…
2.	*	x_2	x_3	x_4		2.	x_1	*	x_3	x_4
3.	…	…	…	…		3.	…	…	…	…
4.	x_1	*	x_3	x_4		4.	*	x_2	x_3	x_4
5.	…	…	…	…		5.	…	…	…	…
6.	*	x_2	x_3	x_4		6.	x_1	*	x_3	x_4
3 варіант						**6 варіант**				
1.	…	…	…	…		1.	…	…	…	…
2.	x_1	*	x_3	x_4		2.	*	x_2	x_3	x_4
3.	…	…	…	…		3.	…	…	…	…
4.	*	x_2	x_3	x_4		4.	x_1	*	x_3	x_4
5.	…	…	…	…		5.	…	…	…	…
6.	*	x_2	x_3	x_4		6.	x_1	*	x_3	x_4

Для визначення його порядкового номеру перевіряємо всі масиви обчислюючи їх хеш та порівнюючи з хешем, збереженим при розділенні файлу на залишки. Якщо шуканий масив знайдено, то він передається модулю читання/запису для формування початкового файлу. У випадку, якщо всі масиви перевірені, а хеш-сума жодного з них не відповідає початковому значенню, відповідно файл неможливо відновити.

Обговорення результатів досліджень.

Серед результатів цього дослідження обговорення потребують деякі пункти.

По перше, при розробці захищеної СРЗЗД на основі надлишкової СЗК запропоновано розглядати виявлення та виправлення лише однієї помилки, оскільки згідно із звітом Backblaze Storage Cloud за 2022 рік [15] відсоток виходу з ладу пристроїв зберігання даних (ПЗД) в незначній мірі залежить від розміру самого пристрою та значною мірою від часу експлуатації. Відповідно, для пристроїв, які безперебійно працюють більше 8 років, ймовірність виходу з ладу не перевищує 3,73% для малих накопичувачів (до 10 Тб), а для дисків розміром 12-16 Тб в середньому становить 1,07%. Це підтверджує високу надійність сучасних пристроїв зберігання даних.

По друге, у роботі пропонується застосовувати перевірку хешу для визначення цілісності початкового файлу та файлів залишків. Проте, замість перевірки хешу можна використовувати і інші методи перевірки цілісності.

Висновки. У роботі розроблено структурну схему системи розподіленого захищеного зберігання даних, яка базується на системі залишкових класів та дає змогу відновити початкову інформацію при пошкоджені або втраті частини фрагментів. Розроблено алгоритми роботи системи, а саме:

- алгоритм розділення файлу на фрагменти;
- алгоритм вибору файлів залишків;
- алгоритм зворотного перетворення;
- алгоритм модуля відновлення.

Для запропонованої системи розраховано ймовірність виправлення помилок, що можуть виникати у фрагментах даних.

Список літератури:

1. Cybersecurity and Infrastructure Security Agency CISA [Електронний ресурс] // Ransomware guide: CISA. – Режим доступу: https://www.cisa.gov/stopransomware /ransomware-guide (дата звернення: 15.03.2022).

2. How Can I Protect Against Ransomware [Електронний ресурс] // U.S. Cybersecurity and Infrastructure Security Agency. – Режим доступу: www.cisa.gov/stopransomware/how-can-i-protect-against-ransomware (дата звернення: 15.03.2022).

3. Zhang Z. A., Zhou S. Decentralized strongly secure attribute-based encryption and authentication scheme for distributed Internet of Mobile Things / Computer Networks. 2021. Vol. 201.

4. Secure the Data, Not the Device: How Decentralized File Storage Creates Resilience Against the Risk of Ransomware Attacks [Електронний ресурс] // Foundation for Defense of Democracies. – Режим доступу: https://www.fdd.org/wp-content/uploads/2021/10/fdd-memo-secure-the-data-not-the-device.pdf (дата звернення: 15.03.2022.).

5. Ali S., Wang B., White and R. L. Cottrell A Blockchain-Based Decentralized Data Storage and Access Framework for PingER / 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications / 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), New York, NY, USA, 2018, pp. 1303-1308, doi: 10.1109/TrustCom/BigDataSE.2018.00179.

6. Antonio Celesti, Maria Fazio, Massimo Villari, Antonio Puliafito. Adding long-term availability, obfuscation, and encryption to multi-cloud storage systems / Journal of Network and Computer Applications, January 2016. Vol. 59. P. 208-218.

7. Hassan S., Ibrahim R., Bingi K., Chung T. and N. Saad Application of wireless technology for control: A wirelesshart perspective / Procedia Computer Science. Vol. 105. P. 240-249.

8. Ge X, Yang and Q.-L. Han. Distributed networked control systems: A brief overview / Information Sciences. 2017. Vol. 380. P. 117-131.

9. Amor Ben R. On decentralized control techniques of interconnected systems-application to a double-parallel inverted pendulum / R. Ben Amor and S. Elloumi // 2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET). Hammamet, Tunisia. 2017. Pp. 85-90, doi: 10.1109/ASET.2017.7983671

10. Rtibi H. Robust decentralized nonlinear control for multimachine power systems / H. Rtibi and S. Elloumi // 2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET), Hammamet, Tunisia. 2017. P. 473-480, doi: 10.1109/ASET.2017.7983739.

11. Kunifuji T. Safety Technologies in Autonomous Decentralized Railway Control System/ T. Kunifuji // 2017 IEEE 13th International Symposium on Autonomous Decentralized System (ISADS), Bangkok, Thailand. 2017. P. 137-142, doi: 10.1109/ISADS.2017.15.

12. Chen, D. BOSSA: A decentralized system for proofs of data retrievability and replication / D. Chen, H. Yuan, S. Hu, Q. Wang, and C. Wang // IEEE Transactions on Parallel and Distributed Systems. 2020. Vol. 32(4). P. 786-798.

13. Xiao H. New Error Control Algorithms for Residue Number System Codes / H. Xiao, H. K. Garg, J.Hu, and G. Xiao // ETRI Journal. 2016. Vol. 38(no 2). P. 326-336.

14. Кулина С. Виявлення та виправлення помилок у захищених системах зберігання даних на основі обчислення проекцій числа / Інформатика та математичні методи в моделюванні. 2022. Т.12 (№ 4). С. 337-345.

15. Backblaze Drive Stats for 2022 [Електронний ресурс] // The Backblaze Storage Cloud.

Режим доступу:
www.backblaze.com/company/about.html (дата звернення: 15.03.2022).

References:

1. Ransomware guide: CISA. (2022). “Cybersecurity and Infrastructure Security Agency CISA” available at: www.cisa.gov/ stopransomware/ ransomware-guide (Accessed: 15.03.2023).

2. U.S. Cybersecurity and Infrastructure Security Agency. (2021). “How Can I Protect Against Ransomware” available at: https://www.cisa.gov/stopransomware/how-can-i-protect-against-ransomware (Accessed: 15.03.2023).

3. Zhang, Zhishuo; Zhou, Shijie. (2021). A decentralized strongly secure attribute-based encryption and authentication scheme for distributed Internet of Mobile Things. Computer Networks, 2021, vol. 201.

4. Ransomware guide: CISA. (2021). Cybersecurity and Infrastructure Security Agency CISA. Available at:www.cisa.gov/stopransomware/ransomware-guide (Accessed: 15.03.2023).

5. Ali, S., Wang, G., White, B., & Cottrell, R. L. (2018). "A Blockchain-Based Decentralized Data Storage and Access Framework for PingER. 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), New York, NY, USA, 2018, pp. 1303-1308, doi: 10.1109/TrustCom/BigDataSE.2018.00179

6. Antonio Celesti, Maria Fazio, Massimo Villari, Antonio Puliafito. (2016). Adding long-term availability, obfuscation, and encryption to multi-cloud storage systems Journal of Network and Computer Applications. – January 2016. Vol. 59. P. 208-218.

7. Hassan, S., Ibrahim, R., Bingi, K., Chung, T., & Saad, N. (2017). "Application of wireless technology for control: A wirelesshart perspective". Procedia Computer Science. Vol. 105, P. 240-249.

8. Ge, X., Yang, F., & Han, Q.-L. (2017) "Distributed networked control systems: A brief overview". Information Sciences. Vol. 380, P. 117-131.

9. Ben Amor, R., & Elloumi, S. (2017). "On decentralized control techniques of interconnected systems-application to a double-parallel inverted pendulum". 2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET). Hammamet, Tunisia, 2017, P. 85-90, doi: 10.1109/ASET.2017.7983671

10. Rtibi, H., & Elloumi, S. (2017). "Robust decentralized nonlinear control for multimachine power systems". 2017 International Conference on Advanced Systems and Electric Technologies (IC_ASET). Hammamet, Tunisia, 2017, P. 473-480, doi: 10.1109/ASET.2017.7983739.

11. Kunifuji, T. (2017). "Safety Technologies in Autonomous Decentralized Railway Control System". 2017 IEEE 13th International Symposium on Autonomous Decentralized System (ISADS). Bangkok, Thailand, 2017, P. 137-142, doi: 10.1109/ISADS.2017.15.

12. Chen, D., Yuan, H., Hu, S., Wang, Q., & Wang, C. (2020). BOSSA: A decentralized system for proofs of data retrievability and replication. IEEE Transactions on Parallel and Distributed Systems. Vol. 32(4), 2020. P. 786-798.

13. Xiao H., Garg, H. K., Hu, J., & Xiao, G. (2016). New Error Control Algorithms for Residue Number System Codes. ETRI Journal. 2016, Vol. 38 (2). P. 326-336.

14. Kulyna, S.V. (2022). “Error detection and correction in protected data storage systems based on the calculation of number projections” Informatics and Mathematical Methods in Simulation. Vol. 12 (2022), No. 4, P. 337-345, doi: 10.15276/imms.v12.no4.337.

15. Data is the Digital World's most precious resource (2022) Backblaze. Available at: http://www.backblaze.com/company/about. html. (Accessed: 15.03.2023).

Науково-методична стаття.
Надійшла до редакції 19.04.2023.
Прийнято до публікації 18.05.2023.