Харківський національний університет імені В. Н. Каразіна
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

Вілігура Владислав Вікторович

УДК 004.056: 004.65

ДИСЕРТАЦІЯ
МОДЕЛІ ТА МЕТОДИ ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ БАЗ ДАНИХ З УНІВЕРСАЛЬНИМ БАЗИСОМ ВІДНОШЕНЬ

Спеціальність 125 Кібербезпека та захист інформації
Галузь знань 12 Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____В. В. Вілігура

Науковий керівник: **Узлов Дмитро Юрійович**, кандидат технічних наук.

Харків – 2024

АНОТАЦІЯ

Вілігура В. В. Моделі і методи забезпечення безпеки баз даних з універсальним базисом відношень. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 125 Кібербезпека та захист інформації (Галузь знань 12 Інформаційні технології). – Харківський національний університет імені В. Н. Каразіна Міністерства освіти і науки України, Харків, 2024.

Дисертація присвячена розробці, удосконаленню та використанню моделей та методів забезпечення безпеки баз даних.

Метою дисертаційної роботи є підвищення ефективності захисту баз даних, побудованих на основі схеми з універсальним базисом відношень, шляхом розробки та застосування моделей, методів та засобів забезпечення їхньої безпеки.

У першому розділі дисертації (*Сучасний стан, проблеми та завдання забезпечення безпеки баз даних*) виконано аналіз сучасного стану, основних проблем забезпечення безпеки баз даних (БД) та постановку задач дослідження. Зокрема, було проведено аналіз підходів та досягнень у галузі забезпечення та оцінки безпеки інформаційних систем загалом та баз даних, як їх основного функціонального компонента, зокрема, в тому числі проведено аналіз формальних моделей управління доступом та забезпечення цілісності даних, як методологічної основи побудови систем захисту та оцінки їхньої безпеки. За результатами аналізу виявлено недоліки та невирішені питання, що стосуються безпеки баз даних та її оцінки, виходячи з яких сформульовано задачі дисертаційного дослідження.

У другому розділі (*Розробка моделі захисту та методу оцінки безпеки бази даних*) вирішено завдання розробки та обґрунтування моделі захисту бази даних на основі системи безпеки з повним перекриттям та методу оцінки безпеки бази даних. Завдяки розширенню моделі Клементса–Гофмана (Clements–Hoffman model) за рахунок включення безлічі вразливостей об'єктів (що дозволяє більш

адекватно оцінювати ймовірність небажаного інциденту (реалізації загрози) у двофакторній моделі), певному інтегральному показнику захищеності БД (як величини зворотної сумарному залишковому ризику, складові компоненти якої представляються у вигляді відповідних лінгвістичних змінних), розробленому методу оцінювання основних компонент бар'єрів безпеки та захищеності бази даних в цілому, що спирається на теорію нечітких множин та ризику, стає можливою кількісна оцінка безпеки бази даних, що аналізується. Отримано *перший та другий наукові результати – удосконалено*:

– модель системи безпеки з повним перекриттям Клементса–Гофмана, яка відрізняється від відомої розширеною, за рахунок доповнення моделі множиною вразливостей об'єктів, як окремо об'єктивно існуючої категорії, та конкретизованим для баз даних складом компонент, що дозволяє більш адекватно оцінювати ймовірність небажаного інциденту (реалізації загрози) та захищеність бази даних у цілому;

– метод оцінювання основних компонент бар'єрів безпеки та захищеності бази даних в цілому, який на відміну від відомих, за рахунок комплексування вдосконаленої моделі Клементса–Гофмана, введеного інтегрального показника безпеки, положень теорії нечітких множин та ризику, дозволяє адаптуватися до нових умов функціонування та прозоро, комплексно та кількісно оцінювати безпеку баз даних з різними моделями даних.

У третьому розділі (*Розробка методів маскування даних*) вирішено завдання розробки та обґрунтування методів маскування даних, що дозволяють зменшити ймовірність реалізації загрози логічного висновку та забезпечити більш ефективне приховування коду критично важливих модулів, що постійно зберігаються, яке вимагає значно більших обчислювальних і часових витрат на його розкриття зловмисником, ніж при використанні існуючих способів, що надаються розробниками деяких сучасних систем керування базами даних (СКБД). Отримано *третій, четвертий та п'ятий наукові результати*:

удосконалено метод маскування MOBAT, що відрізняється від відомого, можливістю обфускації (англ. obfuscation) даних, шляхом математичних

перетворень на основі обчислення операцій за модулем, що застосовуються до елементів даних не тільки числового, а й широко поширеного в базах даних рядкового типу, що дозволяє суттєво розширити охоплення різноманітних маскованих з метою утруднення реалізації зловмисником загрози умовиводу даних БД;

отримали подальший розвиток:

– метод маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних, що відрізняється від відомих оригінальним підходом до процесу перемішування з можливістю випадкової заміни елементів даних різного типу всередині заданого поля рядка та використання технології динамічного маскування, що дозволяє при менших обчислювальних витратах на перетворення і без зміни формату вихідних даних забезпечити ефективне приховування даних, яке ускладнює реалізацію загрози логічного висновку;

– метод приховування коду збережених у базі даних програм, який на відміну від відомих, дозволяє за рахунок випадкової перестановки (що спирається на сучасний варіант алгоритму тасування Фішера-Йейтса) символів коду з можливою заміною кожного з них на інший випадково вибраний із стандарту Unicode забезпечити більше ефективний (якій вимагає значно більших обчислювальних витрат) захист коду від його розкриття зловмисником, при цьому гарантуючи цілісність коду.

Отримано *перший практичний результат*: розроблений метод маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних, орієнтований на заплутування, псевдонімізацію (англ. pseudonymisation) даних та ускладнення реалізації загрози логічного висновку, дозволяє зменшити час на відповідні операції перетворення на (10-17) % щодо методу класичного шифрування, при цьому не наводячи до зміни формату та збільшення розмірності даних, що зберігаються. Даний метод може бути також використаний у невиробничих базах даних, розширюючи можливості так званого статичного маскування даних.

У четвертому розділі (*Розробка методу контролю цілісності та справжності збережених програм, заснованого на можливостях технології блокчейн*) вирішено завдання розробки та обґрунтування методу контролю цілісності та справжності модулів, що постійно зберігаються, заснованого на можливостях технології блокчейн. Отримано *шостий науковий результат*: *вперше* запропоновано метод моніторингу, що ґрунтується на можливостях технології блокчейн, який на відміну від відомих дозволяє за рахунок використання створеної зумовленої структури, правил формування первинного та наступних блоків у блокчейновому ланцюжку, організації зберігання цієї структури в рамках реляційної моделі даних, способів обчислення кореня геш-дерева, суворо контролювати набір програм БД, їх цілісність, справжність при менших обсягах збережених для цього даних і необхідних ресурсів процесора.

Отримано *другий практичний результат*: запропонований метод моніторингу модулів БД, що постійно зберігаються, вимагає менших обсягів збережених для цього даних і ресурсів процесора, ніж відомий метод контрольних сум, який для підтримки аналогічного контролю цілісності та справжності PSM вимагає виконання процедур гешування та цифрового підпису із збереженням відповідних даних для кожного конкретного PSM у конкретній схемі БД, причому однаково, не забезпечуючи контроль всього набору PSM загалом.

У п’ятому розділі (*Реалізація методів і засобів захисту в базах даних, побудованих на основі схеми з універсальним базисом відношень*) вирішено завдання обґрунтування та систематизації реалізованих заходів захисту, що забезпечують конфіденційність, цілісність даних та постійно збережених модулів баз даних з універсальним базисом відношень. Ці заходи ґрунтуються як на загальних формальних моделях управління доступом, забезпечення цілісності даних, методах, засобах, механізмах, що підтримуються СКБД, на платформі якої запропонована схема реалізується, так і на власних, розроблених у рамках створення інваріантної до предметних областей схеми БД. Отримано *третій практичний результат*: розроблені в процесі створення схеми БД з УБВ спеціальні заходи у вигляді відповідних методів, реалізованих об'єктів схеми

(тригерів, процедур, пакетів, таблиць, функцій) та правил їх використання підвищують безпеку таких БД, забезпечуючи високий ступінь контрольованості доступу до даних (аж до конкретного елемента), необхідну конфіденційність, цілісність даних та об'єктів схеми БД, на відміну від традиційних реляційних БД, що не володіють подібними заходами та функціональністю.

У шостому розділі (*Оцінка безпеки бази даних з універсальним базисом відношень*) здійснено оцінку безпеки бази даних з універсальним базисом відношень та наведено порівняльний аналіз захищеності баз даних, побудованих за традиційною технологією та на основі універсального базису відношень. Порівняльний аналіз показав, що використання запропонованих у роботі рішень дозволить підвищити ефективність / результативність захисту баз даних, побудованих на основі схеми з універсальним базисом відношень, більш ніж у 1.5 рази щодо традиційних реляційних БД.

Ключові слова: інформаційна система, база даних, база даних з універсальним базисом відношень, об'єкт бази даних, безпека, кібербезпека, модель безпеки, модель системи захисту з повним перекриттям, ризик, генератор псевдовипадкових чисел, маскування даних, шифрування, криптографічний захист інформації, цілісність, блокчейн.

ABSTRACT

Vilihura, V. V. Models and methods for ensuring the security of databases with the universal basis of relations. – Qualifying scientific work as a manuscript.

Thesis for the degree of Doctor of Philosophy in specialty 125 Cybersecurity and information protection (Field of knowledge 12 Information Technology). – V. N. Karazin Kharkiv National University of the Ministry of Education and Science of Ukraine, Kharkiv, 2024.

The thesis is devoted to the development, improvement and use of models and methods for ensuring database security.

The purpose of the thesis is to increase the efficiency of protection of databases built based on the scheme with the universal basis of relations, through the development and application of models, methods and means of ensuring their security.

In the first section of the thesis (*Current State, problems and tasks of database security*) the analysis of the current state, the main problems of ensuring the security of databases (DB) and the formulation of research tasks are carried out. In particular, an analysis of approaches and achievements in the field of ensuring and assessing the security of information systems in general and databases, as their main functional component, was carried out, in particular, an analysis of formal models of access control and ensuring the integrity of data as a methodological basis for building protection systems and assessing their security. Based on the results of the analysis, disadvantages and unresolved issues related to the security of databases and its assessment have been identified, on the basis of which the tasks of the thesis research have been formulated.

In the second section (*Development of the protection model and the method for assessing database security*), the problem of developing and justifying the database security model based on a full overlap security system and the database security assessment method is solved. Due to the extension of the Clements–Hoffman model due to the inclusion of a set of vulnerabilities of objects (which allows for a more adequate assessment of the probability of an undesirable incident (threat realization) in a two-

factor model), a certain integral indicator of database security (as the inverse of the total residual risk, the constituent components of which are represented in the form of corresponding linguistic variables), the developed method for assessing the main components of security barriers and security of the database as a whole, based on the theory of fuzzy sets and risk, it becomes possible to quantify the security of the analyzed database. The *first and second scientific results were obtained – improved*:

– the model of the security system with a complete overlap of Clements-Hoffmann, which differs from the well-known extended one, due to the addition of the model with a set of vulnerabilities of objects, as a separate objectively existing category, and a specified composition of components for databases, which allows for a more adequate assessment of the probability of an unwanted incident (threat implementation) and the security of the database as a whole;

– the method for assessing the main components of security barriers and security of the database as a whole, which, unlike the known ones, due to the integration of the improved Clements-Hoffmann model, the introduced integral security indicator, the provisions of the theory of fuzzy sets and risk, allows adapting to new operating conditions and transparently, comprehensively and quantitatively assessing the security of databases with different data models.

The third section (*Development of data masking methods*) solves the problem of developing and justifying data masking methods that reduce the probability of the threat of logical inference and ensure more effective hiding of the code of critically stored modules, which requires much more computational and time costs for its disclosure by an attacker than when using the existing methods provided by the developers of some modern database management systems (DBMS). The *third, fourth and fifth scientific results were obtained*:

the method of masking MOBAT, which differs from the known one, in the possibility of obfuscation of data, *has been improved* by means of mathematical transformations based on the calculation of operations by modulus, applied to data elements not only numeric, but also widespread in databases of string type, which allows

to significantly expand the coverage of various masked in order to complicate the threat of inference of database data by an attacker;

have been further developed:

– the method of masking data elements of non-key fields of tuples of tables of the production database, which differs from the known ones in the original approach to the shuffle process with the possibility of random replacement of data elements of different types within a given field of the row and the use of dynamic masking technology, which allows with lower computing costs for transformation and without changing the format of the original data to ensure effective data hiding, which makes difficult to implement the threat of a logical conclusion;

– the method of hiding the code of programs stored in the database, which, unlike the known ones, allows, due to the random rearrangement (based on the modern version of the Fisher-Yates shuffle algorithm) of the code symbols with the possible replacement of each of them with another randomly selected from the Unicode standard, to provide more effective (which requires much higher computing costs) protection of the code from its disclosure by an attacker, while guaranteeing the integrity of the code.

The first practical result was obtained: the developed method of masking data elements of non-key fields of tuples of tables of the production database, focused on obfuscation, pseudonymisation of data and complication of the implementation of the threat of logical inference, allows to reduce the time for the corresponding conversion operations by (10-17)% regarding the method of classical encryption, while not leading to a change in the format and an increase in the dimension of the stored data. This method can also be used in non-production databases, expanding the possibilities of so-called static data masking.

In the fourth section (*Development of the method for controlling the integrity and authenticity of stored programs based on the capabilities of blockchain technology*), the problem of developing and substantiating a method for controlling the integrity and authenticity of permanently stored modules, based on the capabilities of blockchain technology, is solved. *The sixth scientific result was obtained: for the first time* a method of monitoring based on the capabilities of blockchain technology is proposed, which,

unlike the known ones, allows, through the use of the created predetermined structure, rules for the formation of the primary and subsequent blocks in the blockchain chain, the organization of storage of this structure within the framework of a relational data model, methods of calculating the root of the hash tree, to strictly control the set of database programs, their integrity, authenticity with smaller volumes of data stored for this and the necessary resources processor.

The *second practical result is obtained*: the proposed method of monitoring permanently stored database modules requires smaller amounts of data and processor resources stored for this purpose than the well-known method of checksums, which, in order to maintain similar control over the integrity and authenticity of PSM, requires the execution of hashing and digital signature procedures with the preservation of the corresponding data for each specific PSM in a specific database schema, and in the same way, without providing control of the entire PSM set as a whole.

In the fifth section (*Implementation of methods and means of protection in databases built based on the scheme with the universal basis of relations*) the problem of substantiation and systematization of the implemented protection measures that ensure confidentiality, data integrity and permanently stored database modules with the universal basis of relations is solved. These measures are based on both general formal models of access control, data integrity, methods, means, mechanisms supported by the DBMS, on the platform of which the proposed scheme is being implemented, and on our own, developed within the framework of creating a database scheme invariant to subject areas. The *third practical result* was obtained: special measures developed in the process of creating a database schema with UBV in the form of appropriate methods, implemented schema objects (triggers, procedures, packages, tables, functions) and rules for their use increase the security of such databases, providing a high degree of control of access to data (up to a specific element), the necessary confidentiality, integrity of data and objects of the database scheme, in contrast to traditional relational databases, that do not have such measures and functionality.

In the sixth section (*Assessment of the security of database with the universal basis of relations)* an assessment of the security of a database with the universal basis of

relations is carried out and a comparative analysis of the security of databases built on the traditional technology and on the basis of the universal basis of relations is provided. A comparative analysis has shown that the use of the solutions proposed in the work will increase the efficiency / effectiveness of the protection of databases built on the basis of the scheme with the universal basis of relations, more than 1.5 times relative to traditional relational databases.

Keywords: information system, database, database with the universal basis of relations, database object, security, cyber security, security model, full overlap security system model, risk, pseudorandom number generator, data masking, encryption, information cryptographic protection, integrity, blockchain.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові публікації у фахових виданнях України:

1. Yesin V. I., Vilihura V. V. Some approach to data masking as means to counter the inference threat. *Radiotekhnika.* 2019. № 198. P. 113–130.

(Особистий внесок здобувача: розроблені метод маскування даних на основі обчислення операцій за модулем, метод маскування даних поля рядка таблиці бази даних).

2. Вілігура В. В., Горбенко Ю. І., Єсін В. І., Рассомахін С. Г. Використання формальних моделей безпеки в захищених базах даних. *Фізико-математичне моделювання та інформаційні технології*. 2021. № 32. С. 70–74.

(Особистий внесок здобувача: аналіз моделей безпеки на основі дискреційної, мандатної, рольової політики із зазначенням рекомендацій щодо їх застосування при розробці захищених баз даних).

3. Єсін В. І., Вілігура В. В. Дослідження основних методів і схем шифрування з можливостю пошуку. *Радіотехніка*. 2022. № 209. С. 138–155.

(Особистий внесок здобувача: аналіз моделей та архітектур існуючих захищених пошукових систем з урахуванням особливостей сценаріїв їхнього функціонування; порівняльний аналіз продуктивності відомих класичних схем симетричного шифрування з можливістю пошуку).

4. Єсін В. І., Вілігура В. В. Дослідження основних схем шифрування з можливістю пошуку у базах даних, які підтримують SQL. *Радіотехніка*. 2022. № 210. С. 53–74.

(Особистий внесок здобувача: аналіз основних систем шифрування з можливістю пошуку в базах даних, які підтримують мову структурних запитів SQL, з метою виявлення слабких і сильних сторін аналізованих систем та реалізованих у них методів для визначення можливості практичного їх використання в конкретних умовах функціонування).

5. Єсін В. І., Вілігура В. В. Основні категорії NewSQL баз даних та їх особливості. *Радіотехніка*. 2022. № 211. С. 37–66.

(Особистий внесок здобувача: аналіз важливих характеристик (зокрема безпеки), властивих NewSQL, традиційним реляційним і NoSQL системам баз даних, із зазначенням рекомендацій щодо їх застосування у майбутньому, на наступному витку спіралі розвитку технологій баз даних).

6. Єсін В. І., Вілігура В. В., Сватовський І. І. Забезпечення безпеки у розподілених інформаційних системах: основні аспекти. *Радіотехніка*. 2023. Вип. 214. С. 32–63.

(Особистий внесок здобувача: аналіз актуальних сучасних методів, прийомів та засобів забезпечення безпеки розподілених інформаційних системах та їх основного функціонального компонента – бази даних).

7. Єсін В. І., Вілігура В. В., Узлов Д. Ю. Огляд існуючих моделей та основних принципів нульової довіри. *Радіотехніка*. 2024. Вип. 217. С. 39–54.

(Особистий внесок здобувача: аналіз моделей та ключових принципів концепції нульової довіри, як нового фундаментального підходу до інформаційної безпеки, кібербезпеки).

Наукова публікація у зарубіжних виданнях, що входять до міжнародних наукометричних баз Scopus та Web of Science***:***

8. Yesin V. I., Vilihura V. V. Method for development of databases easily adaptable to variations in the subject domain. *Telecommunications and Radio Engineering*. 2019. № 78(7). P. 595–605. (Scopus).

(Особистий внесок здобувача: оцінка завершеності створення бази даних, що відповідає заданим вимогам, на основі аналізу певних значень показників якості).

9. Yesin V. I., Karpinski M., Yesina M. V., Vilihura V. V. Formalized representation for the data model with the universal basis of relations. *International Journal of Computing*. 2019. № 18(4). P. 453–460. (Scopus).

(Особистий внесок здобувача: розробка принципів побудови алгоритму відображення концептуальної моделі предметної області у відношення універсального базису, що дозволяє в умовах динамічних змін предметних

областей спростити процес створення логічних схем реляційних баз даних, що задовольняють заданим вимогам).

10. Yesin V. I., Yesina M. V., Vilihura V. V. Monitoring the integrity and authenticity of stored database objects. *Telecommunications and Radio Engineering*. 2020. № 79(12). P. 1029–1054. (Scopus).

(Особистий внесок здобувача: визначення структур блоків у блокчейновому ланцюжку та методів обчислення кореня геш-дерева, що формується на основі бінарних дерев різних типів, розробка принципів відображення структури блокчейну у відношення реляційної моделі даних, що дозволяє забезпечувати своєчасний моніторинг цілісності, справжності об'єктів, що зберігаються в базі даних при меншому обчислювальному ресурсі і меншій надмірності необхідних для цього даних, порівняно з існуючими методами).

11. Yesin V., Karpinski M., Yesina M., Vilihura V., K. Warwas K. Hiding the Source Code of Stored Database Programs. *Information*. 2020. № 11(12). 576. (Scopus, Web of Science).

(Особистий внесок здобувача: розробка методу та основних принципів побудови алгоритмів маскування вихідного коду збережених програм та дослідження їх властивостей).

12. Yesin V., Karpinski M., Yesina M., Vilihura V., Warwas K. Ensuring Data Integrity in Databases with the Universal Basis of Relations. *Applied Sciences*. 2021. № 11(18). 8781. (Scopus, Web of Science).

(Особистий внесок здобувача: аналіз моделі Кларка-Вілсона та розробка на основі її рекомендацій методів та засобів, що забезпечують цілісність основних компонентів бази даних з універсальним базисом відношень).

13. Yesin V., Karpinski M., Yesina M., Vilihura V., Rajba S. A. Technique for Evaluating the Security of Relational Databases Based on the Enhanced Clements–Hoffman Model. *Applied Sciences*. 2021. № 11(23). 11175. (Scopus, Web of Science).

(Особистий внесок здобувача: розробка вдосконаленої моделі Клементса-Хоффмана та методу оцінювання захищеності бази даних, дослідження безпеки реляційних баз даних, спроектованих за різними технологіями).

14. Yesin V., Karpinski M., Yesina M., Vilihura V., Kozak R., Shevchuk R. Technique for Searching Data in a Cryptographically Protected SQL Database. *Applied Sciences*. 2023. № 13(20). 11525. (Scopus, Web of Science)

(Особистий внесок здобувача: розробка методики пошуку в криптографічно захищеній базі даних, що дозволяє серверу СКБД виконувати функції пошуку за зашифрованими даними так само, як і в незашифрованій базі даних, і що забезпечує належну конфіденційність даних, що зберігаються, при прийнятних накладних витратах).

Наукові праці, які засвідчують апробацію матеріалів дисертації:

15. Вілігура В. В., Єсін В. В. Використання національного криптоалгоритму для захисту персональних даних в СУБД Oracle. *Комп'ютерне моделювання у наукоємних технологіях (КМНТ-2016)*: Праці науково-технічної міжнародної конференції, 26-31 травня 2016 р. Харків: Харківський національний університет імені В. Н. Каразіна, 2016. С. 77–80.

(Особистий внесок здобувача: оцінка можливості та доцільності використання національного криптоалгоритму «Калина» для захисту персональних даних в СКБД Oracle).

16. Yesin V. I., Karpinski M., Yesina M. V., Vilihura V. V., Veselska O., L. Wieclaw L. Approach to Managing Data From Diverse Sources. *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*: Proceedings of the 2019 10th IEEE International Conference, 18-21 September, 2019, Metz, France, Volume 1, P. 1–6. (Scopus, Web of Science).

(Особистий внесок здобувача: розробка методу, операції якого спрямовані на підготовку інформаційних продуктів та конфігураційної бази даних середовища управління простором даних компанії до використання).

17. Вілігура В. В. Систематизація загроз і вразливостей характерних для баз даних і СУБД. *Праці 7-ої Міжнародній конференції «Комп'ютерне моделювання в наукоємних технологіях (КМНТ-2021)*, 21-23 квітня 2021 р. Харків: Харківський національний університет імені В. Н. Каразіна, 2021. С. 83–86.

Наукові публікації, що додатково відображають зміст дисертації:

18. Advances in Information Security and Privacy. In: Lax G., Russo A. (eds). MDPI: Basel, Switzerland. 2022. 344 p. (Yesin V., Karpinski M., Yesina M., Vilihura V., Rajba S. A., Warwas K. P. 257–294). ISBN 978-3-0365-5296-5 (hardback); ISBN 978-3-0365-5295-8 (PDF). https://doi.org/10.3390/books978-3-0365-5295-8.

(Особистий внесок здобувача: аналіз моделей Кларка-Вілсона та Клементса-Хоффмана із розробкою методів та засобів, що забезпечують цілісність основних компонентів бази даних з універсальним базисом відношень, та дозволяють оцінювати захищеність баз даних, спроектованих за різними технологіями).

19. Mathematics and Computer Science – Contemporary Developments. In: El-Sayed Mohamed Abo-Dahab Khedary (eds). BP International: Hooghly, West Bengal, India, 2024. Vol. 1. 95 p. (Yesin V., Karpinski M., Yesina M., Vilihura V., Kozak R., Shevchuk R. Introducing a Technique for Searching Data in a Cryptographically Protected SQL Database. P. 1–29.). ISBN 978-81-977283-5-8 (Print), ISBN 978-81-977283-6-5 (eBook). https://doi.org/10.9734/bpi/mcscd/v1.

(Особистий внесок здобувача: розробка методики пошуку даних у криптографічно захищеній базі даних із збереженням конфіденційності даних, що зберігаються, при прийнятних накладних витратах).

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

ACID – Atomicity, Consistency, Isolation, Durability

BLOB – Binary Large Object

CIA – Confidentiality, Integrity, Availability

CLOB – Character Large Object

CWE – Common Weakness Enumeration

DISA – Defense Information Systems Agency

EISP – Enterprise Information Security Policies

FGAC – Fine Grained Access Control

FPE – Format-Preserving Encryption

HIPAA – Health Insurance Portability and Accountability Act

ISSP – Issue-Specific Security Policies

GDPR – General Data Protection Regulation

ID – Identifier

IMA – Inverse masking algorithm

MA – Masking algorithm

MAC – Message Authentication Code

MOBAT – Modulus Based Technique

NIST – National Institute of Standards and Technology

OLS – Oracle Label Security

PCI DSS – Payment Card Industry Data Security Standard

PK – Primary Key

PSM – Persistent Stored Modules

PRNG – Pseudorandom Number Generators

RAID – Redundant Array of Independent Disks

RBAC – Role-Based Access Control

RDBMS – Relational Database Management System

RLS – Row-Level Security

SQL – Structured Query Language

SysSP – Systems-Specific Security Policies

TAM – Type Access Matrix

TDE – Transparent Data Encryption

UML – Unified Modeling Language

VPD – Virtual Private Database

XML – Extensible Markup Language

АБД – адміністратор бази даних

БД – база даних

БД з УБВ – база даних з універсальним базисом відношень

ГПВЧ – генератор псевдовипадкових чисел

ВК – відкритий ключ

ЗІ – захист інформації

ЗК – закритий ключ

ІБ – інформаційна безпека

ІС – інформаційна система

ІСОУ – інформаційна система організаційного управління

ІТ – інформаційна технологія

ЛКГ – лінійний конгруентний генератор

ММД – мова моделі даних

НСД – несанкціонований доступ

ОС – операційна система

ПЗ – програмне забезпечення

ПрО – предметна область

РБД – реляційна БД

РСКБД – реляційна СКБД

СД – сховище даних

СКБД – система керування базами даних

СУІБ – система управління інформаційною безпекою

УБВ – універсальний базис відношень

ВСТУП

Обґрунтування вибору теми дослідження. У сучасному світі інформація перетворилася на один з найважливіших ресурсів суспільства, а інформаційні системи (ІС), основним функціональним компонентом яких є бази даних (БД), стали необхідним інструментом практично у всіх сферах діяльності людини, надаючи їй достовірну інформацію для прийняття оптимального рішення. При цьому вдосконалення технологій баз даних це напрям, що динамічно розвивається, дослідження в якому не припиняються, а ведуться з наростаючою інтенсивністю, так як з часом змінюються середовища та умови функціонування систем баз даних, удосконалюються апаратні засоби та засоби програмування, з'являються нові сфери застосунків, змінюються їх характер та вимоги. Зростання Великих Даних (Big Data) та бачення світу, керованого даними, відкривають багато цікавих можливостей, одночасно виявляючи безліч невирішених проблем [1, 2]. У тому числі, нова епоха Big Data, що залучила багатьох дослідників у «гру управління даними» і змусила відмовитися від звичних способів проектування, розробки та впровадження рішень управління даними, загострила проблему безпеки даних. Так як зріс інтерес до інформації, що циркулює всередині ІС, не тільки з боку законних користувачів і власників, але і з боку зловмисників. Для останніх бази, сховища даних, як найважливіший інформаційний ресурс, є одним із найуразливіших і найпривабливіших елементів ІС. За своєю природою, як неодноразово заявляв відомий учений, спеціаліст у галузі безпеки Росс Андерсон, «великі бази даних ніколи не будуть вільні від зловживань… якщо ви проектуєте велику систему для полегшення доступу, вона стає небезпечною, а якщо ви робите її водонепроникною (англ. watertight), її неможливо використовувати» [3].

Усе це змушує шукати нові підходи ефективного вирішення новостворених і традиційних проблем: організації архітектури систем баз даних, методів доступу; вдосконалення технологій та методів моделювання даних; розвитку функціональності користувальницьких інтерфейсів; інтеграції інформаційних ресурсів; удосконалення методології розробки застосунків; забезпечення безпеки

даних тощо. При цьому слід враховувати і виходити з того, що для різних конкретних цілей необхідні різні системи як діючі, так і перспективні, що дозволяють вирішувати актуальні проблеми користувачів цих систем в умовах обмежених ресурсів, що виділяються. Разом з тим, як зазначають фахівці всесвітньо відомої компанії Gartner, Inc. [4] вибір продукту, заснований на його особливій популярності, може бути дуже недалекоглядним.

Проведені дослідження актуального стану інформатизації у різних компаніях, організаціях, установах свідчать, що у багатьох із них сьогодні експлуатуються різнопланові інформаційні системи організаційного управління (ІСОУ), призначені для автоматизації функцій управлінського персоналу, як промислових підприємств, так і непромислових організацій. Основними функціями подібних систем є: оперативний контроль та регулювання, оперативний облік та аналіз, перспективне та оперативне планування, бухгалтерський облік, управління збутом та постачанням та інші економічні та організаційні завдання. При цьому для вирішення нових завдань, пов'язаних з розширенням сфери діяльності та, відповідно, предметних областей (ПрО), у них виникає потреба у більш функціональних, з покращеними характеристиками якості інформаційних систем, у тому числі що забезпечують високі вимоги безпеки даних, що зберігаються, та в сукупності, що вимагають менших витрат на супровід.

У цих умовах затребуваними стають проекти з розробки нових ІСОУ та їх інтеграції з інформаційними системами, що існують; з розробки нових ІСОУ з метою заміни існуючих ІС; модернізації існуючих ІСОУ. Суть даних проектів полягає у систематичній трансформації існуючої системи – проведенні процедур реінжинірингу існуючих ІС та їх основного функціонального компонента – бази даних. При цьому однією з важливих вимог, що висуваються до процесу реінжинірингу існуючих ІСОУ та їх баз даних, є своєчасність завершення відповідних проектів з інформаційних технологій (ІТ) у рамках запланованого бюджету із заданими характеристиками якості. Проте, як свідчать результати аналізу ІТ-проектів [5-7], багато проектів було провалено або завершено із

запізненням, причому з набагато більшими витратами, ніж планувалося. Це, як правило, пов'язане з обмеженістю функціональності відповідних методів проектування. У контексті баз даних, конкретніше реляційних баз даних (РБД), як тих, що отримали найбільше поширення (цю тезу підтверджують результати DB-Engines і PYPL (PopularitY of Programming Language) рейтингів [8, 9], а також звіти експертів Gartner, Inc.), в тому числі і в IC класу, що розглядається, зазначена обмеженість обумовлена орієнтацією традиційної методології їх проектування на ітераційну, досить складну та трудомістку процедуру створення унікальних концептуальної моделі, логічної та фізичної схем при розробці нової БД, або на істотне їх перетворення при модернізації. Що часто спричиняє значні, не завжди прогнозовані об'єктивні витрати часових та фінансових ресурсів. У зв'язку з чим для вирішення цієї проблеми в роботах [10-15] було методологічно обґрунтовано та запропоновано використання баз даних, побудованих на основі схеми з універсальним базисом відношень. Однак, хоча в частині цих робіт і були порушені питання присвячені необхідності забезпечення безпеки даних, що зберігаються в ній, але глибокого опрацювання ця проблема в них не отримала.

На сучасному етапі розвитку теорії та практики забезпечення безпеки інформації, у тому числі в базах та сховищах даних, склалася суперечлива ситуація, коли з одного боку є підвищена увага до цих питань, що виражається:

1) у виконанні великої наукової та практичної роботи зі створення, організації та дослідження процесів функціонування, удосконалення та розвитку систем захисту інформації, що була проведена і тієї, що проводиться сьогодні вітчизняними та зарубіжними вченими, серед яких R. J. Anderson, E. Bertino, L. J. Hoffman, W. Mao, C. P. Pfleeger, R. S. Sandhu, B. Schneier, І. Д. Горбенко, В. О. Хорошко, В. В. Домарєв, О. Г. Корченко та багатьох інших;

2) у постійному зростанні асигнувань на забезпечення захисту. За оцінками експертів Gartner, Inc. [16], витрати кінцевих користувачів на ринку інформаційної безпеки та управління ризиками, будуть збільшуватися із середнім річним темпом зростання в 10.8 % з 2020 по 2025 рік і досягнуть 228 мільярдів доларів США;

3) у прийнятті великої кількості різних міжнародних, вітчизняних стандартів та інших законодавчих актів у галузі інформаційної безпеки (ІБ), що передбачають високі вимоги до захисту інформації (ЗІ) в інформаційних системах, що створюються та експлуатуються, та штрафи за їх невиконання. Наприклад, за недотримання положень Загального регламенту захисту персональних даних Європейського Союзу (General Data Protection Regulation – GDPR) передбачаються серйозні штрафні санкції до компаній-власників баз даних до 20 мільйонів євро або до 4% загального річного обороту попереднього фінансового року, залежно від того що вище [17]. За умисне розкриття персональних даних про здоров'я (з метою продажу, передачі або використання інформації) відповідно до Закону про переносимість та відповідальність медичного страхування (HIPAA) передбачається штраф до 250 тис. доларів США, позбавлення волі до 10 років або те й інше [18].

Що в цілому має покращити ситуацію із захищеністю ІС та їх основного функціонального компонента – БД.

З іншого боку, спостерігається постійне зростання завданих власникам інформаційних ресурсів збитків (тобто дія породжує протидію), про що свідчать дані, що регулярно публікуються авторитетними експертами. Згідно зі статистикою, останніми роками у світі неухильно зростає кількість витоків та обсяг скомпрометованих даних. Згідно з дослідженнями Dell Technologies [19-21] було виявлено, що значне зростання породило закономірні складнощі, пов'язані з проблемами забезпечення безпеки. Найбільшу небезпеку для даних становить зростаюча кількість інцидентів: від кібератак до втрати даних та простою систем. Зловмисники продовжують удосконалювати шкідливе програмне забезпечення (ПЗ) і виводять його на нові рівні складності та сили ураження. При цьому вони все більше починають використовувати шифрування, щоб уникнути виявлення. Зростання і різноманітність типів і сімейств шкідливого ПЗ продовжують посилювати хаос у ландшафті атак і практично зводять нанівець зусилля захисників щодо запобігання і стримування загроз [22]. Шкідливе ПЗ – це інноваційна загроза, що швидко розвивається [23]. Також у наші дні серйозною

проблемою є атаки, пов'язані зі зловживанням законним програмним забезпеченням хакерами, які намагаються сховатися за нормальною активністю користувачів і систем. Зростаючі можливості кібер-злочинних груп, що використовують нові техніки злому, а також складність виявлення таких проникнень звичайними засобами викликає серйозну занепокоєність у департаментах IT-безпеки. Ця проблема посилюється, до того ж, гострою нестачею кваліфікованих спеціалістів з питань безпеки. Так експерти Panda Security [24] вважали, що до 2021 дефіцит таких фахівців може становити 3,5 мільйона осіб. Кіберзлочинці вдаються до більш витончених методів атак, здійснюючи їх, наприклад, навіть за допомогою пристроїв, що застосовуються для контролю та забезпечення безпеки, або, адаптуючи шкідливі програми на базі відкритих вихідних кодів, щоб перетворити їх на загрози [25].

З усього вищесказаного стає очевидним, що сучасні підходи до забезпечення безпеки інформації не повною мірою відповідають відповідним вимогам щодо її захисту. Зокрема, без необхідного захисту баз і сховищ даних, разом із відповідними чутливими даними, нові інформаційні технології здатні порушити як приватне життя людей, а й діяльність різних великих організацій.

У ситуації, що склалася, беручи до уваги сучасний стан розвитку технологій баз, сховищ даних, у тому числі побудованих на основі схеми з універсальним базисом відношень, науково-практичні досягнення в галузі ІБ, кваліфікацію зловмисників, які постійно вдосконалюють можливості відповідного впливу за допомогою шкідливого програмного забезпечення, положення та рекомендації різних нормативно-правових актів, доцільним є перегляд підходу до вирішення проблеми управління даними та забезпечення їх безпеки, результатом якого були певні методи, прийоми, засоби, актуальні як у теоретичному, так і в прикладному аспектах.

При цьому, доцільність досліджень саме баз даних з універсальним базисом відношень (УБВ) обумовлена тим, що, по-перше, це дозволить переконатися в безпеці даних, що зберігаються і оброблюються в них, а також показати певні переваги в захищеності таких БД перед традиційними реляційними базами даних

ІСОУ. По-друге, на їх прикладі, через те, що бази даних з УБВ можуть використовуватися в різній якості – як звичайна БД, сховище даних різних предметних областей (ПрО) або конфігураційна БД середовища управління простором даних [10-15, 26], застосовуючи певні нові підходи, стає можливою розробка деякого цілісного рішення, що забезпечує безпеку реляційних баз даних. Окремі елементи такого рішення можуть бути використані для захисту баз та сховищ даних із різними моделями (реляційними, NoSQL, NewSQL [7, 27-33]).

Все вищенаведене дозволяє зробити висновок про актуальність і своєчасність рішення науково-прикладного завдання, яке полягає в розробці та застосуванні моделей, методів і засобів, що дозволяють підвищити захищеність баз даних, побудованих на основі схеми з універсальним базисом відношень.

Зв'язок роботи з науковими програмами, планами, темами.

Дисертаційні дослідження проводились в рамках науково-дослідницьких робіт: № 39-18 «Механізми, методи, протоколи та засоби криптографічного захисту інформації у пост квантовий період» (Шифр «Квант-2019»), № 29-19 «Механізми та засоби електронного підпису у пост квантовий період», (Шифр «Квант-2020»), № 28-20 «Механізми та засоби асиметричних криптоперетворень у постквантовий період» (Шифр «Квант-2021»), «Математичні та програмні моделі, методи та механізми криптографічного захисту інформації для пост-квантового середовища в інтересах національної безпеки держави» (№ ДР 0121U109939), № 34-21 «Методи та алгоритми постквантових криптоперетворень, їх стандартизація та впровадження» (Шифр «Квант-2022»), № 09-22 «Методи та засоби генерування псевдовипадкових та випадкових послідовностей на основі класичних та квантових ефектів» (Шифр «Квант-2023»).

Мета і завдання дослідження. Метою дисертаційної роботи є підвищення ефективності захисту баз даних, побудованих на основі схеми з універсальним базисом відношень, шляхом розробки та застосування моделей, методів та засобів забезпечення їхньої безпеки.

Для досягнення поставленої мети були сформульовані та вирішені такі завдання:

1. Аналіз сучасного стану та основних проблем забезпечення безпеки баз даних. Обґрунтування напряму досліджень.

2. Розробка та обґрунтування моделі захисту бази даних на основі системи безпеки з повним перекриттям та методу оцінки безпеки БД.

3. Розробка та обґрунтування методів маскування даних, що дозволяють зменшити ймовірність реалізації загрози логічного висновку та приховати код критично важливих модулів, що постійно зберігаються.

4. Розробка та обґрунтування методу контролю цілісності та справжності модулів, що постійно зберігаються, заснованого на можливостях технології блокчейн.

5. Обґрунтування та систематизація реалізованих заходів захисту, що забезпечують конфіденційність, цілісність даних та постійно збережених модулів баз даних з універсальним базисом відношень.

6. Оцінка безпеки бази даних із універсальним базисом відношень.

Об'єкт дослідження – процес забезпечення безпеки баз даних в умовах реалізації різних типів загроз.

Предмет дослідження – моделі та методи управління доступом, забезпечення конфіденційності, цілісності та справжності даних, що зберігаються в базах даних з універсальним базисом відношень.

Методи дослідження.

Методи досліджень визначені сутністю розв'язуваних задач і включають основні положення теорії множин, у тому числі нечітких, теорії графів, баз даних, розділи сучасної криптографії, методи математичної логіки та статистики, методи маскування, що знайшли застосування у певних класах завдань із приховування інформації в базах та сховищах даних, сучасний варіант алгоритму тасування Фішера-Йейтса, методи верифікації блоків даних у сучасній блокчейновій моделі.

Наукова новизна одержаних результатів. У результаті вирішення поставлених завдань були отримані наступні нові наукові результати.

1. Вперше запропоновано метод моніторингу збережених програм, що ґрунтується на можливостях технології блокчейн, який на відміну від відомих

дозволяє за рахунок використання створеної зумовленої структури, правил формування первинного та наступних блоків у блокчейновому ланцюжку, організації зберігання цієї структури в рамках реляційної моделі даних, способів обчислення кореня геш-дерева, суворо контролювати набір програм БД, їх цілісність, справжність при менших обсягах збережених для цього даних і необхідних ресурсів процесора.

2. Удосконалено:

– модель системи безпеки з повним перекриттям Клементса–Гофмана, яка відрізняється від відомої розширеною, за рахунок доповнення моделі множиною вразливостей об'єктів, як окремо об'єктивно існуючої категорії, та конкретизованим для баз даних складом компонент, що дозволяє більш адекватно оцінювати ймовірність небажаного інциденту (реалізації загрози) та захищеність бази даних у цілому;

– метод оцінювання основних компонент бар'єрів безпеки та захищеності бази даних в цілому, який на відміну від відомих, за рахунок комплексування вдосконаленої моделі Клементса–Гофмана, введеного інтегрального показника безпеки, положень теорії нечітких множин та ризику, дозволяє адаптуватися до нових умов функціонування та прозоро, комплексно та кількісно оцінювати безпеку баз даних з різними моделями даних;

– метод маскування MOBAT, що відрізняється від відомого, можливістю обфускації даних, шляхом математичних перетворень на основі обчислення операцій за модулем, що застосовуються до елементів даних не тільки числового, а й широко поширеного в базах даних рядкового типу, що дозволяє суттєво розширити охоплення різноманітних маскованих з метою утруднення реалізації зловмисником загрози умовиводу даних БД.

3. Отримали подальший розвиток:

– метод маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних, що відрізняється від відомих оригінальним підходом до процесу перемішування з можливістю випадкової заміни елементів даних різного типу всередині заданого поля рядка та використання технології динамічного

маскування, що дозволяє при менших обчислювальних витратах на перетворення і без зміни формату вихідних даних забезпечити ефективне приховування даних, яке ускладнює реалізацію загрози логічного висновку;

– метод приховування коду збережених у базі даних програм, який на відміну від відомих, дозволяє за рахунок випадкової перестановки (що спирається на сучасний варіант алгоритму тасування Фішера-Йейтса) символів коду з можливою заміною кожного з них на інший випадково вибраний із стандарту Unicode забезпечити більше ефективний (якій вимагає значно більших обчислювальних витрат) захист коду від його розкриття зловмисником, при цьому гарантуючи цілісність коду.

Практичне значення отриманих результатів.

1. Розроблений метод маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних, орієнтований на заплутування, псевдонімізацію даних та ускладнення реалізації загрози логічного висновку, дозволяє зменшити час на відповідні операції перетворення на (10-17) % щодо методу класичного шифрування, при цьому не наводячи до зміни формату та збільшення розмірності даних, що зберігаються. Даний метод може бути також використаний у невиробничих базах даних, розширюючи можливості так званого статичного маскування даних.

2. Запропонований метод моніторингу модулів БД, що постійно зберігаються, вимагає менших обсягів збережених для цього даних і ресурсів процесора, ніж відомий метод контрольних сум, який для підтримки аналогічного контролю цілісності та справжності PSM вимагає виконання процедур гешування та цифрового підпису із збереженням відповідних даних для кожного конкретного PSM у конкретній схемі БД, причому однаково, не забезпечуючи контроль всього набору PSM загалом.

3. Розроблені в процесі створення схеми БД з УБВ спеціальні заходи у вигляді відповідних методів, реалізованих об'єктів схеми та правил їх використання підвищують безпеку таких баз даних, забезпечуючи високий ступінь контрольованості доступу до даних (аж до конкретного елемента),

необхідну конфіденційність, цілісність даних та об'єктів схеми БД, на відміну від традиційних РБД, що не володіють подібними заходами та функціональністю. Використання запропонованих рішень дозволяє підвищити ефективність / результативність захисту баз даних, побудованих на основі схеми з універсальним базисом відношень, більш ніж у 1.5 рази щодо традиційних реляційних БД.

Теоретичні та практичні результати дисертаційних досліджень реалізовані у приватному акціонерному товаристві «Інститут інформаційних технологій» та застосовуються у навчальному процесі Харківського національного університету імені В. Н. Каразіна.

Особистий внесок здобувача.

Дисертаційна робота є самостійно виконаною науковою працею, в якій викладено моделі і методи забезпечення безпеки баз даних з універсальним базисом відношень. Усі наукові результати, викладені в дисертаційній роботі, одержані автором особисто і відображені у наукових публікаціях. З наукових праць, виданих у співавторстві, у роботі використані лише ті положення, що становлять індивідуальний внесок автора. Конкретний внесок здобувача в цих роботах зазначений у списку наукових публікацій, опублікованих за темою дисертації.

Апробація матеріалів дисертації. Основні результати дисертаційної роботи були представлені, доповідались та обговорювались на 7 наукових конференціях: XII Міжнародній науково-практичній конференції «Теоретичні та прикладні аспекти побудови програмних систем (TAAPSD’2015)» (м. Київ, 2015 р.), Міжнародній науково-практичній конференції «Комп'ютерне моделювання в наукоємних технологіях (КМНТ-2016)» (м. Харків, 2016 р.), Восьмої міжнародної науково-технічної конференції «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління» (м. Харків, 2018 р.), X Міжнародній конференції IEEE «Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)» (м. Мец, Франція, 2019 р.), III Міжнародній науково-практичній конференції «Проблеми кібербезпеки інформаційно-телекомунікаційних систем (PCSITS)» (м. Київ,

2020 р.), Міжнародній науково-практичній конференції «Комп'ютерне моделювання в наукоємних технологіях (КМНТ-2021)» (м. Харків, 2021 р.), Міжнародній наукової конференції «Питання оптимізації обчислень (ПОО-XLVII)» (Львів, 2021 р.).

Публікації. Основні результати дисертаційних досліджень опубліковано у 19 наукових працях, серед яких: 7 статей у фахових виданнях України, 7 статей у зарубіжних виданнях (індексується у Scopus, Web of Science), 2 розділи у колективних монографіях, 3 матеріали та тези доповідей на конференціях (у тому числі 1 конференція, матеріали якої індексується у Scopus та Web of Science).

Структура та обсяг дисертації.

Дисертація складається зі вступу, 6 розділів, висновків, переліку посилань та додатків. Загальний обсяг дисертації складає 252 сторінки, з яких анотація на 10 сторінках, список публікацій здобувача за темою дисертації на 5 сторинках, зміст на 3 сторінках, перелік умовних позначень на 2 сторинках, основний текст на 168 сторінках, список використаних джерел із 247 найменувань на 25 сторінках, додатки на 39 сторінках. Робота містить 12 таблиць та 39 рисунків.

1. СУЧАСНИЙ СТАН, ПРОБЛЕМИ ТА ЗАВДАННЯ ЗАБЕЗПЕЧЕННЯ БЕЗПЕКИ БАЗ ДАНИХ

1.1. Сучасний стан та основні проблеми забезпечення безпеки баз даних

Як зазначалося у вступі, у всьому світі фахівці виявляють великий постійний науковий та практичний інтерес до проблеми інформаційної безпеки ІС та БД. Особливої актуальності питання захисту даних набули після прийняття Загального регламенту захисту персональних даних Європейського Союзу [17], закону України «Про захист персональних даних» [34], необхідності дотримання різних міжнародних законів та стандартів, у тому числі Стандарту безпеки даних індустрії платіжних карток [35, 36] та деяких інших. Відповідно до цих законодавчих актів власники баз даних зобов'язані забезпечити їх захист. При цьому значущість питань захисту даних істотно зростає, якщо розглядаються не просто одиночні джерела даних, а структури у вигляді безлічі інформаційних продуктів, у тому числі, об'єднаних деяким середовищем управління, подібно до того, що розглядається в роботах [15, 37]. А оскільки ландшафт даних стає більш складним, організаціям потрібні гнучкі, стійкі стратегії захисту даних, які можуть працювати і масштабуватись у багатоплатформеному, багатохмарному світі [20].

Серед тих, що мають безпосереднє відношення до актуальних сьогодні проблем забезпечення безпеки баз, сховищ даних (основні поняття, визначення даної галузі знань, які використовуються в подальшому в дисертаційній роботі наведені в Додатку Б), з можливими варіантами їхнього розв'язання, можна відзначити такі дослідження, що відбилися у різних авторитетних джерелах. Так, у відомій роботі [38] автори, розглядаючи основні концепції інформаційної безпеки, особливу увагу приділяють необхідності збалансованого захисту конфіденційності, цілісності та доступності даних, дотримання політик безпеки, формулюють основні принципи забезпечення цілісності даних та визначають механізми СКБД, які полегшують застосування цих принципів. Для забезпечення безпеки даних БД розглядається можливість використання кількох різних, але

взаємно підтримувальних підходів. А саме використання методу контролю доступу, аудиту і так званого методу толерантності, при якому допускається можливість деяких порушень безпеки (допускаючи певний рівень ризику щодо потенційних порушень безпеки, важливо розуміти, який ризик допустимий). Фокусуючись на реляційних системах, автори пропонують використання мандатних та дискреційних моделей управління доступом. Окрему занепокоєність викликає проблема логічного висновку, яка, на думку авторів, існуватиме навіть у ідеальній системі, повністю вільній від прихованих каналів.

Автори роботи [39], досліджуючи найбільш важливі концепції, що лежать в основі безпеки баз даних, і говорячи про доступність та можливості застосування для цього таких методів, як шифрування, цифрові підписи, відзначають, що дійсно всеосяжний підхід до захисту даних повинен також включати механізми забезпечення дотримання політик контролю доступу. При цьому контроль доступу повинен ґрунтуватися на вмісті даних, кваліфікацій, характеристик суб'єктів та деякої іншої контекстної інформації. Аналізуючи розвиток підходів до безпеки в історичному аспекті автор дослідження [40] зазначає, що багато проблем із захистом даних, що зберігаються в базі даних, найчастіше виникають не через відсутність досліджень, а через недостатню безпеку у відповідній реалізації бази даних або працюючих з ній застосунків. Незважаючи на значні покращення в моделях контролю доступу та механізмах безпеки, більшість баз даних, як і раніше, уразливі для різних загроз безпеці констатують автори роботи [41]. Причиною цього вони вважають те, що існуючі бази даних рідко розробляються з великим урахуванням вимог до безпеки, покладаючись на політики та механізми безпеки, які додаються згодом спеціальним чином. У таких випадках необхідний узгоджений підхід, який дозволяє організаціям спочатку оцінити поточне настроювання безпеки бази даних, тобто її політики та механізми, а потім перебудувати та вдосконалити механізми цілеспрямованим чином, тобто застосувати еволюційний, а не революційний підхід до підвищення безпеки баз даних. У монографії [42] авторами наводиться та досліджується список вимог до безпеки баз даних, розглядаються проблеми забезпечення

надійності та цілісності БД, особливо виділяється проблема виведення та агрегації, ідеального вирішення якої, на їхню думку, не існує. Однак при цьому вони пропонують деякі підходи до її вирішення, використовуючи методи приховування явно конфіденційної інформації; відстеження того, що знає користувач (це, природно, може призвести до максимально безпечного викриття, але це надзвичайно дорого, оскільки інформація повинна зберігатися про всіх користувачів, навіть якщо більшість із них не намагається отримати конфіденційні дані); маскування даних. У роботі [43] рекомендується зосередити зусилля на забезпеченні безпеки бази даних (під якою передбачається забезпечення конфіденційності, цілісності, доступності та неможливості відмови від авторства), враховуючи, насамперед, вплив втрати даних для бізнесу. Автори надають перевагу кількісному підходу до оцінки ризику перед якісним. Для забезпечення безпеки БД у роботі [44] пропонується використовувати такі методи як контроль доступу, боротьба з ін'єкціями SQL, шифрування, маскування даних. Не можна не відзначити результати досліджень, викладені в монографії [7], в якій наголошується, що питання безпеки належать не тільки до даних, що зберігаються в базі даних. Порушення безпеки можуть вплинути на інші частини системи баз даних, що, у свою чергу, наражає на небезпеку і власне базу даних. Отже, безпека БД може бути забезпечена лише у випадку, якщо буде забезпечена безпека обладнання, програмного забезпечення, персоналу та даних. Щодо загроз, які можуть негативно вплинути на роботу розрахованих на багато користувачів систем баз даних рекомендується використовувати такі заходи безпеки як: авторизацію; контроль доступу (дискреційний, мандатний методи контролю доступу, багаторівневі відносини та поліінстанцію), застосування уявлень; резервне копіювання та відновлення; підтримку цілісності; шифрування; використання відмовостійкої апаратури.

Сьогодні ми живемо в суспільстві, що базується на даних. Дедалі частіше на прийняття рішень впливають результати аналізу відповідних даних. [2]. Дані знаходяться в центрі всього. Тобто область баз даних збільшила широту охоплення, а, отже, з'явилися й нові проблеми. Наприклад, потреба в управлінні

даними (причому дані можуть зберігатися і переміщуватися не тільки всередині різних ІС, але також між організаційними одиницями та національними кордонами) призвела до появи конфіденційних хмарних обчислень, що передбачають зберігання даних у зашифрованому вигляді при використанні хмарних ресурсів. Занепокоєння викликають питання етики та правомірності використання даних (відповідно до встановлених політик). На новий рівень виходить проблема агрегації та логічного висновку. Як відомо, серед загроз, які складно контролювати в рамках СКБД, особливе місце займає загроза, пов'язана з можливістю зловмисниками робити умовиводи на основі різних алгебраїчних обчислень, порівнянь, фільтрації даних, до яких він допущений, без необхідності прямого доступу до самого захищеного об'єкта [42]. Проблема агрегації і об'єднання виникає щоразу, коли набір даних, що агрегується чи об'єднується, утворює більш важливі відомості порівняно з важливістю тих окремо взятих даних, на основі яких і виходять ці відомості. Використання складних, а також послідовності простих логічно пов'язаних запитів дозволяє зловмиснику отримувати дані, безпосередній доступ до яких йому навпростець закрито. У міру того, як ми продовжуємо агрегувати дані, ключовою проблемою стає балансування дотримання конфіденційності / приватності даних (англ. data privacy) з їх аналітичним використанням для підтримки прийняття рішень [2]. В епоху великих даних проблема захищеності чутливих даних ще більше загострюється, тому що технічні засоби захисту недоторканності приватного життя здають свої позиції [45]. І нині не існує ідеальних рішень проблем, пов'язаних із логічним висновком, агрегацією та об'єднанням [42]. Як правило, протидія подібним загрозам здійснюється такими методами, як [38, 42, 44, 46-48]: блокування відповіді за неправильної кількості запитів; контроль запитів, що надходять (або аудит); спотворення відповіді шляхом навмисного коригування даних (обмежене придушення відповіді (часткова видача запитаних даних), об'єднані результати, випадкове збурення даних, свопінг / обмін даних, приховування, випадковий вибір запису для обробки, контекстно-орієнтований захист, поліінстанція (багатоекземплярність – наявність відносин з кількома

екземплярами з різним рівнем (класом) доступу), диференціальна приватність та деякими іншими.

При цьому важливо розуміти, що, обмежуючи зловмиснику можливість робити логічні висновки щодо отриманої інформації, використовуючи такі методи, ми автоматично обмежуємо запити користувачам, які не передбачають здійснення несанкціонованого доступу до даних. Більше того, спроби перевірити запитані звернення щодо можливості робити неприпустимі, з точки зору власника даних, умовиводи можуть знизити продуктивність СКБД [42]. Більшість досліджень, пов'язана з вивченням можливості робити умовиводи на основі отриманих різними способами даних з баз, сховищ даних була проведена на початку 1980-х років. Однак і на сьогоднішній день аспекти, пов'язані з компрометацією конфіденційності даних, що видобуваються шляхом умовиводу, значною мірою залишаються невирішеними [42, 48, 49]. Наприклад, відомі різні методи маскування даних БД, сховищ даних, що знайшли досить широке застосування у певних класах завдань, а саме [42, 44, 50-59]: заміна, перестановка, випадкове відхилення даних (існуюче значення замінюється випадковим у певному діапазоні), шифрування, у тому числі шифрування із збереженням формату (FPE), видалення (просте видалення даних стовпця шляхом заміни його значеннями NULL), заміна на символ константу (даний метод є окремим випадком методу підстановки, коли всі символи, що маскуються, замінюються одним і тим же символом, наприклад, «X», метод маскування числових даних шляхом певних математичних перетворень з використанням операцій з модулю (MOBAT), складне маскування, токенізація, а також деякі інші. Однак більшість цих методів, за винятком методів шифрування, у тому числі FPE [57, 58], токенезації [60, 61] і MOBAT [52, 53, 55], використовується для статичного маскування невиробничих БД і після їх застосування не дозволяє скасувати операції, щоб повернутися до вихідних даних, що не є прийнятним, особливо для виробничих БД. При цьому метод шифрування, у тому числі із збереженням формату [57, 58] є досить ресурсомістким [44, 51]. Техніка MOBAT спеціально розроблена для маскування лише числових значень [52, 53], а досить часто

виникає потреба маскування не лише числових значень. При цьому спроби застосування процедури маскування до не числових даних у модифікованому методі MOBAT, викладені в роботі [55], не вирішують цю проблему повною мірою. Досі неясно, яким чином диференціальну приватність можна ефективно впровадити до платформ управління базами даних без суттєвого обмеження областей запитів [2].

Забезпечення інформаційної безпеки баз даних неможливе без розгляду аспектів забезпечення цілісності даних. Багато, особливо комерційних організацій більше стурбовані цілісністю своїх даних, ніж їх конфіденційністю [60]. Цілісність для них є більш важливою. Якщо ви публікуєте інформацію в Інтернеті на Web-сервері і вашою метою є зробити її доступною для найширшого кола людей, то конфіденційність у цьому випадку не потрібна. Зате істотно підвищується відповідальність за надання неспотвореної інформації, що отримується з БД, наприклад, про відомості, що зберігаються в ній, з офіційних правових, нормативних, фінансових, медичних та інших документів організації, у тому числі і самих цих документів. Інформація має бути справжньою чи непідробною. Дані повинні бути правильними, правдивими, бути справжнім відображенням реальності. У цілому ж, і в комерційному, і у військовому середовищі важко уявити систему, для якої були б не важливі властивості цілісності [62]. Залежно від важливості аналізованого аспекту цілісності та області використання даних, можуть використовуватись різні методи та засоби, що гарантують цілісність даних при різних можливих загрозах. Так правильність, неспотвореність і незмінність даних може забезпечуватися з допомогою методів і засобів спеціальних технологій розмежування доступу, що ґрунтуються на формальних моделях цілісності [63]. Неспотвореність даних при зберіганні та передачі в інформаційних системах може забезпечуватись за рахунок криптографічних примітивів, таких як: цифровий підпис, криптографічні геш-функції, коди автентичності. Також важливу роль у механізмі забезпечення цілісності БД грає концепція правильно сформованої транзакції [38]. Крім того, беручи до уваги двоїсту природу систем баз даних, як інформаційного продукту з

двома компонентами (активами): власне збереженими в БД даними, доступними для використання, та програмними засобами СКБД, а також можливості шкідливого впливу на ці активи, доцільним є забезпечення безпеки їх обох. Тобто потрібно забезпечити захист не тільки даних (фактів), що зберігаються в БД, але й інших важливих об'єктів бази даних, які здійснюють управління даними. Постійний моніторинг цих об'єктів бази даних є дуже важливим, оскільки деякі з атак на БД і не тільки на неї (наприклад, можна атакувати операційну систему через вразливість сервера БД) можуть бути виявлені саме на основі аналізу зміни (навмисної або випадкової) цих об'єктів (порушення їх цілісності, автентичності), або їх набору (збільшення чи зменшення їхньої кількості) на сервері БД. Мова йде про так звані модулі, що постійно зберігаються (PSM), – спеціальним чином оформлені програми, що включають оператори SQL, які зберігаються в базі даних, можуть викликатися застосунками та виконуються всередині СКБД. У PSM визначаються модулі, які є колекціями визначень функцій та процедур, оголошень тимчасових таблиць та деяких інших необов'язкових оголошень [32]. Подібні фрагменти коду зазвичай створюються за допомогою мови SQL і конкретної реалізації у вибраній СКБД. З іншого боку, із низки причин: дотримання прав інтелектуальної власності; комерційна цінність; код забезпечує вирішення завдань захисту та розподілу прав доступу до даних; неприпустимість модифікації коду іншими користувачами або процесами (шкідливими програмами), код цих програм доцільно приховати. Для приховування коду розробники деяких СКБД пропонують різноманітні засоби. Так в СКБД Microsoft SQL Server можна скористатися механізмами шифрування коду процедур, що зберігаються, для чого в конструкціях `CREATE PROCEDURE` і `ALTER PROCEDURE` слід використовувати опцію `WITH ENCRYPTION` [64], в СКБД Oracle для приведення коду PL/SQL до нечитаного вигляду можуть використовуватися утиліта wrap, вбудовані пакети `DBMS_DDL` та `DBMS_WRAP` [65, 66]. Однак, як відзначають фахівці в галузі безпеки баз даних і показує практика, використовувані вбудовані засоби приховування збережених програм, які поставляються разом із СКБД, є недостатньо ефективними і їх порівняно легко можуть оминути зловмисники,

особливо ті, що мають права привілейованого користувача [67-69]. Наприклад, збережені процедури, зашифровані за допомогою опції `WITH ENCRYPTION` в СКБД Microsoft SQL Server, можуть бути досить просто розшифровані за допомогою утиліти «dSQLSRVD», збереженої процедури «Decryptsp2K» або нових безкоштовних автономних інструментів, таких як ApexSQL Decrypt і dbForge SQL Dec. На сьогоднішній день відомі алгоритм і програми, що виконують зворотне перетворення нечитаного звичайними засобами, так званого «wrap» коду збережених процедур, функцій, пакетів у СКБД Oracle, наприклад, такі, як on-line програма Unwrap It!, PL/SQL Unwrapper для SQL Developer. Таким чином, можна зробити висновок, що наявні можливості вбудованих в деякі СКБД засобів, не повною мірою можуть забезпечити ефективне приховування коду збережених програм, не кажучи вже про іншу категорію СКБД, де такі взагалі відсутні.

Одним з основних завдань у галузі проектування та управління ІБ є отримання достатніх та достовірних доказів безпеки досліджуваної системи [70]. Розробникам, керівникам проектів та виконавчому керівництву потрібна інформація про стан безпеки системи на різних етапах її життєвого циклу. А для того, щоб можна було б перевірити висновки щодо ступеня забезпечення безпеки, її необхідно якось виміряти. Вимірювання безпеки – це складна проблема, яку не можна недооцінювати. Показники інформаційної безпеки, як наголошується в документі NIST [71], – це важливий фактор при прийнятті обґрунтованих рішень з різних аспектів безпеки, починаючи від проектування архітектур та засобів управління безпекою до ефективності / результативності (англ. effectiveness) та ефективності / продуктивності (англ. efficiency) операцій із забезпечення безпеки. При цьому під ефективністю / результативністю розуміється властивість об'єкта оцінки, що представляє, наскільки добре він забезпечує безпеку в контексті його фактичного або передбачуваного використання [72, 73]. Ефективність / результативність безпеки означає впевненість у тому, що механізми забезпечення безпеки системи відповідають заявленим цілям безпеки (тобто вони не роблять нічого, крім того, що вони повинні робити, задовольняючи при цьому очікування щодо відмовостійкості) [70, 71, 74]. Під ефективністю / продуктивністю безпеки

розуміється впевненість (гарантія) у тому, що в досліджуваній системі було досягнуто належної якості безпеки та дотримано обмежень щодо ресурсів, часу та вартості [70, 74].

Основою для проведення будь-яких робіт у галузі інформаційної безпеки, включаючи оцінювання ефективності захисту, є міжнародні стандарти, зокрема такі, як ISO/IEC 15408 [75], ISO/IEC 27001 [76], ISO/IEC 27004 [77]. Так міжнародний стандарт ISO/IEC 15408 визначає загальний набір вимог до функціональних можливостей безпеки продуктів інформаційних технологій, які можуть бути реалізовані у вигляді апаратного, програмно-апаратного або програмного забезпечення, та до заходів довіри, що застосовуються до цих продуктів ІТ при оцінці безпеки, а також загальну методологію оцінки з урахуванням загроз, вразливостей, активів, ризиків заподіяння збитків та вибору контрзаходів. ISO/IEC 15408 застосовується до ризиків, що виникають внаслідок людської діяльності (зловмисної або іншої), та до ризиків, що виникають не внаслідок дій людини. Він є досить гнучким, що дозволяє застосовувати низку методів оцінки до низки властивостей безпеки ІТ-продуктів. Тому користувачам стандарту рекомендується виявляти обережність, щоб запобігти зловживанню цією гнучкістю. Наприклад, використання положень стандарту у поєднанні з невідповідними методами оцінки, невідповідними властивостями безпеки або невідповідними продуктами ІТ може призвести до безглуздих результатів оцінки.

На цей час було зроблено багато серйозних спроб виміряти чи оцінити безпеку, зокрема, використовуючи критерії оцінки довірених комп'ютерних систем [78], критерії оцінки безпеки інформаційних технологій [72], модель зрілості можливостей проектування систем безпеки [73], загальні критерії [79]. Проте кожна спроба мала лише обмежений успіх [71]. Пропоновані сьогодні різні підходи до оцінки ефективності захисту активів [80-86], які умовно можна класифікувати як вартісні, функціональні та засновані на аналізі ризиків [80], з відповідними методами та показниками, також повною мірою не вирішують проблему. Так як у загальному випадку постановка завдання забезпечення ІБ може змінюватись у широких межах, а ефективність функціонування системи ЗІ

залежить від безлічі факторів та оцінюється сукупністю показників, що перебувають у складних взаємозв'язках, тому закономірним є різноманіття таких різних не пов'язаних методів оцінки ефективності захисту активів. У зв'язку з чим, через відсутність єдиного загальноприйнятого підходу до розв'язання завдань даного класу – оцінювання ефективності системи захисту, визначення певної суворої методології, що дозволяє оцінювати безпеку баз даних, необхідні подальші дослідження.

Викладене вище свідчить, що незважаючи на досягнуті результати та прогрес у вирішенні безлічі проблем безпеки БД, через об'єктивні обставини, пов'язані з розвитком технологій БД, збільшенням широти охоплення, зростаючою кваліфікацією зловмисників, що постійно вдосконалюють можливості шкідливого впливу, введенням нових положень та рекомендацій різних нормативно-правових актів, є насущна потреба у перегляді підходу до вирішення проблеми управління даними та забезпечення їхньої безпеки шляхом знаходження нових рішень у вигляді певних моделей, методів, прийомів, засобів, актуальних як у теоретичному, так і у прикладному аспектах. Цей напрямок залишається плідною сферою досліджень.

1.2. Аналіз формальних моделей безпеки та їх застосування для баз даних

Аналіз теоретичних і прикладних аспектів забезпечення ІБ дозволив виявити і сформулювати ряд загальних принципів, якими слід керуватися при проектуванні і експлуатації захищених ІС [38, 87, 88]: розумної достатності; мінімуму привілеїв; поділу обов'язків / привілеїв; цілеспрямованості; системності; комплексності; безперервності; керованості; поєднання уніфікації та оригінальності.

Забезпечити безпеку легше, якщо є чітка модель того, що потрібно захищати і кому і що дозволено робити. [89]. Тому невід'ємною частиною будь-якого проекту зі створення чи оцінки безпеки баз даних, як зазначається у роботі [90], є

наявність моделі безпеки, під якою розуміється формальний вираз політики безпеки [91, 92]. Основна мета створення політики безпеки ІС та опису її у вигляді формальної моделі – це визначення умов, яким має підпорядковуватися поведінка системи, визначення показників та критерію безпеки, а також проведення формального доказу відповідності системи, що захищається, цьому критерію при дотриманні встановлених правил та обмежень [91]. Формальна модель політики безпеки (що представлена у вигляді математичних виразів, схем, діаграм, алгоритмів і т. д.) грає важливу роль у процесах розробки та дослідження інформаційних систем у цілому, та БД зокрема, оскільки забезпечує системний підхід. Моделі безпеки дозволяють вирішити ряд завдань, що виникають у ході розробки та дослідження захищених систем, таких як [87]: вибір та обґрунтування базових принципів архітектури захищених ІС, що визначають механізми реалізації засобів та методів ЗІ; підтвердження властивостей захищеності створюваних систем шляхом формального доказу дотримання політики безпеки; складання формальної специфікації політики безпеки як найважливішої складової частини організаційного та документаційного забезпечення створюваних захищених ІС. Розумне використання відповідних моделей – це один із важливих кроків на шляху забезпечення безпеки ІС і бази даних, а розробка відповідних моделей – це перший крок на шляху до створення теоретичних основ забезпечення безпеки в ІС [88].

Найбільш поширеною парадигмою моделей забезпечення безпеки даних є суб'єктно-об'єктна абстракція. Формально суб'єктно-об'єктна модель ІС це трійка:

$$\langle S, O, Op \rangle, \tag{1.1}$$

де S і O це на два класи сутностей: активні сутності – суб'єкти (користувачі, процеси) $S = \{s_1, s_2, ..., s_m\}$ та пасивні сутності – об'єкти $O = \{o_1, o_2, ..., o_n\}$ (в якості таких об'єктів для РБД можуть розглядатися таблиці, уявлення, домени (типи), атрибути, кортежі, процедури, функції, тригери та деякі інші). Активність сприймається як можливість виконувати операції над об'єктами (пасивними сутностями). $Op = \{op_1, op_2, ..., op_L\}$ – множина операцій над об'єктами.

Серед формальних моделей безпеки доречно проаналізувати такі їх основні класи, що набули широкого поширення і мають пряме відношення до аспектів, що розглядаються в роботі:

a) моделі управління доступом: моделі безпеки на основі дискреційної політики; моделі безпеки на основі мандатної політики; моделі безпеки на основі рольової політики;

b) моделі забезпечення цілісності даних.

Розгляд цих моделей безпеки є доцільним з кількох причин. По-перше, вони можуть бути безпосередньо використані для аналізу безпеки як існуючих, так і перспективних ІС та їх основного функціонального компонента – БД, особливо у випадках, коли потрібне отримання гарантій захищеності ІС. Класичні моделі безпеки ІС дозволяють формально аналізувати властивості різних механізмів захисту ІС По-друге, існуючі моделі безпеки можуть бути використані як основа для розробки більш досконалих моделей, що дозволяють більш точно описувати та досліджувати особливості функціонування механізмів захисту сучасних ІС. По-третє, володіння знаннями про моделі безпеки ІС надає фахівцю в галузі комп'ютерної безпеки можливості для суворого наукового та теоретично обґрунтованого викладу результатів прикладних досліджень [93].

1.2.1. Моделі управління доступом

У цьому підрозділі розглянемо основні положення найпоширеніших моделей безпеки, що ґрунтуються на контролі доступу суб'єктів до об'єктів, та загальному критерії безпеки ІС, який формулюється наступним чином.

Визначення 1. Інформаційна система безпечна тоді і лише тоді, коли суб'єкти не мають жодних можливостей порушувати (обминати) встановлену в ній політику безпеки.

1.2.1.1. Моделі безпеки на основі дискреційної політики

Роботи з моделей дискреційного доступу до інформації в ІС з'явилися ще в (60-70)-х роках минулого століття. Вони досить широко висвітлені у науковій

літературі. Найбільш відомі з них – це модель ADEPT-50 [94], п'ятивимірний простір Хартсона [95], модель Харрісона-Руццо-Уллмана [96], модель Take-Grant [97]. Авторами цих моделей був зроблений значний внесок у теорію безпеки комп'ютерних систем. Їхні роботи заклали основу для подальшого створення та розвитку захищених ІС.

У теоретичному та практичному плані найбільшого розвитку та застосування отримали дискреційні моделі, засновані на *матриці доступу* – таблиці (M), яка описує права доступу суб'єктів (S) до об'єктів (O), рядки якої відповідають суб'єктам доступу $s_1, s_2, ..., s_m$, стовпці об'єктам доступу $o_1, o_2, ..., o_n$, а в комірках (елементах матриці $M[s_i, o_j]$) записуються дозволені операції (види доступу) $op_1, op_2, ..., op_L$ відповідного суб'єкта над відповідним об'єктом. У наведеній на рисунку 1.1 матриці доступу M види доступу op_l ($l = 1...L$) допускають такі операції (види доступу): читання (rd), запис з модифікацією (w), запис без модифікації (тільки з новим записом або дописуванням в файл) (a), запуск об'єкта на виконання (e). Однак, як зазначається в монографії [88], при необхідності елементи матриці можуть містити покажчики на процедури. Ці процедури виконуються при кожній спробі доступу до заданого об'єкта. Тим самим рішення про доступ може прийматися на підставі складніших залежностей не настільки очевидних, як у простій матриці доступу.

	o_1	o_2	…	o_j		o_n
s_1	rd	rd, w		e		rd
s_2	rd, a	-		rd		e
…						
s_i	rd	-		-		rd
…						
s_m	rd, w	-		e		e

Рисунок 1.1. Матриця доступу M

Дана модель передбачає, що всі спроби доступу до об'єктів перехоплюються і перевіряються спеціальним керуючим процесом (так званим монітором безпеки).

Таким чином, суб'єкт s_i отримає ініційований ним доступ op_l до об'єкта o_j тільки в разі, якщо елемент матриці $M[s_i, o_j]$ має значення op_l.

За принципом управління доступом виділяються два підходи [87, 90]: примусове керування доступом; добровільне керування доступом. На практиці ж у більшості випадків, у тому числі і для БД, застосовується комбінований спосіб управління доступом, коли певна частина повноважень доступу до об'єктів встановлюється привілейованим користувачем (принцип примусового управління), а інша частина – власниками об'єктів (принцип добровільного управління) [90]. У багатьох ІС права володіння об'єктами можуть передаватися. В результаті при добровільному управлінні доступом реалізується повністю децентралізований принцип управління процесом розмежування доступу. Такий підхід забезпечує гнучкість формування правил розмежування доступу для конкретної сукупності користувачів до активів, але призводить до ускладнення загального контролю стану безпеки активів в системі. Що в свою чергу вимагає додаткового дослідження умов і процесів поширення прав доступу [87].

У теоретичному плані вперше дана проблема була досліджена Харрісоном (Harrison), Руццо (Ruzzo) і Уллманом (Ullman), які для цього розробили спеціальну формальну модель дискреційного доступу (скорочено – модель HRU) [96]. У даній моделі додатково до суб'єктів S, об'єктів O (для того, щоб включити в область дії моделі і відносини між суб'єктами, прийнято вважати, що всі суб'єкти одночасно є і об'єктами: $S \subseteq O$) і кінцевого набору прав доступу $R = (r_1, r_2, ..., r_K)$ ($M[s,o] \subseteq R$), вводиться також простір станів системи $Q = (S, O, M)$. Простір станів системи утворюється декартовим добутком множин складових її об'єктів, суб'єктів і прав: $S \times O \times M$. Будь-який елемент матриці M містить набір прав суб'єкта s_i до об'єкта o_j, що належать множині прав доступу R. Поведінка системи в часі розглядається як послідовність станів $\{Q_v\}$, кожний наступний стан є результатом застосування деякої команди ($\alpha_\tau \in C$, де C – кінцевий набір

команд) до попереднього: $Q_{v+1} = \alpha_\tau(Q_v)$. *Критерій безпеки* в моделі HRU формулюється в такий спосіб.

Визначення 2. Система є безпечною щодо права r_k ($k = 1...K$; для простоти часто надалі замість r_k використовуватимемо узагальнене позначення $r \in R$), якщо для заданого початкового стану $Q_0 = (S_0, O_0, M_0)$ не існує застосовної до Q_0 послідовності команд, в результаті якої право r буде занесено до елементу матриці M, в якій воно було відсутнє в початковому стані Q_0. Іншими словами це означає, що суб'єкт ніколи не отримає право доступу r до об'єкта, якщо він не мав його спочатку. Якщо ж право r виявилося в комірці матриці M, в якій воно спочатку було відсутнє, то кажуть, що стався витік права r [98].

З критерію безпеки випливає, що для цієї моделі ключову роль грає вибір значень прав доступу та їх використання в умовах певних команд. По суті дана модель описує не лише доступ суб'єктів до об'єктів, а поширення прав доступу від суб'єкта до суб'єкта, оскільки саме зміна змісту елементів матриці доступу визначає можливість виконання команд, у тому числі команд, що модифікують саму матрицю доступу, які потенційно можуть призвести до порушення критерію безпеки. Однак, Харрісон, Руццо та Уллман довели, що в загальному випадку не існує алгоритму, який може для довільної системи, її початкового стану $Q_0 = (S_0, O_0, M_0)$ та загального права r вирішити, чи ця конфігурація є безпечною. Тим не менш, саме модель HRU послужила основою для цілого класу моделей політик безпеки таких, наприклад, як модель TAM [99], модель TAKE-GRANT [97] та деяких інших, які використовуються для управління доступом та контролю за поширенням прав у різних системах [91]. Розвиток моделей дискреційного управління доступом полягає переважно у побудові різноманітних модифікацій моделі HRU, а також у пошуку мінімально можливих обмежень, які можна накласти на опис системи, щоб питання її безпеки було обчислювально вирішуваним [98]. Так модель TAKE-GRANT спрямована на аналіз шляхів поширення прав доступу в системах дискреційного керування доступом. Основна її мета – визначення та обґрунтування умов перевірки, що алгоритмічно

перевіряються, можливості витоку права доступу за вихідним графом доступів, що відповідає деякому стану системи [96]. Використовуючи правила перетворення: «*take*», «*grant*», «*create*» и «*delete*», можна відтворити стани, в яких буде перебувати система залежно від розподілу та зміни прав доступу. Модель TAKE-GRANT, як і її розширена модель [100], відіграє важливу методологічну роль, надаючи теоретико-графовий інструмент аналізу систем розмежування доступу з погляду санкціонованого та несанкціонованого з боку певних суб'єктів поширення прав доступу в рамках дискреційної політики. Основні рішення розглянутих вище моделей знайшли сьогодні застосування в різних СКБД (переважно реляційних) при наданні користувачам прав на виконання тих чи інших операцій з тими чи іншими об'єктами. Дискреційна модель розмежування доступу до СКБД ґрунтується на таких основних положеннях.

1. Усі суб'єкти (S) та об'єкти (O) БД повинні бути ідентифіковані, тобто кожній сутності (активній, пасивній) має бути присвоєний унікальний ідентифікатор.

2. Для кожного об'єкта (o_j) БД має бути визначено користувач-власник ($s^{owner_j} = s_i \in S$).

3. Для суб'єктів (S) мають бути визначені права доступу (привілеї чи повноваження) до різних об'єктів (O). У стандарті SQL визначено такі типи привілеїв доступу [32, 101]: `SELECT`, `INSERT`, `DELETE`, `UPDATE`, `REFERENCES`, `USAGE`, `TRIGGER`, `EXECUTE`, `UNDER`. Інформація про привілеї (аналог матриці доступів) зберігається в системному каталозі СКБД у вигляді повноважень, виражених за допомогою певної мови опису. Важлива роль забезпечення безпеки даних у традиційних СКБД відводиться уявленням, які дозволяють контролювати, які дані доступні тому чи іншому суб'єкту [102].

4. Власник об'єкта (s^{owner_j}) повинен мати право визначення прав доступу до об'єкта іншим суб'єктам ($s_i \in S$). Суб'єкт отримує / втрачає певні права (привілеї) двома способами: перший пов'язаний із створенням об'єкта (o_j) та його володінням, другий – шляхом передачі/відкликання певних прав одним

суб'єктом – іншому [32]. Для цих цілей у стандарті SQL визначені оператори `GRANT / REVOKE`, які дають змогу одному користувачеві призначати (надавати) певні права іншому або відкликати їх у іншого.

5. У системі існує привілейований користувач ($s^{pv} \in S$), що має право повного доступу до будь-якого об'єкта. При цьому слід зазначити, що реалізація цього положення не повинна дозволяти такому користувачеві використати свої повноваження непомітно для реального власника об'єкта (що найчастіше сьогодні не завжди виконується).

Підсумовуючи вищесказане, можна зробити певні висновки про переваги та недоліки дискреційної політики керування доступом та моделях безпеки, побудованих на її основі. До переваг дискреційної політики доступу можна віднести відносну простоту та гнучкість реалізації системи керування доступом. Що підтверджується діючими ІС, безпека яких забезпечується виконанням вимог цієї політики. Наприклад, класична модель HRU досі широко використовується при проведенні формальної верифікації коректності побудови систем розмежування доступу у автоматизованих системах, що добре захищені [98, 103]. До недоліків дискреційних моделей відносяться: статичність встановлених правил керування доступом (зазвичай, вони не враховують динаміку змін станів ІС); неможливість контролювати повною мірою потоки інформації між об'єктами; складність відстеження наданих суб'єктам привілеїв за їх великої кількості; відсутність механізму керування доступом до конфіденційної інформації. Крім цього, при використанні таких моделей досить складно (алгоритмічно важко вирішувана задача) перевірити чи приведуть дії суб'єкта, який керується деяким набором правил, до порушення безпеки чи ні. Усе це зумовило подальший пошук та розробку інших моделей управління доступом.

1.2.1.2. Моделі безпеки на основі мандатної політики

Найбільшого поширення серед моделей мандатного управління доступу (багаторівневого захисту) набула модель Белла-ЛаПадули [104], основними

елементами якої є: S – множина суб'єктів; O – множина об'єктів; $R=\{read, write, append, execute\}=\{r_d, w, a, e\}$ – множина видів доступу (a=*append* – доступ на запис в кінець об'єкта); $B=\{b\subseteq S\times O\times R\}$ – множина можливих множин поточних доступів у системі; Λ_L – решітка рівнів безпеки L (наприклад, $L=\{U, SU, C, S, TS\}$, де U – нетаємно, SU – чутливо, але нетаємно, C – конфіденційно, S – таємно, TS – цілком таємно, $U<SU<C<S<TS$); $M=\{M_1, M_2, ..., M_c\}$ – множина можливих матриць доступів, де $c=n\cdot m\cdot 2^p$, $p=|R|$, $M_\eta[s,o]\subseteq R$ – права доступу суб'єкта s до об'єкта o, $\eta=1..c$; $(f_s, f_o, f_c)\in F=L^S\times L^O\times L^S$ – трійка функцій (f_s, f_o, f_c), що задають відповідно: $f_s: S\to L$ – рівень безпеки (доступу) суб'єктів; $f_o: O\to L$ – рівень безпеки об'єктів; $f_c: S\to L$ – поточний рівень безпеки (доступу) суб'єктів, при цьому для будь-якого $s\in S$ виконується нерівність $f_c(s)\le f_s(s)$; $V=B\times M\times F$ – множина станів системи; Q – множина запитів системі; D – множина відповідей на запити, наприклад, $D=\{yes, no, error\}$; $W\subseteq Q\times D\times V\times V$ – множина дій системи, де четвірка $(q, d, v^*, v)\in W$ означає, що система за запитом q з відповіддю d перейшла зі стану v до стану v^* і деякі інші.

Основні властивості (правила, що забезпечують розмежування доступу) безпеки моделі, що розглядається:

1. *Властивість простої безпеки* (*ss*). Суб'єкт на рівні безпеки $l\in L$ може проводити операцію читання тільки щодо об'єктів свого або нижчого рівня. Ця властивість також відома як правило – немає читання нагору (NRU).

2. *Властивість* * (**-property*; правило – немає запису вниз (NWD)). Суб'єкт із заданим рівнем безпеки $l\in L$ може здійснювати запис тільки в об'єкти свого або вищого рівня.

3. *Властивість дискреційної безпеки* (*ds*). Права дискреційного доступу суб'єкта до об'єкта визначаються з урахуванням матриці доступу M. Термін «дискреційна» безпека є доречним у контексті конкретних рішень цієї моделі, оскільки в модель включена можливість змінювати M (структуру дозволів).

Для СКБД особливо багато уваги методам мандатного управління доступом стало приділятися на початку 90-х. Це було пов'язано, як зазначає К. Дейт [27], з тим фактом, що згідно з вимогами Міністерства оборони США будь-яка СКБД, що використовується в цьому відомстві, повинна була підтримувати принципи мандатного управління доступом. Тому розробникам СКБД довелося розпочати суперництво за якнайшвидшу розробку методів такого управління. Мандатна модель управління доступу до СКБД ґрунтується на таких основних положеннях:

1. Усі суб'єкти (S) та об'єкти (O) БД мають бути ідентифіковані.
2. Повинні бути визначені решітка рівнів безпеки L .
3. Кожному об'єкту БД $o \in O$ повинен бути присвоєний рівень безпеки (мітка конфіденційності), що визначає обмеження на доступ до даного об'єкта.
4. Кожному суб'єкту БД $s \in S$ має бути присвоєний рівень безпеки – рівень доступу, що задає рівень повноважень даного суб'єкта.
5. Суб'єкт $s \in S$ може отримати доступ до об'єкта БД $o \in O$ тільки у випадку, коли рівень доступу суб'єкта дозволяє надати йому даний доступ до об'єкта із заданим рівнем конфіденційності, і реалізація доступу не призведе до виникнення інформаційних потоків від об'єктів з високим рівнем конфіденційності до об'єктів з низьким рівнем конфіденційності.

Важливою відмінністю мандатного управління доступу від дискреційного є те, що в мандатній моделі контролюються не операції, що виконуються суб'єктом над об'єктом, а потоки інформації, які можуть бути лише двох видів: або від суб'єкта до об'єкта (запис), або від об'єкта до суб'єкта (читання).

У реляційній моделі в якості структури, що володіє міткою, природно вибрати кортеж (рядок). Таке рішення дозволяє забезпечити достатню вибірковість доступу. Місцем зберігання самих міток може бути вибраний відповідний атрибут кортежу, визначений на домені рівнів безпеки L . При цьому має бути вирішене питання з механізмом формування та зміни цього атрибута. Один із підходів до побудови системи управління доступом у реляційних СКБД на основі мандатної політики описаний у групі патентів [105-107]. Запропонований підхід забезпечує контроль доступу на рівні рядків (RLS) у

таблиці РБД. У різних публікаціях такий механізм згадується під різними назвами: детальний контроль доступу (FGAC), віртуальна приватна база даних (VPD) та деякі інші [108, 109]. Суть детального контролю доступу полягає в наступному. Таблиця бази даних містить стовпець мітки безпеки, у рядки якого записуються конкретні значення міток, визначених в ієрархії рівнів безпеки L. Коли суб'єкт $s \in S$ запитує доступ до рядка (що є об'єктом доступу $o \in O$, у випадку, що розглядається) таблиці, механізм безпеки порівнює рівень безпеки суб'єкта з рівнем безпеки (міткою конфіденційності), зазначеному у відповідному атрибуті рядка. Якщо рівень безпеки суб'єкта s переважає рівень безпеки, зазначений у відповідному атрибуті рядка, суб'єкту надається доступ до рядка. Мітка визначає рівень безпеки для суб'єкта у багаторівневій схемі безпеки та певні привілеї для доступу до даних БД. Крім того, мітка безпеки може визначати категорії безпеки в межах рівня безпеки, до якого суб'єкта дозволено доступ. Значення, що зберігається в мітці, формується деяким способом, який дозволяє зрозуміло висловити інформацію про рівень та категорію безпеки для системи безпеки. Доступ до відповідного рядка дозволено лише в тому випадку, якщо безпека суб'єкта переважає над безпекою рядка, в якому виконуються обидва наступні умови:

– рівень безпеки, вказаний міткою безпеки суб'єкта, більший або дорівнює рівню безпеки, зазначеному міткою безпеки рядка;

– категорії безпеки, пов'язані з міткою безпеки рядка, є належною підмножиною категорій безпеки, пов'язаних з міткою безпеки суб'єкта.

Реалізації технології мандатного доступу до різних СКБД можуть відрізнятися. Наприклад, у СКБД Oracle вона накладається на реалізацію дискреційної моделі. А саме можливість доступу до даних проходить перевірку, що містить два етапи. Спочатку перевіряються стандартні обмеження дискреційного доступу. Потім для суб'єктів, що пройшли перевірку першого етапу, перевіряється можливість доступу до об'єктів. Ця можливість базується на обмеженнях мандатної моделі. Механізм VPD Oracle дозволяє регламентувати доступ до частин таблиці.

Починаючи з версії 8.1.7, у Oracle з'явився інший засіб – Oracle Label Security (OLS). В результаті, реалізація технології мандатного доступу, що ґрунтується на механізмі OLS, стала спиратися не тільки на дискреційну модель доступу (спочатку перевіряються права суб'єкта на виконання відповідної операції над таблицею), але й на механізм VPD (якщо суб'єкт має відповідні привілеї, перевіряється, чи не прикріплені до таблиці якісь політики VPD). І тільки після всього цього перевіряється наявність політик OLS, призначених таблиці, що захищається: порівнюються мітки, присвоєні окремим рядкам, з авторизацією міток користувачів, дозволяючи або забороняючи доступ. Важливим етапом розвитку щодо механізму VPD у OLS стала можливість формувати складову мітку доступу. В інтерпретації Oracle мітка безпеки – це трійка:

$$\lambda = \langle \lambda_1, \lambda_2, \lambda_3 \rangle, \tag{1.2}$$

де λ_1 – елемент лінійно впорядкованої множини Θ_1 (множина рівнів чутливості / таємності). Завдання у мітці доступу рівня таємності є обов'язковим. Теоретично допускається до 10000 рівнів [110, 111]; λ_2 – підмножина елементів з множини Θ_2 (множина відділень). Відділення визначають області, що описують чутливість помічених даних, забезпечуючи більш тонкий рівень деталізації в межах рівня; λ_3 – елемент ієрархічно упорядкованих елементів (дерева) множини Θ_3 (множина груп). Усі дані, що відносяться до певного відділення, можуть мати групу цього відділення у мітці. Групи необов'язкові, але корисні для контрольованого поширення даних та своєчасного реагування на організаційні зміни.

Підсумовуючи вищесказане, можна зробити певні висновки про переваги і недоліки мандатної політики управління доступом та моделі безпеки, побудовані на її основі. Модель Белла-ЛаПадули відіграла величезну роль у розвитку теорії комп'ютерної безпеки та її положення були запроваджені як обов'язкові вимоги до систем, що обробляють інформацію, що містить державну таємницю, у стандартах захищених ІС. Однак при практичній реалізації моделі Белла-ЛаПадула виникає низка проблем, наприклад, таких як [112]:

1) завищення рівня безпеки (для деякої інформації може бути визначений рівень безпеки вище за те, що вона заслуговує);

2) запис наосліп (наприклад, у ситуації, коли суб'єкт здійснює запис об'єкта з вищим рівнем безпеки, операція не порушує правила NWD, проте після її завершення суб'єкт не може перевірити правильність виконання запису об'єкта шляхом виконання контрольного читання, оскільки це порушує правило NRU);

3) привілейовані суб'єкти. Ця проблема пов'язана з роботою системного адміністратора, адміністратора БД (АБД), яка передбачає виконання таких критичних операцій, як додавання та видалення користувачів, відновлення системи після аварій, встановлення програмного забезпечення, усунення помилок. Однак такі операції не вписуються у рамки моделі, що означає неможливість здійснення правильного адміністрування без порушення правил цієї моделі.

Розширення моделі Белла-ЛаПадули [113, 114], пов'язані з пошуком умов та обмежень, що підвищують її безпеку, також не знімають всіх недоліків мандатного доступу. У цілому ж основним недоліком багаторівневих моделей є неможливість управління доступом до конкретних об'єктів на основі обліку індивідуальних особливостей кожного суб'єкта.

Таким чином, обидва розглянуті вище підходи не повною мірою можуть ефективно та гнучко керувати безпечним доступом до даних. Доводиться шукати компроміс між ефективністю, гнучкістю та безпекою. Очевидно, що оптимальне вирішення питань безпеки має вироблятися із застосуванням обох видів моделей.

1.2.1.3. Рольова модель управління доступом

Концепція управління доступом на основі ролей, що поєднує в собі як мандатний підхід до організації доступу – через певну агрегацію суб'єктів та об'єктів доступу, так і дискреційний підхід, що забезпечує гнучкість у налаштуванні системи розмежування доступу в ІС для конкретної предметної області, почалася з розрахованих на багато користувачів систем, що вперше

з'явилися в 1970-х роках [115]. Дещо пізніше з'явилися формальні висловлювання управління доступом на основі ролей (RBAC – *Role-Based Access Control*), що використовуються сьогодні. Основою рольових моделей є введення у суб'єктно-об'єктну модель ІС додаткової категорії активних сутностей – ролей. Рольова модель описує систему у вигляді таких множин: U – множина користувачів; R – множина ролей; P – сукупність повноважень доступу (реалізована, наприклад, у вигляді матриці доступу) до об'єктів; C – множина сеансів роботи користувачів із системою. Рольові відносини встановлюються такими відображеннями множин сутностей системи: $PA \subseteq P \times R$ – відображення множини повноважень на множину ролей, що задає для кожної ролі встановлений набір повноважень (відношення типу «багато до багатьох»); $UA \subseteq U \times R$ – відображення безлічі користувачів на безліч ролей, що визначає набір ролей, доступних даному користувачеві (відношення типу «багато до багатьох»).

Управління доступом у системі здійснюється на основі введення наступних функцій: $user : C \to U$ – функція, яка ставить у відповідність кожному сеансу $c_i \in C$ одного користувача $u \in U$: $u = user(c_i)$ (не змінюється протягом сеансу); $roles : S \to 2^R$ – функція, яка ставить у відповідність кожному сеансу c_i набір ролей з множини R, доступних у даному сеансі $roles(c_i) = \{r \mid (user(c_i), r) \in UA\}$: (набір ролей може змінюватися з часом); $permissions : C \to P$ – функція, яка ставить у відповідність кожному сеансу c_i набір доступних у ньому повноважень: $permissions(c_i) = \bigcup\limits_{r \in roles(c_i)} \{p \mid (p, r) \in PA\}$ інакше кажучи, сукупність повноважень всіх ролей, доступних у цьому сеансі. Взаємозв'язок користувачів, ролей, повноважень (привілеїв) та сеансів показано на рисунку 1.2.

Основне правило (критерій безпеки) рольового доступу визначається наступним чином: система вважається безпечною, якщо і тільки якщо будь-який користувач $u \in U$ в системі, що працює в сеансі $c \in C$, може здійснювати дії, що вимагають повноважень $p \in P$, тільки в тому випадку, якщо $p \in permissions(c)$.

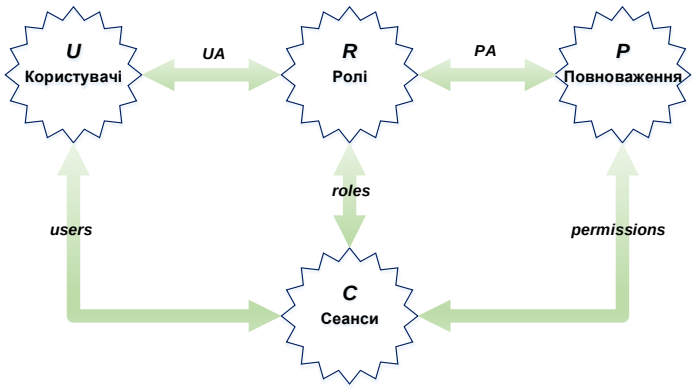


Рисунок 1.2. Взаємозв'язок ролей, повноважень, користувачів та сеансів

На практиці управління доступом в ІС при використанні рольової моделі здійснюється головним чином не за допомогою призначення нових повноважень ролям, а шляхом завдання відношення UA (визначення ролей, доступних даному користувачеві) і функції $roles$, що визначає доступний в сеансі набір ролей. Тому численні інтерпретації рольової моделі [87, 90, 91, 115]: з ієрархічною організацією системи ролей ($PH \subseteq R \times R$); з взаємовиключними на будь-які (всі) сеанси ролями (модель статичного розподілу обов'язків); з взаємовиключними на один сеанс ролями (модель динамічного розподілу обов'язків); з кількісними обмеженнями за ролями; з групуванням ролей і повноважень різняться видом функцій $user$, $roles$ і $permissions$ а також обмеженнями, що накладаються на відношення PA і UA.

Використання рольової моделі дозволяє підвищити ефективність адміністрування складних ІС, тому цей підхід є затребуваним, у тому числі для СКБД. Ролі дозволяють надавати безліч привілеїв однією простою командою. Це стало можливим завдяки включенню до стандарту SQL підтримки концепції ролей, як іменованого набору привілеїв [102]. Проте слід зазначити, що підтримка ролей у різних реалізаціях SQL дещо відрізняється. До того ж, не всі СКБД підтримує механізм ролей. Серед сучасних СКБД рольова модель розмежування доступу підтримується, наприклад, у Microsoft SQL Server, Oracle, IBM DB2, PostgreSQL, MySQL 8.0 та деяких інших.

Отже, ролі дозволяють реалізувати гнучкі правила розмежування доступу, що динамічно змінюються у процесі функціонування ІС. Ефективність ролей

особливо помітно проявляється при організації доступу до ресурсів складних ІС з великою кількістю користувачів та об'єктів (у тому числі завдяки можливості побудови ієрархій ролей). Оперувати ролями набагато зручніше, ніж суб'єктами. Такий підхід дозволяє отримувати прості та зрозумілі правила контролю доступу, які можуть бути застосовані на практиці. Однак, на відміну від інших розглянутих раніше моделей, ця модель практично не гарантує безпеку за допомогою формального доказу, а лише визначає характер обмежень, дотримання яких і є критерієм безпеки системи [91]. Тому безпека рольових моделей ґрунтується на контрольних механізмах дискреційних чи мандатних моделей, засобами яких регулюється доступ рольових суб'єктів до об'єктів системи. До недоліків рольового розмежування доступу також слід віднести: можливість внесення надмірності (дублювання) при наданні користувачам прав доступу, складність конструювання ролей.

1.2.2. Модель забезпечення цілісності даних Кларка-Вілсона

Як зазначалося у п. 1.2, забезпечення інформаційної безпеки неможливе без розгляду концепції захисту достовірності та коректності даних, що становить суть забезпечення їхньої цілісності. Порушення цілісності не обмежуються навмисними атаками. Помилка користувача, недогляд або невмілі дії також є причиною багатьох випадків несанкціонованої зміни інформації. Події, які призводять до порушень цілісності, включають зміну або видалення файлів, даних у БД, введення невірних даних, зміну конфігурації, помилки в командах, впровадження вірусу та виконання шкідливого коду. Порушення цілісності може статися через дії будь-якого користувача, включаючи адміністраторів, як через недогляд у політиці безпеки і, так і через неправильно налаштований контроль безпеки. Тому цілісність доцільно досліджувати із трьох сторін [60, 61]:
1) запобігання (перешкоди) внесення змін неавторизованими суб'єктами;
2) запобігання внесенню авторизованими суб'єктами несанкціонованих змін, наприклад, помилок; 3) підтримання внутрішньої та зовнішньої узгодженості

об'єктів, щоб їх дані були правильним та істинним відображенням реального світу, а будь-які стосунки (зв'язки) з будь-яким дочірнім, рівним чи батьківським об'єктом були дійсними, узгодженими та такими, що перевіряються. Правильно реалізований захист цілісності надає засоби для авторизованих змін, одночасно захищаючи від зловмисних несанкціонованих дій (таких як віруси та вторгнення), а також від помилок, допущених авторизованими користувачами (таких як помилки або недогляди). Це гарантує, що дані залишаються правильними (відсутні логічні помилки у структурі та значеннях даних), незмінними (тотожність даних певному еталону), неспотвореними (відсутність підробки даних) і збереженими. Якщо механізм безпеки забезпечує цілісність, він забезпечує високий рівень гарантії того, що дані, об'єкти та ресурси не будуть змінені порівняно з їх початковим захищеним станом.

Залежно від важливості аналізованого аспекту цілісності та області використання даних, як зазначалося раніше, існують різні методи та засоби [87]. Так правильність, неспотвореність і незмінність даних може забезпечуватися з допомогою методів і засобів технологій розмежування доступу, що ґрунтуються на формальних моделях цілісності. Неспотвореність даних при зберіганні та передачі в ІС може забезпечуватись за рахунок криптографічних примітивів, таких як: цифровий підпис, криптографічні геш-функції, коди автентифікації.

Для кращого розуміння та уявлення умов, яким має підпорядковуватися поведінка системи, щоб забезпечити її цілісність, розумно звернутися до однієї з найвідоміших формальних моделей цілісності даних – моделі, запропонованої Кларком з Вілсоном [116]. Ця модель – це сукупність практичних рекомендацій щодо побудови системи забезпечення цілісності в ІС, основа та керівництво для формалізації політик безпеки, в якій наголошується на важливості затвердження керівництвом процесів та політик безпеки, яким має слідувати організація [117]. Основні компоненти цієї моделі: S – множина суб'єктів; D – множина даних в ІС (множина об'єктів), причому $D = CDI \cup UDI$, $CDI \cap UDI = \varnothing$, де CDI («*обмежений елемент даних*») – дані (будь-який елемент даних), цілісність яких контролюється (захищена моделлю безпеки); UDI («*необмежений елемент*

даних») – дані, цілісність яких не контролюється моделлю безпеки; *IVP* – процедура перевірки цілісності *CDI* (процедура, яка сканує елементи даних та підтверджує їх цілісність, наприклад, шляхом розрахунку контрольної суми); *TP* – процедура перетворення – компонент, який може ініціювати транзакцію (послідовність операцій), що переводить систему з одного стану до іншого. Процедури перетворення – єдині процедури, яким дозволено змінювати *CDI* . Обмежений доступ до *CDI* через *TP* становить основу моделі цілісності Кларка-Вілсон. Модель Кларка-Вілсона ґрунтується на трійках [60, 87, 118]: «*суб'єкт – операція (транзакція), яка не порушує цілісність – об'єкт*». Суб'єкти не мають прямого доступу до об'єктів. Доступ до об'єктів можна отримати лише через *TP* . У моделі виділяються два основних механізми, що забезпечують базовий контроль доступу та цілісність. А саме, правильно сформована транзакція зберігає цілісність даних та запобігає довільному маніпулюванню даними суб'єктами. Слід зазначити, що концепція правильно сформованої транзакції добре вписується в стандартну концепцію транзакцій у традиційних СКБД [38]. Поділ обов'язків вимагає, щоб кожна критична операція складалася з двох або більше частин, кожна з яких повинна виконуватись іншим суб'єктом чи суб'єктом з іншою роллю. Модель складається з двох наборів правил: правил сертифікації (С), що проводиться адміністратором безпеки, власником системи та правил виконання (Е), яке здійснюється системою. Правила виконання відповідають функціям безпеки, незалежним від додатків, а правила сертифікації дозволяють включати до моделі визначення цілісності для конкретних застосунків. Іншими словами, правила виконання визначають вимоги безпеки, які мають підтримуватись механізмами захисту в базовій системі (в СКБД). Правила сертифікації визначають вимоги безпеки, яких повинна дотримуватись прикладна система (в даному випадку це запропоновані рішення, що враховують особливості та можливості СКБД). Бажано мінімізувати правила сертифікації, оскільки процес сертифікації складний, схильний до помилок і повинен повторюватися після кожної зміни процедури перетворення (програми). Дещо перефразовані щодо оригіналу правила моделі Кларка-Вілсона наведені нижче:

1. Правило С1 (С1). У системі повинні бути IVP, здатні підтвердити цілісність будь-якого CDI.

2. (С2). Всі процедури перетворення TP повинні бути добре сформованими транзакціями, тобто не порушувати цілісності даних, і застосовуватися лише до списку елементів CDI, що встановлюються адміністратором безпеки (відношення $(TP_i,(CDI_a,CDI_b,CDI_c,...))$).

3. (Е1). Система повинна контролювати допустимість застосування TP до елементів CDI відповідно до списків, зазначених у правилі С2.

4. (Е2). Система повинна підтримувати список дозволених конкретним користувачам процедур перетворення TP із зазначенням допустимого для кожної $TP_i \in TP$ та даного суб'єкта ($s_j \in S$) набору елементів CDI, що обробляються (тобто трійки: $(s_j,TP_i,(CDI_a,CDI_b,CDI_c,...))$).

5. (С3). Список, визначений правилом Е2, має відповідати вимозі розмежування функціональних обов'язків (зокрема спільного виконання).

6. (Е3). Система повинна автентифікувати кожен суб'єкт, який намагається виконати будь-яку процедуру перетворення TP.

7. (С4). Кожне застосування TP має реєструватися в спеціальному елементі CDI – журналі реєстрації, що містить інформацію, достатню для відновлення повної картини кожного застосування цієї процедури перетворення, і доступному лише для додавання інформації.

8. (С5). Будь-яка TP, яка приймає UDI як вхідні дані, може виконувати тільки дійсні (англ. valid) транзакції для всіх можливих значень UDI. TP або приймає (конвертує до CDI), або відхиляє UDI. Тобто спеціальні TP можуть коректно обробляти UDI, перетворюючи їх на CDI.

9. (Е4). Тільки спеціально уповноважений суб'єкт (користувач, агент, якому можна сертифікувати об'єкти) може змінювати списки, визначені в правилах С3 та Е2. Цей суб'єкт (агент), який має право сертифікувати сутність (відповідні списки), може не мати жодних прав на виконання щодо цієї сутності.

На рисунку 1.3 представлена схема застосування даних правил для керування роботою системи та даними. *UDI* представляють дані, що існують поза захищеною системою. Правила сертифікації забезпечують правильну перевірку таких даних під час входу до системи. Наприклад, правило С5 вимагає, щоб правильно сформовані *TP*, які перетворюють *UDI* на *CDI*, виконували тільки перевірені перетворення. Правила С1 та С2 вимагають, щоб *CDI* задовольняли вимогам цілісності у початковому стані та після подальших перетворень. Правило С4 вимагає реєстрації всіх транзакцій, як це зазвичай буває з базами даних. Ведення журналу бази даних більшою мірою призначене для відновлення даних після збою, відмови (для відкату – повернення до попереднього стану), а ведення журналу моделі Кларка-Вілсона – для аудиту. Хоча у БД може вестись і журнал аудиту.

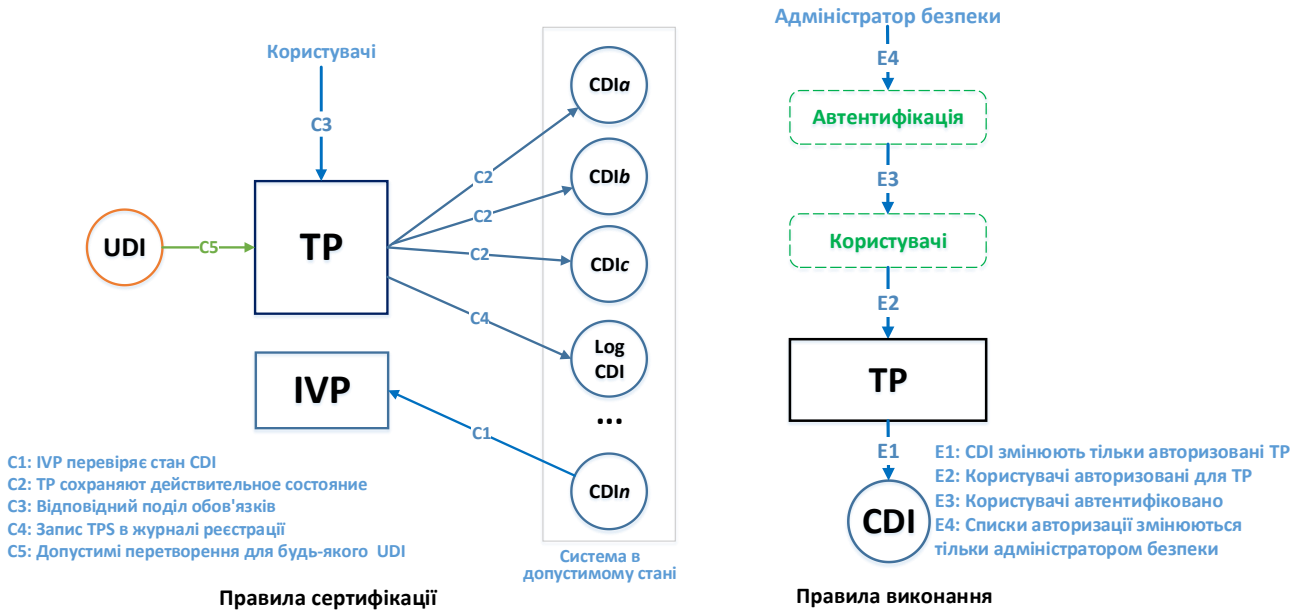


Рисунок 1.3. Схема застосування правил моделі Кларка-Вілсона

Правило С3 потребує відповідного розподілу обов'язків. Оскільки дані можуть бути введені лише відповідно до правил сертифікації, для систем, які нас цікавлять, слід, що всі дані у базі даних мають бути *CDI*. Правила виконання запобігають зміні *CDI* способами, що суперечать *IVP*. Правила E2 – E4 відносяться до авторизації доступу *TP*. У той час як E1 гарантують, що тільки сформовані сертифіковані (перевірені) *TP* можуть використовуватися для зміни

CDI . Основний недолік, який зазвичай згадується для моделі Кларка-Вілсона, полягає в тому, що *IVP* і пов'язані з ними методи непросто реалізувати в реальних комп'ютерних системах [118].

Наприклад, основною проблемою реалізації механізмів контролю цілісності файлових об'єктів є їх сильний вплив на завантаження обчислювального ресурсу системи, що обумовлюється такими причинами [119]: по-перше, може знадобитися контроль великих обсягів інформації, що пов'язано зі значною тривалістю виконання процедури *IVP* ; по-друге, може знадобитися безперервна підтримка файлового об'єкта в еталонному стані. У зв'язку з чим виникає закономірне питання: якою має бути періодичність запуску процедури *IVP* ? Якщо виконувати її часто, це призведе до істотного зниження продуктивності системи, якщо рідко, ефективність такого контролю може бути низькою. Тому одним із основних завдань при реалізації механізмів контролю цілісності файлових об'єктів є вибір принципів та механізмів запуску процедури перевірки цілісності *CDI* . Інша проблема реалізації механізму контролю цілісності – це контроль цілісності самої контролюючої програми, якщо контроль цілісності реалізується програмно. Все це вимагає певного додаткового опрацювання та прийняття відповідних рішень, що залежать, як правило, від особливостей конкретних ІС, БД. Однак у контексті СКБД зазначений вище загальний недолік моделі Кларка-Вілсона, зумовлений складністю реалізації та пов'язаних з ними методами, значною мірою можна подолати. Так, наприклад, для реляційних СКБД деякі обмеження цілісності закладені в теорії: цілісність сутностей, посилальна цілісність. Інші можуть бути вказані як статичні обмеження за допомогою SQL (так звана декларативна підтримка обмежень цілісності). Треті – як динамічні обмеження цілісності (так звана процедурна підтримка обмежень цілісності), які можуть бути реалізовані за допомогою тригерів та програм, що зберігаються. Всі вони забезпечують цілісність *CDI* , до яких здійснюється доступ та їх модифікація за допомогою процедур перетворення *TP* .

Таким чином, традиційні СКБД підтримують багато механізмів моделі Кларка-Вілсона. Однак реалізації, що базуються на стандартному SQL, вимагають

деяких компромісів. Наприклад, популярний принцип поширення (надання) прав доступу `WITH GRANT OPTION` суперечить моделі Кларка Вілсона (правилу Е4). Актуальними для СКБД також є питання, пов'язані з механізмами контролю цілісності збережених процедур, функцій (як файлових об'єктів). Це зумовлює необхідність проведення додаткових досліджень у відповідних напрямках. В цілому ж, безумовними перевагами цієї моделі є її відносна простота і легкість спільного використання з іншими моделями безпеки.

Отже, аналізуючи різноманітність формальних моделей та безліч підходів до їх реалізації досить складно визначитися, яка з них є кращою. І це природно, тому що кожна з них має свої переваги, які необхідно використовувати саме в конкретній ситуації, так само як і звести до мінімуму недоліки кожної з них. При цьому слід розуміти, що сама модель безпеки не забезпечує захист, а лише надає принцип побудови захищеної ІС, реалізація якого і повинна забезпечити закладені в моделі властивості безпеки. Безпека системи рівною мірою визначається трьома факторами [91]: властивостями самої моделі (одної або декількох), її (їх) адекватністю загрозам, що впливають на систему, і тим наскільки вона (вони) коректно реалізована(і). Крім того, як відомо, існуючі теоретичні розробки та практичні реалізації забезпечення безпеки ІС ґрунтуються не лише на парадигмі формального моделювання політики безпеки, а й на іншій не менш важливій парадигмі – криптографії, націленій на вирішення певних завдань. Причому ці різні за походженням та розв'язуваними завданнями підходи доповнюють один одного: криптографія пропонує актуальні методи та примітиви для захисту інформації, забезпечуючи ідентифікацію, автентифікацію, шифрування, контроль цілісності даних, а формальні моделі безпеки надають розробникам захищених ІС основоположні принципи, які лежать в основі архітектури захищеної системи та визначають концепцію її побудови [91].

Тому доцільним є проведення подальших досліджень, результатом яких була б деяка методологія комплексного використання різних моделей безпеки та криптографії при проектуванні та експлуатації відповідних ІС та їх основного функціонального компонента – БД, що веде до підвищення ефективності їх

захисту, яку можна охарактеризувати рівнем (ступенем) захищеності як здатності протистояти навмисним та ненавмисним діям з боку законних користувачів та зловмисників в умовах можливих загроз. У тому числі актуальними є дослідження у напрямках:

– створення, використання моделей захисту та оцінки безпеки БД;

– розширення підходів для вирішення проблеми логічного висновку;

– забезпечення конфіденційності, цілісності збережених даних та спеціальним чином оформлених програм за рахунок комплексного використання формальних моделей безпеки та криптографічних методів з метою забезпечення гарантії того, що дані та постійно збережені модулі залишаються правильними (відсутні логічні помилки у структурі та у значеннях), незмінними (тотожність певному еталону), неспотвореними (відсутність підробки),
що проводяться та розглядаються в даній роботі.

1.3. Висновки за розділом

1. Аналіз ключових проблем, сучасного стану та розвитку технологій баз, сховищ даних показав, що існуючі підходи до забезпечення безпеки даних, що зберігаються і оброблюються в них, не повною мірою відповідають відповідним вимогам різних міжнародних, вітчизняних стандартів та інших законодавчих актів у галузі ІБ. Без необхідного захисту баз та сховищ даних, нові інформаційні технології внаслідок помилкових дій з боку законних користувачів та власників, а більшою мірою протиправних дій з боку зловмисників, здатні порушити не лише приватне життя людей, а й діяльність великих організацій. Тому в ситуації, що склалася, беручи до уваги сучасний стан розвитку технологій баз, сховищ даних, науково-практичні досягнення в галузі інформаційної безпеки, кваліфікацію зловмисників, доцільним є перегляд підходу до вирішення проблеми управління даними та забезпечення їх безпеки. Тобто є необхідність вирішення нових науково-прикладних завдань. Одним з таких актуальних завдань є завдання розробки моделей, методів і засобів, що дозволяють підвищити захищеність баз

даних, побудованих на основі схеми з універсальним базисом відношень, які можуть використовуватися як звичайна БД, сховище даних різних ПрО або конфігураційна БД середовища управління простором даних. Зважаючи на те, що такі БД можуть використовуватися в різній якості, пропоновані моделі, методи та засоби можуть надалі дозволити розробити деяке цілісне рішення, що забезпечує безпеку РБД. Причому окремі елементи цього рішення можуть бути використані для захисту баз та сховищ даних із різними моделями.

2. Проведений аналіз формальних моделей, як методологічної основи побудови систем захисту та оцінки їх безпеки, дозволив виявити їх основні переваги та слабкі сторони, які в залежності від конкретної ситуації доцільно використовувати розумно та комплексно, виходячи з того факту, що сама модель безпеки не забезпечує захист, а лише надає принцип побудови захищеної ІС, БД, реалізація якого має забезпечити закладені у моделі властивості безпеки. Безпека системи однаковою мірою визначається: властивостями самої моделі (однієї або кількох), її (їх) адекватністю загрозам, що впливають на систему, і тим, наскільки вона (вони) коректно реалізована(і).

3. Перевагою моделей безпеки, побудованих на основі дискреційної політики управління доступом, є їхня простота і гнучкість. Ці моделі доцільно застосовувати під час проведення формальної верифікації коректності побудови систем розмежування доступу до захищених ІС та БД. При цьому слід пам'ятати, що таким моделям властиві певні недоліки, що обмежують їхнє застосування, а саме: статичність встановлених правил керування; неможливість контролювати повною мірою потоки даних між об'єктами; складність відстеження наданих суб'єктам привілеїв за їх великої кількості та неконтрольованої передачі прав від одного суб'єкта до іншого; відсутність механізму управління доступом до конфіденційної інформації.

4. Важливою вигідною відмінністю мандатного управління доступу від дискреційного є те, що в мандатній моделі контролюються не операції, що їх суб'єкт виконує над об'єктом, а потоки інформації. Однак, при практичній реалізації цих моделей може виникнути низка проблем: завищення рівня безпеки;

запис наосліп; неможливість здійснення правильного адміністрування без порушення правил цієї моделі. Крім того, основним недоліком багаторівневих моделей є неможливість управління доступом до конкретних об'єктів на основі обліку індивідуальних особливостей кожного суб'єкта.

5. Моделі безпеки на основі рольової політики дозволяють реалізувати гнучкі (що динамічно змінюються у процесі функціонування ІС, БД) правила розмежування доступу. Ефективність цих моделей особливо помітно проявляється при організації доступу до ресурсів систем із великою кількістю користувачів та об'єктів. Разом з тим, у рольових моделях немає суворих доказів безпеки системи відповідно до певних формалізованих критеріїв, а при наданні користувачам прав доступу їм властива певна надмірність (дублювання).

6. Безумовними перевагами моделі Кларка-Вілсона, як основи та керівництва для формалізації політик безпеки, що забезпечують цілісність, є її відносна простота та легкість спільного використання з іншими моделями безпеки. Основним недоліком даної моделі вважається складність реалізації в реальних ІС методів та процедур перевірки цілісності даних, хоча в контексті традиційних баз даних цього недоліку можна уникнути. Однак для БД актуальними залишаються питання, пов'язані з механізмами контролю цілісності збережених процедур, функцій, що обумовлює необхідність проведення додаткових досліджень у відповідних напрямках.

7. Існуючі теоретичні розробки та практичні реалізації забезпечення безпеки ІС, БД ґрунтуються на двох парадигмах: формального моделювання політики безпеки та криптографії. Причому ці різні за походженням і вирішуваним завданням підходи доповнюють один одного. Тому доцільним є проведення подальших досліджень, результатом яких була б деяка методологія комплексного використання різних парадигм, що веде до підвищення ефективності захисту баз даних.

Основні питання, розглянуті у розділі, опубліковані у роботах [47, 59, 110, 120-126].

2. РОЗРОБКА МОДЕЛІ ЗАХИСТУ ТА МЕТОДУ ОЦІНКИ БЕЗПЕКИ БАЗИ ДАНИХ

Як відомо [60, 61, 127, 128], інформаційна безпека зосереджена на тріаді CIA (Confidentiality, Integrity, Availability): конфіденційність, цілісність, доступність. Безпека цих трьох характеристик дуже важлива, незважаючи на те, що сьогодні все частіше модель CIA тріади зазвичай розглядається як більш неадекватна в умовах, що постійно змінюються. Загрози конфіденційності, цілісності та доступності інформації перетворилися на великий набір подій, включаючи випадкове або навмисне ушкодження, руйнування, крадіжку, ненавмисну або несанкціоновану зміну або інше неправомірне використання з боку людей або інших загроз. Цей широкий спектр загроз, що постійно розвиваються, стимулював розробку більш адекватної моделі, що враховує складності поточного середовища інформаційної безпеки. Тим не менш, враховуючи той факт, що розширені моделі інформаційної безпеки еволюціонують з моделі CIA, а термінологія тріад використовується досить широко (крім того, окремі характеристики, наприклад, справжність / автентичність (англ. authenticity), підзвітність / спостереженість (англ. accountability), неспростовність (англ. non-repudiation), можуть розглядатися як деякі аспекти цілісності [60]), далі більшою мірою питання безпеки баз даних будуть розглядатися саме в термінології цих основних властивостей інформації як об'єкта захисту. При цьому, враховуючи дуже широке коло розглянутих проблем, пов'язаних з повним комплексним забезпеченням безпеки інформаційних систем в цілому, у цій роботі більша увага приділяється більш вузькій проблемі, а саме, дослідженню моделі захисту та оцінки безпеки БД, забезпеченню конфіденційності даних, що зберігаються, а також цілісності даних, баз даних та програмного забезпечення СКБД, з акцентом на БД, побудованим на основі схеми з універсальним базисом відношень, через можливість знаходження деякого цілісного рішення, що забезпечує безпеку реляційних баз даних. Так як саме такі БД можуть використовуватися в різній якості – як звичайна база даних, але при цьому, яку досить просто адаптувати до

динамічних змін в ПрО, сховище даних різних предметних областей, що істотно відрізняються, або конфігураційна БД середовища управління простором даних. До того ж окремі елементи такого рішення можуть бути використані для захисту баз та сховищ даних із різними моделями.

2.1. Формалізація задачі забезпечення безпеки бази даних на основі системи захисту з повним перекриттям

Безпека (захищеність) є однією з найважливіших характеристик якості інформаційних систем загалом [129] та баз даних, як їх основної складової, зокрема. Тому наявність системи захисту інформації (СЗІ), як комплексу програмних, технічних, криптографічних, організаційних та інших методів, засобів та заходів, що забезпечують цілісність, конфіденційність, автентичність та доступність інформації в умовах впливу на неї загроз природного чи штучного характеру, є невід'ємною рисою будь-якої сучасної ІС, БД. При цьому високий рівень безпеки даних має бути забезпечений без зниження функціональності ІС, БД і практично без ускладнення роботи користувача в системі [90]. Разом з тим, щоб можна було б перевірити висновки щодо ступеня забезпечення безпеки, її необхідно будь-яким чином виміряти.

Аналіз різних підходів і досягнень у галузі оцінки безпеки інформаційних систем в цілому та баз даних, зокрема, [7, 27, 38, 42, 44, 70-81, 83-86, 88, 130-147] показав, що ці підходи та методи більшою мірою засновані на інтуїції, є емпіричними та розрізненими. Водночас було б корисним мати певний науково-методологічний, загальний підхід до вирішення цієї проблеми. Тому, спираючись на результати даного аналізу, було зроблено висновок про можливість та доцільність в якості основи моделі захисту БД та оцінки її безпеки використовувати модель Клементса–Гофмана [85, 88]. Дана модель спирається на теорію графів, нечітких множин і ймовірностей. Вона традиційно вважається основою формального опису систем захисту. А як відомо, завдяки формальним

моделям можна, спираючись на об'єктивні та незаперечні положення математичної теорії, довести безпеку системи.

Основним положенням м*оделі системи безпеки з повним перекриттям* (модель Клементса–Гофмана) є теза про те, що система, спроектована на її підставі, повинна мати принаймні один захід (механізм, метод, засіб) для забезпечення безпеки на кожному можливому шляху проникнення у систему. У моделі розглядається взаємодія «області загроз», «області, що захищається» та «системи захисту». Вважається, що несанкціонований доступ (як будь-який доступ, що порушує заявлену політику безпеки [148]) до кожного з набору об'єктів O області, що захищається, пов'язаний з деякою величиною збитку, який може бути визначений кількісно (інакше його вважають рівним деякій умовній величині). При цьому кількісна категорія збитків може бути виражена у вартісному еквіваленті (сума фінансових втрат), або в термінах, пов'язаних з цільовою функцією системи (наприклад, часу, необхідного для відновлення функціональних можливостей ІС в цілому, та БД зокрема, після зловмисного впливу) [112].

Для опису системи безпеки з повним перекриттям стосовно баз даних спочатку введемо такі позначення:

– $T = \{t_i\}$, $i = 1..I$ – множина загроз безпеці БД. Для формування набору загроз, спрямованих на порушення безпеки, за можливістю визначаються усі потенційні зловмисні дії по відношенню до усіх об'єктів безпеки;

– $O = \{o_j\}$, $j = 1..J$ – множина об'єктів БД, що захищаються;

– $W = \{w_k\}$, $k = 1..K$ – множина заходів забезпечення безпеки (у тому числі методів, засобів, механізмів, що забезпечують реалізацію політик безпеки, формальним представленням яких є моделі безпеки, а також методів та примітивів для захисту інформації БД, заснованих на криптографії).

Елементи усіх перелічених вище множин перебувають між собою у певних відносинах, причому зв'язок між загрозами та об'єктами не є зв'язком «один до одного». Загроза $t_i \in T$ може поширюватися на будь-яку кількість об'єктів O, а

об'єкт $o_j \in O$ може бути вразливий з боку більш ніж однієї загрози T. Для кращого розуміння систему захисту у межах даної формалізації доцільно подати дводольним графом (рисунок 2.1), у якому безліч відносин «загроза – об'єкт» представляється як дуги (t_i, o_j), існуючих лише тоді, коли t_i є загрозою, спрямованої на порушення безпеки об'єкта $o_j \in O$.

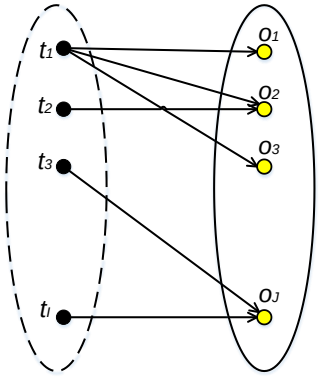


Рисунок 2.1. Уявлення відносини «об'єкт – загроза»

Захист забезпечується шляхом перекриття усіх можливих дуг графа за рахунок створення відповідного бар'єра (засобу забезпечення безпеки $w_k \in W$) на кожному шляху. В результаті дводольний граф перетворюється на тридольний (рисунок 2.2).

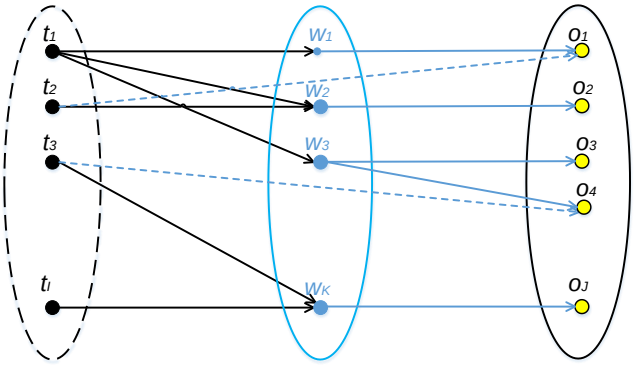


Рисунок 2.2. Уявлення відносин між загрозами, засобами забезпечення безпеки та об'єктами

У захищеній системі усі дуги моделі видаються як (t_i, w_k) і (w_k, o_j). Будь-яка дуга у формі (t_i, o_j) визначає незахищений об'єкт (дуги (t_2, o_1) та (t_3, o_4) на рисунку 2.2).

Формально умову забезпечення повної захищеності можна уявити як: $\forall(t_i) \in T$, $\exists(w_k) \in W$. Тобто для кожної загрози $t_i \in T$ існує засіб забезпечення безпеки $w_k \in W$, що перекриває шлях реалізації цієї загрози до об'єкту захисту o_j. При цьому слід зауважити, що один і той же засіб (захід) забезпечення безпеки може перекривати більше однієї загрози та / або захищати більше одного об'єкта [88]. Однак надалі скористаємося розширенням даної моделі (а саме так званою базовою системою забезпечення безпеки Клементса), описаної в роботах [85, 88], у вигляді 5-мірного кортежу (п'ятірки):

$$S' = \{O, T, W, V, B\}, \tag{2.1}$$

яка передбачає включення набору / множини вразливостей V (що являють собою шляхи реалізації загроз T щодо об'єктів O), що визначається підмножиною декартового добутку $V = T \times O$ (набором упорядкованих пар $v_r = (t_i, o_j)$, $r = 1..R$), і набору бар'єрів (що представляють собою точки, в яких потрібно здійснювати захист) B, що визначається підмножиною декартового добутку $B = V \times W = T \times O \times W = \{b_l = (t_i, o_j, w_k),\ l = 1..\Lambda\}$, як відображення $T \times O \times W$ на набір впорядкованих трійок $b_l = (t_i, o_j, w_k)$.

Для даної моделі умову повного перекриття можна записати у такому вигляді: $\forall(v_r = (t_i, o_j)) \in V$, $\exists(b_l = (t_i, o_j, w_k)) \in B$. Ця умова означає, що для кожного шляху реалізації загроз T щодо об'єктів O засобом безпеки $w_k \in W$ створюється бар'єр $b_l \in B$, який усуває цю загрозу для конкретного об'єкта.

В ідеалі кожен механізм захисту (захисні заходи, заходи безпеки – англ. security controls, security measures) повинен унеможливлювати відповідний шлях реалізації загрози. На практиці ж ці механізми забезпечують лише деякий ступінь опірності загрозам безпеці (наприклад, паролі мають кінцеву довжину; шифри мають різну криптографічну стійкість; різна частота точок синхронізації між базою даних та журналом транзакцій призводить до всіляких іноді неприйнятних часів відновлення при збоях, відмовах; залежність захищеності від актуальності та своєчасності встановлюваних оновлень, параметрів конфігурації тощо).

2.2. Показник захищеності бази даних

Щоб мати деяку кількісну оцінку рівня захищеності об'єктів, автори моделі системи безпеки з повним перекриттям [85, 88] вважають, що можна виміряти рівень забезпечення безпеки системи. В якості підходящої структури для вираження таких заходів вони пропонують лінгвістичну змінну. Для цього вони перевизначають бар'єри безпеки B, кожен із яких ($b_l \in B$) представляють як складову лінгвістичну змінну, компонентами якої є лінгвістичні змінні: P_l – ймовірність виникнення загрози; L_l – величина шкоди при успішній реалізації загрози щодо об'єкта, що захищається; R_l – ступінь опірності засобу захисту w_k, що характеризується ймовірністю його подолання. При цьому зазначається, що ці компоненти оцінюються в контексті специфічного бар'єру ($b_l = (t_i, o_j, w_k)$), який вони формують (індекси у P_l, L_l, R_l такі ж як індекс бар'єру, а не такі як у компонентів бар'єру $b_l = (t_i, o_j, w_k)$ в базовій системі захисту – загрози, об'єкти та засоби захисту). Однак, автори пояснюючи, що значення опірності визначає ступінь підвищення або зниження загальної безпеки системи, а неформальна комбінація ймовірності та величини втрат дає важливість (вагу) бар'єру у зведеному рейтингу (оцінці), і в цілому ці значення визначають внесок бар'єру у загальну безпеку системи, нічого не говорять про конкретні способи їх отримання (оцінювання), а також про існування, вид та використання інтегрального показника, що дозволяє оцінювати захищеність об'єктів та системи в цілому.

Тому проаналізувавши різні підходи, викладені в релевантних джерелах [149-152], в якості такого показника було обрано залишковий ризик Rr (ризик, що залишається після обробки ризику [153]), пов'язаний з можливістю реалізації загрози $t_i \in T$ щодо об'єкта БД $o_j \in O$ при використанні засобу безпеки $w_k \in W$. Природно, що кількісний підхід до оцінки ризику краще якісного, оскільки він пропонує більш відчутну оцінку ситуації [44]. Величину залишкового ризику, що характеризує стійкість бар'єру $b_l \in B$, можна визначити так [124, 149-151]:

$$Rr_l = P_l L_l (1 - R_l). \qquad (2.2)$$

Залишковий ризик за суттю є мірою незахищеності активу. Тоді величину захищеності БД можна визначити шляхом обчислення зворотної величини сумарного залишкового ризику подібно [149-151]:

$$S = \sum_{\forall b_l \in B} \frac{1}{P_l L_l (1 - R_l)}, \tag{2.3}$$

де $P_l, L_l \in (0,1)$, $R_l \in [0,1)$.

За відсутності в системі бар'єрів b_l, що перекривають певні шляхи реалізації загроз щодо об'єктів, ступінь опірності механізму захисту R_l приймається нульовим. З формального боку це можна уявити шляхом введення так званого засобу захисту з нульовим ступенем забезпечення безпеки (w_o), що додається до множини W [85, 88]. Кожному незахищеному об'єкту приписується такий засіб. Таким чином, для $\forall (t_i, o_j) \in V$, для якого $(\forall k \in K)\ (t_i, o_j, w_k) \notin B$ до B додається бар'єр (t_i, o_j, w_o).

2.3. Удосконалена модель Клементса–Гофмана для баз даних

Автори [85], вводячи поняття вразливості (англ. vulnerability), формально представляють його як відображення $T \times O$ на набір упорядкованих пар $v_r = (t_i, o_j)$, а не окремо об'єктивно існуючу категорію вразливість, як слабкого місця активу або засобу управління, яке може бути використане однією або більшою загрозою [153]. Загрози існують окремо від слабких місць активу. Вразливість сама по собі не завдає шкоди (збитку), це лише умова або набір умов, що дають змогу заподіяти шкоду (збиток) активам. У разі реалізації загрози може використовуватися одна або більше вразливостей активу [154]. При цьому один тип вразливості може призвести до багатьох загроз безпеці різної спрямованості. Тому загрози та вразливості (як тип, а не як екземпляр для конкретного активу) доцільно розглядати у комплексі. Тільки разом вони можуть спричинити небажаний інцидент, який може завдати шкоди системі (активам). І в цьому випадку необхідно чітко визначити загрози, вразливості та взаємозв'язок між

ними. У зв'язку з цим розширимо представлену вище модель з повним перекриттям до 6-мірного кортежу (шістки) за рахунок включення множини вразливостей (слабких місць) об'єктів (Γ):

$$S = \{O, T, \Gamma, W, V, B\}. \quad (2.4)$$

Тоді після відповідного уточнення моделі під набором V будемо розуміти безліч упорядкованих трійок $v_r = (t_i, \gamma_\psi, o_j)$, $\psi = 1..\Psi$, де $\gamma_\psi \in \Gamma$ – вразливість (як певний її тип), яка використовується загрозою $t_i \in T$, спрямованою на порушення безпеки об'єкта $o_j \in O$. Набір бар'єрів відповідно визначатиметься як: $B = V \times W = T \times \Gamma \times O \times W = \{b_l = (t_i, \gamma_\psi, o_j, w_k), \quad l = 1..L\}$. А умова забезпечення повної захищеності для цієї моделі набуде наступного вигляду: $\forall (v_r)$, $\exists (b_l = (t_i, \gamma_\psi, o_j, w_k)) \in B$. Ця умова означає, що для кожної трійки (t_i, γ_ψ, o_j) з множини V створюється бар'єр $b_l \in B$, що унеможливлює реалізацію небажаного інциденту (реалізацію загрози $t_i \in T$, яка використовує вразливість $\gamma_\psi \in \Gamma$) щодо об'єкта захисту $o_j \in O$. Засіб захисту з нульовим ступенем забезпечення безпеки (w_o), що додається до множини W і приписується до незахищеного об'єкта, формально можна виразити таким чином: для $\forall (t_i, \gamma_\psi, o_j) \in V$, для якого $(\forall k \in K)\ (t_i, \gamma_\psi, o_j, w_k) \notin B$, до B додається бар'єр $(t_i, \gamma_\psi, o_j, w_o)$. Відповідно у виразах (2.2), (2.3) під ймовірністю P_l розумітиметься ймовірність небажаного інциденту (реалізації загрози), як добуток ймовірності виникнення загрози P_{t_i} (так званої мотиваційної складової ймовірності реалізації загрози [155]) на ймовірність використання (вдалого) вразливості P_{γ_ψ} : $P_l = P_{t_i} \cdot P_{\gamma_\psi}$ [152, 156]. А величину шкоди (збитку) L_l щодо об'єкта, що захищається, слід розглядати з позиції успішної реалізації загрози t_i, яка використовує вразливість γ_ψ.

Для кращого представлення (розуміння) взаємозв'язку основних елементів системи безпеки, що розглядається, на рисунку 2.3 у вигляді діаграми класів у нотації UML (англ. Unified Modeling Language) наведені ці високорівневі поняття

безпеки та їх взаємозв'язок. Зв'язки між елементами системи безпеки, що розглядаються, це зв'язки типу «багато до багатьох», що підрозділяються на так звані асоціації (англ. association) – представлені прямими лініями, і залежності (англ. dependency) – пунктирні лінії.

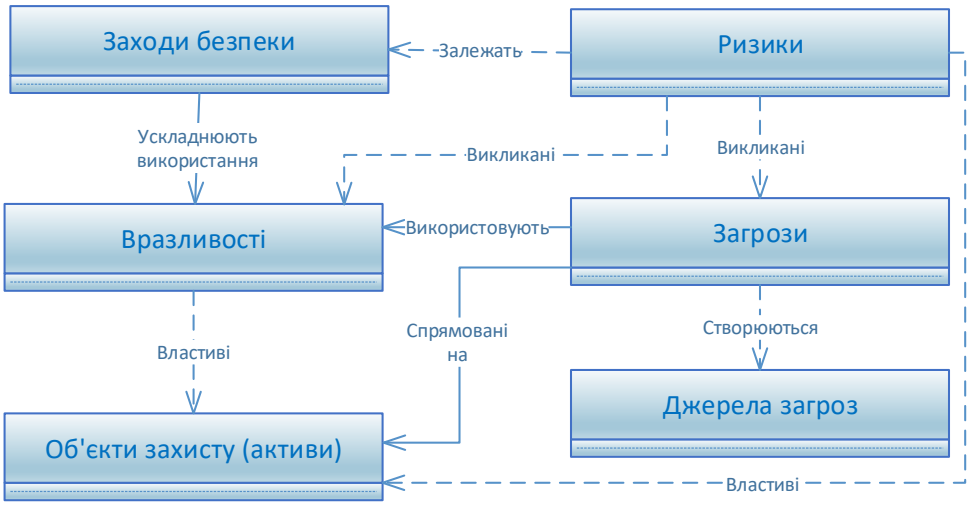


Рисунок 2.3. Поняття безпеки та їх взаємозв'язок

Виходячи з усього вищесказаного, для опису системи безпеки з повним перекриттям стосовно баз даних, конкретизуємо елементи множин об'єктів БД, що захищаються, загроз і вразливостей характерних для БД, а також заходів (засобів контролю) забезпечення безпеки. А саме, визначимо об'єкти захисту БД $o_j \in O$ з характерним для них списком загроз $t_i \in T$ та вразливостей $\gamma_\psi \in \Gamma$, завдяки яким стає можливою реалізація відповідної загрози, а також ідентифікуємо реалізовані засоби / заходи безпеки $w_k \in W$.

1. Враховуючи, що системи БД є інформаційними продуктами з двоїстою природою – двома компонентами (активами) у вигляді програмних засобів СКБД, незалежних від сфери їх застосування, структури, смислового змісту накопичуваних та оброблюваних даних та власне збережених даних, а також можливість шкідливого впливу на ці активи, доцільним є забезпечення безпеки їх обох. Для реляційних БД, як тих, що отримали найбільше поширення (про що йшлося у вступі цієї роботи), з урахуванням можливості різного ступеня деталізації цих компонентів можна виділити такі об'єкти захисту [38, 157]:

– базу даних у цілому – o_1 (усі об'єкти схем БД; під схемою розуміється опис змісту, структури та обмежень, що використовуються для створення та підтримки бази даних [158]);

– таблиці – o_2;

– уявлення (англ. views) – o_3;

– кортежі (рядки) таблиць – o_4;

– окремі поля (значення атрибутів) рядків – o_5;

– тригери – o_6;

– модулі, що постійно зберігаються – o_7 та деякі інші.

Основними найбільшими і важливими загрозами (типами загроз) безпеки баз даних, носіями яких є різні джерела загроз (більшою мірою нас цікавитимуть антропогенні (це люди або групи осіб, внаслідок дій або бездіяльності яких відбулося порушення безпеки системи, що розглядається)) [151, 159, 160], у тому числі з можливими сценаріями дій зловмисників (на прикладі СКБД Oracle), представленими на рисунку 2.4), згідно з проведеними дослідженнями [38, 42, 44, 160-163] є:

– надмірні привілеї та привілеї, що не використовуються. Для визначеності позначимо цей тип загрози як t_1;

– зловживання законними привілеями – t_2;

– ін'єкції введення – t_3;

– зловмисне програмне забезпечення – t_4;

– недостатність заходів з аудиту даних (слабкі аудиторські сліди) – t_5;

– незахищеність носіїв (резервних копій) інформації – t_6;

– експлуатація вразливих, неправильно сконфігурованих баз даних – t_7;

– некеровані конфіденційні дані – t_8;

– логічний висновок (англ. inference) – t_9;

– відмова в обслуговуванні – t_{10};

– брак знання та досвіду з питань інформаційної безпеки – t_{11} та деякі інші.

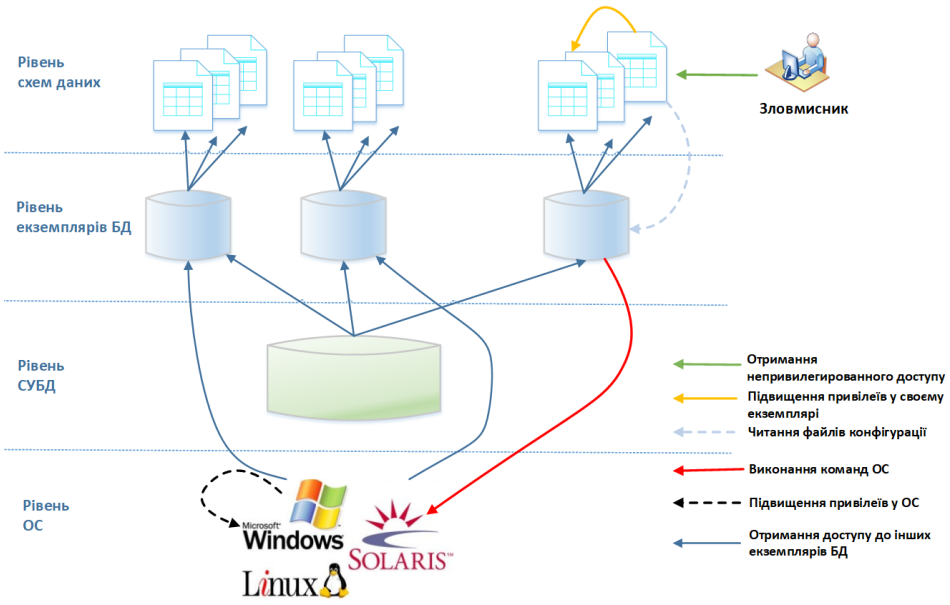


Рисунок 2.4. Схема можливих дій зловмисника

На основі проведеного аналізу існуючих таксономій вразливостей, що мають відношення до конкретного екземпляра продукту або системи (а не до основних недоліків), які можуть бути безпосередньо використані зловмисником для реалізації загроз безпеці [164], загальних слабких місць (англ. weakness) програмного та апаратного забезпечення , які можуть призвести до виникнення вразливостей [165, 166], а також деяких інших класифікацій вразливостей та недоліків безпеки активів [159, 167] було визначено перелік основних загальних слабких місць (недоліків), як деяких типів вразливостей. За основу було взято специфікацію з Common Weakness Enumeration (CWE), точніше класифікацію абстрактного уявлення Концепції дослідження (Research Concepts) CWE [168], використовувану академічними дослідниками, аналітиками вразливостей, постачальниками засобів оцінки. З урахуванням специфіки аналізованих аспектів, зумовлених характерними особливостями забезпечення безпеки, притаманними базам даних і СКБД (не враховуючи можливості реалізації загроз за допомогою вразливостей, пов'язаних з недоліками в програмному забезпеченні, архітектурі та конфігуруванні мереж та операційних систем), до їх числа увійшли такі основні слабкі місця досить високого рівня абстракції:

1) *неналежне керування привілеями*: неправильне призначення привілеїв, підвищення (ескалація) привілеїв, виконання операцій із зайвими привілеями;

2) *неналежна авторизація*: неправильне призначення дозволів для критичного ресурсу, відсутня авторизація, некоректна авторизація, розкриття конфіденційної інформації через метадані та запити даних. Не виконується або неправильно виконується перевірка авторизації, коли суб'єкт намагається отримати доступ до ресурсу або виконати певну дію;

3) *неналежна автентифікація*: слабкий пароль, застарілий пароль, обхід автентифікації, неправильна реалізація алгоритму автентифікації, невідповідний термін дії сеансу, використання пароля гешу замість пароля для автентифікації і т.д.;

4) *неконтрольоване споживання ресурсів*: належним чином не контролюється розподіл обмеженого ресурсу, тим самим дозволяючи суб'єкту впливати на кількість споживаних ресурсів, що в кінцевому підсумку призводить до їх вичерпання;

5) *зберігання конфіденційної інформації у відкритому вигляді*;

6) *недостатня стійкість шифрування*;

7) *неналежне очищення конфіденційних даних із виведеного з експлуатації пристрою*: очищення може бути відсутнім, бути недостатнім або некоректним;

8) *використання зламаного або небезпечного криптографічного алгоритму*: використання нестандартного, з недоведеною стійкістю криптографічного примітиву;

9) *використання недостатньо випадкових значень*;

10) *недостатня перевірка автентичності даних*: завантаження коду без перевірки цілісності, неправильна перевірка (відсутність перевірки) значення контрольної суми, неправильна перевірка (відсутність перевірки) криптографічного підпису;

11) *неправильна перевірка введених даних*: неправильна перевірка синтаксичної правильності вхідних даних, неправильна перевірка вказаного типу вхідних даних, неправильна перевірка узгодженості вхідних даних, неправильна

перевірка небезпечної еквівалентності вхідних даних. Вхідні дані або не перевіряються, або перевіряються неправильно – без гарантії того, що їх використання не призведе надалі до неправильної та небезпечної обробки даних;

12) *використання забороненого коду*: використовуються функції, бібліотеки або сторонні компоненти, які були заборонені розробником або замовником;

13) *вбудований зловмисний код*: троянський кінь (англ. Trojan horse), люк (англ. trapdoor), time bomb, logic bomb, шпигунське ПЗ (англ. spyware);

14) *порушення принципів безпечного проектування*: непотрібна складність у механізмі захисту (використовується більш складний механізм, ніж необхідно); опора на єдиний фактор при ухваленні рішення про безпеку; недостатня компартменталізація (англ. compartmentalization) – недостатньо поділяються функціональність або процеси, що вимагають різних рівнів привілеїв, прав чи дозволів; не передбачено перевірку доступу до об'єкта, що захищається, яка виконується щоразу при зверненні суб'єкта до цього об'єкта; недостатня психологічна прийнятність (складність та незручність використання механізму захисту часто спонукає користувачів не зловмисників відключати або обходити його випадково чи навмисно); опора на безпеку через невідомість (використовується механізм захисту, сила якого значною мірою залежить від його невідомості); недосконаліть механізму підтримки цілісності даних;

15) *неправильне надання зазначеної функціональності*: код не працює відповідно до опублікованих специфікацій, що може призвести до неправильного використання;

16) *прихована функціональність*: є функції, які не задокументовані, не є частиною специфікації та недоступні через інтерфейс чи послідовність команд (у тому числі, наприклад, функції, що реалізують шкідливий код);

17) *неповна документація*: немає описів всіх відповідних елементів продукту, таких як його використання, структура, інтерфейси, проектування, реалізація, конфігурація, експлуатація і т. д., що, природно, ускладнює обслуговування, опосередковано впливаючи на безпеку через недостатню обізнаність, ускладнюючи пошук та / або виправлення вразливостей або забираючи багато часу, що також може спростити впровадження вразливостей;

18) *вада конфігурації*: недотримання вимог безпеки при інсталяції та конфігурації БД (встановлені адміністративні, допоміжні, навчальні облікові записи, що прописуються в БД за замовчуванням без належного їх аналізу та зміни паролів за замовчанням, не встановлені обмеження на довжину та складність паролів, не заблоковані облікові записи, що не використовуються, не встановлені критичні оновлення, неналежним чином налаштована система аудиту подій тощо).

Для визначеності позначимо їх відповідно як $\gamma_1,...,\gamma_{18}$.

Після ідентифікації загроз і вразливостей, а також оцінки можливості їх зв'язування необхідно визначити ймовірності небажаного інциденту (реалізації загрози) для відповідних пар «загроза-уразливість» (t_i,γ_ψ), де $i=\overline{1,11};\ \psi=\overline{1,18}$, як добуток ймовірності виникнення відповідної загрози P_{t_i} на ймовірність відповідної вразливості P_{γ_ψ} : $P_l = P_{t_i}\cdot P_{\gamma_\psi}$.

Таким чином, модель Клементса–Гофмана була розширена до 6-мірного кортежу за рахунок включення множини вразливостей об'єктів як окремо об'єктивно існуючої категорії. Це дозволяє більш адекватно оцінювати можливість небажаного інциденту і захищеність бази даних в цілому. Крім цього в результаті вдосконалення моделі Клементса–Гофмана з урахуванням двоїстої природи системи реляційних БД та різного ступеня деталізації її компонентів були визначені:

- основні об'єкти захисту;
- перелік основних загальних слабких місць, як деяких типів вразливостей;
- основні значущі загрози для безпеки баз даних;
- інтегральний показник захищеності БД.

2.4. Метод оцінювання основних компонентів бар'єрів безпеки та захищеності бази даних

Неважко бачити, що при відомих значеннях ймовірності небажаного інциденту (реалізації загрози) P_l, величини шкоди (збитку) L_l (при вдалому

здійсненні загрози щодо об'єкта, що захищається), ступеня опірності відповідного засобу захисту R_l можна оцінити захищеність БД, скориставшись виразом (2.3).

Однак одержання точних значень P_{t_i}, P_{γ_ψ}, L_l, R_l не просте завдання. Найчастіше на практиці це неможливо [154]. До того ж, перефразовуючи Заде [169], зі збільшенням складності системи аналітична точність зменшується [85]. Тому, як правило, у таких випадках доцільно вдатися до числових оцінок у деякому діапазоні величин, тим більше, що кожному кількісному діапазону можна зіставити певну якісну шкалу, з якою за певних потреб працювати суттєво простіше. Придатною структурою для вираження таких величин, як зазначалося вище, може бути лінгвістична змінна. З цих причин, у першу чергу, відповідно до введених змін моделі перевизначимо бар'єри безпеки B, кожен з яких ($b_l \in B$) представимо у вигляді складової лінгвістичної змінної, компонентами якої є лінгвістичні змінні: ймовірність виникнення загрози (P_t), ймовірність використання вразливості (P_γ), величина шкоди / збитку (L) при вдалому здійсненні загрози щодо об'єкта, що захищається, ступінь опірності засобу захисту (R), що характеризується ймовірністю його подолання. При цьому, знову ж таки зауважуємо, що дані компоненти оцінюються в контексті специфічного бар'єру, який вони формують (індекси у $P_l = f(P_{t_i}, P_{\gamma_\psi}), L_l, R_l$ такі ж як індекс бар'єру, а не такі як у компонентів бар'єру $b_l = (t_i, \gamma_\psi, o_j, w_k)$ – загрози, вразливості, об'єкта та засобу захисту в базовій системі захисту).

Формалізацію відповідних компонентів почнемо з ймовірності виникнення загрози P_t, яка може бути представлена у вигляді лінгвістичної змінної:

$$\langle x, T(x), X, G, M \rangle, \tag{2.5}$$

де x – назва лінгвістичної змінної (у нашому випадку – це ймовірність виникнення загрози P_t); $T(x)$ – множина значень лінгвістичної змінної (терм-множина), що являють собою найменування нечітких змінних (α_ε, де $\varepsilon = 1, 2, ...$ ($\varepsilon \in \mathbb{N}^*_{<n}$), n – максимальна кількість нечітких змінних), областю визначення

кожної з яких є множина X – універсальна множина або універсум (в даному випадку це числові значення ймовірності виникнення загрози); G – деяка синтаксична процедура, що дозволяє оперувати елементами терм-множини T, зокрема, генерувати нові терми (значення); M – семантична процедура, що дозволяє перетворити кожне нове значення лінгвістичної змінної, одержуване за допомогою процедури G, на нечітку змінну, тобто сформувати відповідну нечітку множину. У даному випадку можна обмежитися припущенням про тривіальний характер G і M, тобто ніяких логічних зв'язок і модифікаторів використовуватися не буде.

Імовірність виникнення тієї чи іншої загрози інформації визначається експертним шляхом на підставі показника, що характеризує наскільки ймовірним є виникнення загрози безпеці в системі, що розглядається, з урахуванням особливостей її структури та функціонування. На практиці для обчислення ризику найчастіше використовується не математична ймовірність загрози, а приблизна частота її реалізації за певний період. Щоб не було плутанини, замість математичного терміну *probability* в стандартах навмисно використовується поняття *likelihood*, яке також перекладається як «ймовірність». При цьому експерти не визначають функцію правдоподібності у статистичному значенні. Натомість вони на основі наявних даних, досвіду та експертних суджень визначають бал (рейтинг, оцінку, позначку – англ. *score*) ймовірності [170].

Аналіз різних авторитетних джерел з проблем управління інформаційними ризиками [154, 170-175] показав, що для оцінки ймовірності виникнення загрози P_t достатньо запровадити три вербальні градації з відповідними приблизними кількісними оцінками, без яких будь-яка якісна шкала позбавляється сенсу:

– низька ймовірність (Н). Виникнення цієї загрози малоймовірне. Не існує інцидентів, статистики, мотивів, які б вказували на те, що це може статися. Очікувана частота загрози не перевищує 1 раз на 5 років;

– середня ймовірність (С). Існують передумови до появи загрози (зафіксовані випадки, у минулому відбувалися інциденти), існує статистика або є інша інформація, що вказує на можливість виникнення цієї загрози, зловмисник має

мотивацію для реалізації відповідних дій. Очікувана частота появи цієї небезпеки – приблизно раз на рік;

– висока ймовірність (В). Є об'єктивні передумови виникнення загрози. Існують інциденти, статистика або інша інформація, яка вказує на те, що загроза швидше за все здійсниться, зловмисник має мотиви для реалізації відповідних дій. Очікувана частота появи загрози – у середньому 1 раз на чотири місяці чи частіше.

Такої трирівневої шкали зазвичай достатньо для початкової високорівневої оцінки загроз. Надалі її можна розширити, додавши ще кілька проміжних рівнів. При цьому слід зазначити, що оцінки очікуваної частоти виникнення загрози від рівня до рівня за якісною шкалою різняться в рази, тому малоймовірно, щоб компетентні експерти сильно помилялися б у своїх оцінках.

З іншого боку, частотну оцінку наявної величини можна перетворити на числовий еквівалент ймовірності виникнення загрози, що відповідає деякому діапазону значень. Під терміном «імовірність» у даному випадку розуміється так звана суб'єктивна ймовірність – міра впевненості деякої людини чи групи людей (агентів) у цьому, що це подія насправді матиме місце [172, 176].

Виходячи з узагальнення проаналізованих джерел [151, 172, 174], будемо вважати, що в числовому еквіваленті ймовірність виникнення такої загрози на відповідному рівні може перебувати у відповідному діапазоні:

– для рівня Н – $P_t = [0, 0.2]$;

– для рівня С – $P_t = [0.2, 0.6]$;

– для рівня В – $P_t = [0.6, 1]$.

Тоді скориставшись застосовуваними при оцінці ризиків інформаційної безпеки відомими якісними шкалами [154, 170-172, 175], зокрема, трирівневою якісною шкалою, визначимо найменування нечітких змінних – множина значень терм-множини $T(x) = T(P_t) =$ {«низька ймовірність», «середня ймовірність», «висока ймовірність»}={«Н», «С», «В»}, тобто α_1 =«Н», α_2 = «С», α_3 =«В».

Як відомо, коли мова йде про нечітку змінну α, завжди мається на увазі деяка нечітка множина $A = \{\mu_A(x) / x\}$, яка визначає її можливі значення, де

$\mu_A(x)$ – *функція належності* ($\mu_A(x) \in [0,1]$; $\mu_A(x): X \to [0,1]$), яка вказує ступінь належності елемента x до нечіткої множини A.

Найбільшого поширення при побудові функцій належності нечітких множин набули прямі та опосередковані методи [177, 178]. Через те, що $x \in X$ можуть бути виміряні в кількісній шкалі, скористаємося прямим способом, коли експерт чи група експертів задають для кожного $x \in X$ значення функції належності $\mu_A(x)$. При цьому, як зазначається в роботі [177], теорія нечітких множин при використанні прямих методів побудови функції не вимагає абсолютно точного її завдання. Дуже часто буває достатньо зафіксувати лише найбільш характерні значення та вид (тип) функції $\mu_A(x)$. А сама функція належності може бути визначена [179]: графічно (графік, діаграма); аналітично (формули); у вигляді таблиці, суми чи інтеграла, вектору ступенів належності. Як показує досвід зручно використовувати функції належності, які допускають аналітичне уявлення у вигляді деякої простої математичної функції [177]. На підставі аналізу основних функцій належності, що використовуються для представлення таких властивостей нечітких множин, що характеризуються невизначеністю типів, як: «невелике значення», «незначна величина»; «розташований в інтервалі», «приблизно дорівнює»; «велике значення», «значна величина», для аналізованих нечітких змінних «Н», «С», «В» були обрані трапецієподібна, лінійна Z- і лінійна S-подібні функції. Кожна з цих функцій може бути представлена таким чином:

– лінійна Z-подібна функція належності нечіткої множини $A_{\text{Н}} = \{\mu_{\text{Н}}(x) / x\}$, відповідної нечіткої змінної «Н» для лінгвістичної змінної – ймовірність виникнення загрози (P_t):

$$\mu_{\text{Н}}(x; a, b) = \begin{cases} 1, & x \le a, \\ \dfrac{b - x}{b - a}, & a < x < b, \\ 0, & b \le x, \end{cases} \qquad (2.6)$$

де a, b – упорядковані ставленням $a \le b$ числові параметри, що приймають у розглянутому випадку значення $a = 0,2$; $b = 0.25$;

– трапецієподібна функція належності нечіткої множини $A_{\text{С}} = \{\mu_{\text{С}}(x) / x\}$, відповідної нечіткої змінної «С» для лінгвістичної змінної P_t:

$$\mu_{\text{С}}(x;a,b,c,d) = \begin{cases} 0, & x \le a, \\ \dfrac{x-a}{b-a}, & a \le x \le b, \\ 1, & b \le x \le c, \\ \dfrac{d-x}{d-c}, & c \le x \le d, \\ 0, & d \le x, \end{cases} \tag{2.7}$$

де a, b, c, d – упорядковані ставленням $a \le b \le c \le d$, числові параметри, що приймають в даному випадку значення $a = 0,2$; $b = 0.25$; $c = 0.55$; $d = 0.6$;

– лінійна S-подібна функція належності нечіткої множини $A_{\text{В}} = \{\mu_{\text{В}}(x) / x\}$, відповідної нечіткої змінної «В» для лінгвістичної змінної P_t:

$$\mu_B(x;c,d) = \begin{cases} 0, & x \le c, \\ \dfrac{x-c}{d-c}, & c < x < d, \\ 1, & d \le x, \end{cases} \tag{2.8}$$

де c, d – числові параметри ($c \le d$).

На рисунку 2.5 представлені всі три графіки функцій належності нечітких змінних, що використовуються для завдання лінгвістичної змінної – ймовірність виникнення загрози P_t.

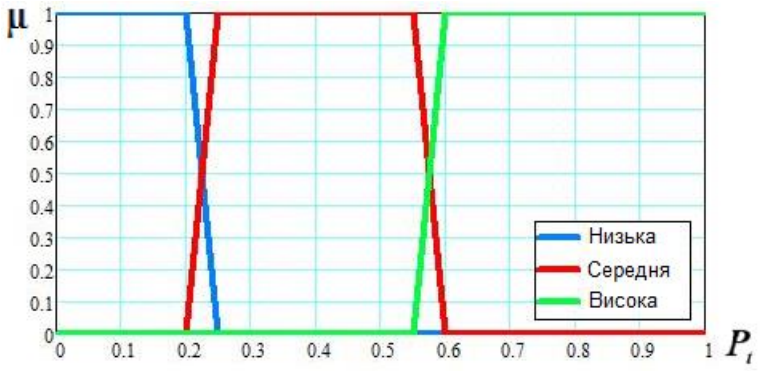


Рисунок 2.5. Графіки функції належності нечітких множин $A_{\text{Н}}$, $A_{\text{С}}$, $A_{\text{В}}$

Експерт, на підставі апріорних знань, надає лінгвістичні значення, що є найменуваннями нечітких змінних, для кожної ймовірності виникнення загрози

P_{t_i}, як компоненти відповідного специфічного бар'єру b_l. В даному випадку ці значення можуть представлятися вербально, як: «низька ймовірність», «середня ймовірність», «висока ймовірність» (або «Н», «С», «В»). При цьому, оскільки з кожним таким значенням пов'язується відповідна функція належності з відповідними приблизними кількісними оцінками, то в принципі для кожної загрози $t_i \in T$ можна визначити з обмеженим ступенем точності чисельне значення цієї ймовірності P_{t_i}, наприклад, як *модальне значення нечіткої множини*. Якщо ядро нечіткої множини містить більше одного елемента, то для такої множини модальне значення визначається як середнє значення елементів ядра. *Ядро нечіткої множини* A являє собою чітку підмножину області визначення X, що містить всі елементи, що належать множині A зі ступенем, що дорівнює 1 [179]: $C(A) = core(A) = \{x : \mu_A(x) = 1,\ x \in X\}$.

Далі, скориставшись вищевикладеним підходом, представимо у вигляді відповідної лінгвістичної змінної ймовірність використання вразливості – P_γ (ймовірність того, що у разі реалізації загрози щодо активу ця загроза буде реалізована успішно з використанням даної вразливості). Вразливості, так само як і загрози, можуть бути оцінені за якісною трирівневою шкалою. Для оцінки P_γ введемо три вербальні градації з відповідними приблизними кількісними оцінками: *висока* (В). Вразливість легко використовувати і існує слабкий захист або захист взагалі відсутній. Імовірність використання вразливості (імовірність успішної реалізації загрози за рахунок цієї вразливості) перебуває у діапазоні [0.7, 1]; *середня* (С). Вразливість може бути використана, але існує певний захист. Імовірність використання вразливості перебуває у діапазоні [0.3, 0.7]; *низька* (Н). Вразливість складно використовувати, і є гарний захист. Імовірність використання вразливості перебуває у діапазоні [0, 0.3].

Так само як і з загрозами, для первісної високорівневої оцінки вразливостей може вистачити такої трирівневої шкали. Надалі для більш детальної оцінки її можна розширити. Тоді, скориставшись введеними позначеннями, визначимо

найменування нечітких змінних (β_ε, де $\varepsilon \in \mathbb{N}^*_{<n}$) – множина значень терм-множини T_γ лінгвістичної змінної P_γ: T_γ={«висока вразливість», «середня вразливість», «низька вразливість»)={«В», «С», «Н»}, тобто β_1=«В», β_2= «С», β_3=«Н». Областю визначення кожної з нечітких змінних є множина числових значень ($X \in [0,1]$) імовірності використання вразливості. У цьому випадку теж можна обмежитися припущенням про тривіальний характер G_γ і M_γ (без логічних зв'язок і модифікаторів).

На підставі аналізу основних функцій належності, подібно до наведеного вище, для нечітких змінних β_1=«В», β_2= «С», β_3=«Н» були обрані трапецієподібна, лінійна Z- і лінійна S-подібні функції. Кожна з цих функцій може бути представлена як:

– лінійна Z-подібна функція належності нечіткої множини $A^\nu_{\text{Н}} = \{\mu^\nu_{\text{Н}}(x) / x\}$, відповідної нечіткої змінної «Н» для лінгвістичної змінної P_γ:

$$\mu^\nu_{\text{Н}}(x;a,b) = \begin{cases} 1, & x \le a, \\ \dfrac{b-x}{b-a}, & a < x < b, \\ 0, & b \le x; \end{cases} \tag{2.9}$$

– трапецієподібна функція належності нечіткої множини $A^\nu_{\text{С}} = \{\mu^\nu_{\text{С}}(x) / x\}$, відповідної нечіткої змінної «С» для лінгвістичної змінної P_γ:

$$\mu^\nu_{\text{С}}(x;a,b,c,d) = \begin{cases} 0, & x \le a, \\ \dfrac{x-a}{b-a}, & a \le x \le b, \\ 1, & b \le x \le c, \\ \dfrac{d-x}{d-c}, & c \le x \le d, \\ 0, & d \le x; \end{cases} \tag{2.10}$$

– лінійна S-подібна функція належності нечіткої множини $A^\nu_{\text{В}} = \{\mu^\nu_{\text{В}}(x) / x\}$, відповідної нечіткої змінної «В» для лінгвістичної змінної P_γ:

$$\mu_B^v(x;c,d)=\begin{cases}0, & x\le c,\\ \dfrac{x-c}{d-c}, & c<x<d,\\ 1, & d\le x.\end{cases} \tag{2.11}$$

На рисунку 2.6 представлені три графіки функцій належності нечітких змінних, які використовуються для завдання лінгвістичної змінної – ймовірність використання вразливості P_γ.

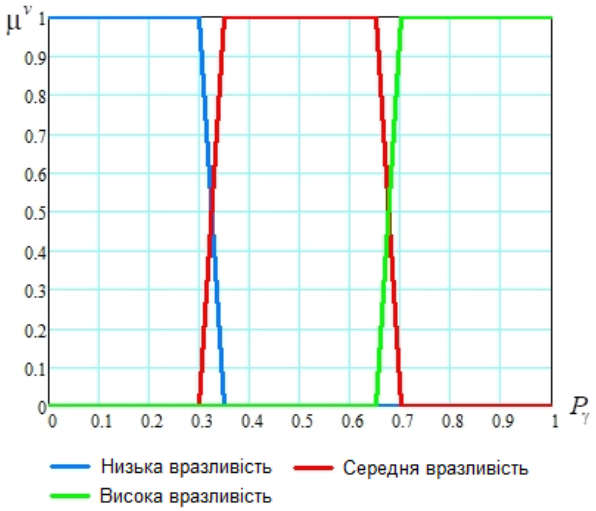


Рисунок 2.6. Графіки функції належності нечітких множин $A_{\text{Н}}^v$, $A_{\text{С}}^v$, $A_{\text{В}}^v$

Експерт, на підставі апріорних знань, присвоює лінгвістичні значення, що є найменуваннями нечітких змінних, для кожної імовірності використання вразливості P_γ як компоненти відповідного бар'єру b_l, завдяки якій стає можливою реалізація відповідної загрози t_i. Ці значення представляються вербально, як: «Н», «С», «В». Так як з кожним таким значенням зв'язується відповідна функція належності з відповідними приблизними кількісними оцінками, то для кожної вразливості γ_ψ можна обчислити з обмеженим ступенем точності чисельне значення цієї імовірності P_{γ_ψ}, наприклад, як модальне значення відповідної нечіткої множини.

За аналогією можна визначити ступінь опірності засобів захисту (званих у літературі [127, 149, 153, 154, 175, 180, 181] також як механізми, заходи, засоби контролю (англ. controls), до яких відноситься будь-який процес, політика,

пристрій, практика, що встановилася, чи інші дії, які змінюють ризик [153]), що характеризується ймовірністю їх подолання ($P_l^{ov} = 1 - R_l$). Відповідні рівні контролю (ступінь опірності) можуть бути визначені таким чином:

– В – високий ступінь опірності засобу (заходу, механізму) захисту (високий рівень контролю). Малоймовірно, що такий механізм вдасться подолати. Імовірність подолання (обходу) такого механізму знаходиться в діапазоні – $P_l^{ov} \in [0,\ 0.4]$;

– С – середній ступінь опірності засобу захисту. Такий засіб (захід) забезпечує певний захист, проте є можливість його подолати, витративши певні зусилля. Імовірність подолання відповідного заходу захисту перебуває у діапазоні [0.4, 0.8];

– Н – низький ступінь опірності засобу захисту. Такий засіб досить просто подолати. Імовірність подолання відповідного заходу захисту перебуває у діапазоні [0.8, 1].

Тоді, скориставшись цією шкалою, визначимо найменування нечітких змінних (δ_ε, де $\varepsilon \in \mathbb{N}^*_{<n}$) – множина значень терм-множини T_R лінгвістичної змінної R: T_R={«високий ступінь опірності», «середній ступінь опірності», «низький ступінь опірності»}={«В», «С», «Н»}, тобто δ_1=«В», δ_2= «С», δ_3=«Н». Областю визначення кожної з нечітких змінних є множина числових значень ($X \in [0,1]$) імовірності подолання засобів захисту. У цьому випадку також обмежимося припущенням про тривіальний характер G_R і M_R.

Подібно наведеному вище підходу, для нечітких змінних, що розглядаються, δ_1=«В», δ_2= «С», δ_3=«Н» (з якими зв'язуються відповідні нечіткі множини, що визначають їх можливі значення: $A_{\text{Н}}^{ov} = \{\mu_{\text{Н}}^{ov}(x) / x\}$, $A_{\text{С}}^{ov} = \{\mu_{\text{С}}^{ov}(x) / x\}$, $A_{\text{В}}^{ov} = \{\mu_{\text{В}}^{ov}(x) / x\}$) були обрані трапецієподібна, лінійна Z- і лінійна S- образні функції належності ($\mu_{\text{Н}}^{ov}(x)$, $\mu_{\text{С}}^{ov}(x)$, $\mu_{\text{В}}^{ov}(x)$).

На рисунку 2.7 представлені графіки функцій належності нечітких змінних, що використовуються для визначення лінгвістичної змінної – ступінь опірності

засоби захисту R ($R=1-P^{ov}$; у деяких джерелах [174] P^{ov} називають зворотної силою контролю (англ. reverse of the control strength)).

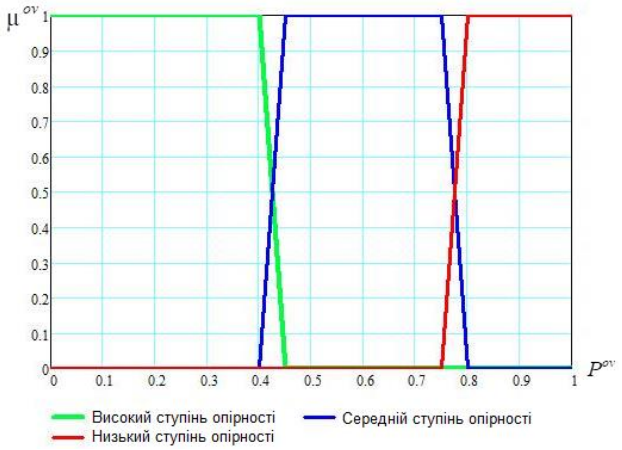


Рисунок 2.7. Графіки функції належності нечітких множин $A^{ov}_{\text{Н}}$, $A^{ov}_{\text{С}}$, $A^{ov}_{\text{В}}$

Експерт, на підставі апріорних знань про засоби захисту (захисні заходи), що ускладнюють використання відповідної вразливості γ_{ψ}, завдяки якій стає можливою реалізація відповідної загрози t_i, присвоює лінгвістичні значення «високий ступінь опірності», «середній ступінь опірності», «низький ступінь опірності» або «В», «С», «Н» для кожної R_l як компоненти відповідного бар'єру b_l. Зважаючи на те, що з кожним таким значенням пов'язується відповідна функція належності з відповідними приблизними кількісними оцінками, то для кожного засобу забезпечення безпеки $w_k \in W$ бар'єра b_l можна визначити чисельне значення як P_l^{ov}, так і $R_l = 1 - P_l^{ov}$. Знову ж таки, як модальне значення відповідної нечіткої множини.

Збитки (як втрати), завдані внаслідок інцидентів безпеки, пов'язуються з цільовою функцією системи – одним із відповідних показників, таким як втрачена вигода, втрата конкурентних переваг, погіршення репутації організації, заподіяння шкоди інтересам третьої сторони, фінансові втрати, пов'язані з відновленням ресурсів, дезорганізація діяльності у зв'язку з недоступністю даних тощо. Для різних організацій важливість кожного з них може мати істотно різне значення. З економічної точки зору збитки активам зручно представляти у

термінах фінансових втрат. Однак на практиці отримання точних кількісних значень шкоди часто утруднене або взагалі неможливе [152]. Проте більшість втрат, що не піддаються кількісному опису, можна представити в чисельному вигляді шляхом використання емпіричної шкали рівня збитку – якісної шкали вимірювання, розділеної на області (ранги), відповідні різним ступеням задоволення цих вимог, наприклад, п'ятибальної шкали: від 1 до 5. Кожному з таких рівнів (рангів) можна зіставити значення терм-множини T_L (T_L={«Дуже низький», «Низький», «Середній», «Високий», «Дуже високий»)={«VL», «L», «M», «H», «VH»}) лінгвістичної змінної – величина шкоди (збитку) L. Областю визначення кожної з нечітких змінних є множина числових значень величини збитку / рівня шкоди (у балах) $X \in (0,6)$. У цьому випадку також можна обмежитися припущенням про тривіальний характер G_L і M_L.

Для нечітких змінних, що розглядаються, ρ_1=«VH», ρ_2=«H», ρ_3=«M», ρ_4=«L», ρ_5=«VL» (з якими зв'язуються відповідні нечіткі множини, що визначають їх можливі значення: $A^L_{\text{VH}} = \{\mu^L_{\text{VH}}(x)/x\}$, $A^L_{\text{H}} = \{\mu^L_{\text{H}}(x)/x\}$, $A^L_{\text{M}} = \{\mu^L_{\text{M}}(x)/x\}$, $A^L_{\text{L}} = \{\mu^L_{\text{L}}(x)/x\}$, $A^L_{\text{VL}} = \{\mu^L_{\text{VL}}(x)/x\}$) були обрані трикутні, лінійна Z- та лінійна S-подібні функції належності ($\mu^L_{\text{VH}}(x)$, $\mu^L_{\text{H}}(x)$, $\mu^L_{\text{M}}(x)$, $\mu^L_{\text{L}}(x)$, $\mu^L_{\text{VL}}(x)$):

– $$\mu^L_{\text{VL}}(x;a,b) = \begin{cases} 1, & x \le a, \\ \dfrac{b-x}{b-a}, & a < x < b, \\ 0, & b \le x, \end{cases} \qquad \text{где } a=1;\ b=2; \tag{2.12}$$

– $$\mu^L_{\text{H}}(x;a,b,c,d), \mu^L_{\text{M}}(x;a,b,c,d), \mu^L_{\text{L}}(x;a,b,c,d) = \begin{cases} 0, & x \le a, \\ \dfrac{x-a}{b-a}, & a \le x \le b, \\ \dfrac{c-x}{c-b}, & b \le x \le c, \\ 0, & c \le x, \end{cases} \tag{2.13}$$

де: для μ^L_{H} a=1, b=2, c=3; для μ^L_{M} a=2, b=3, c=4; для μ^L_{L} a=3, b=4, c=5;

– $\mu_{\text{VH}}^{L}(x;c,d)=\begin{cases}0, & x\le c,\\ \dfrac{x-c}{d-c}, & c<x<d,\\ 1, & d\le x,\end{cases}$ где $c=4;\ d=5.$ (2.14)

На рисунку 2.8 представлені графіки функцій належності нечітких змінних, що використовуються для завдання лінгвістичної змінної – величина шкоди L.

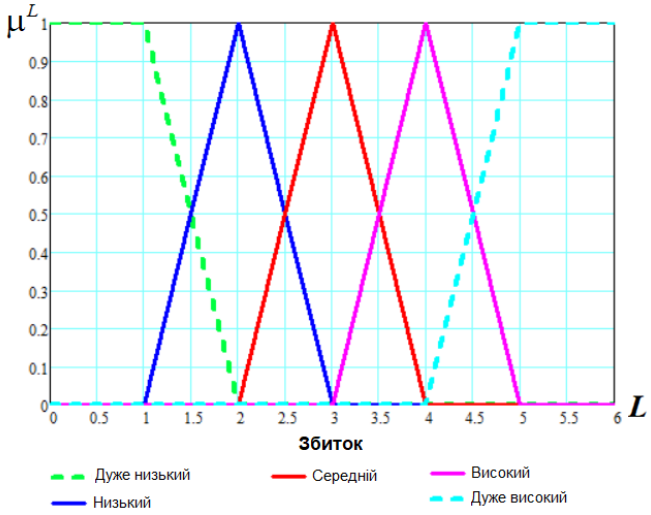


Рисунок 2.8. Графіки функції належності нечітких множин $A_{\text{VH}}^{L}, A_{\text{H}}^{L}, A_{\text{M}}^{L}, A_{\text{L}}^{L}, A_{\text{VL}}^{L}$

У таблиці 2.1 представлена оцінка величини збитків у п'ятибальній шкалі та її семантична характеристика. Щоб оцінка цінності активів мала економічний сенс, якісну шкалу оцінки збитків доцільно співвіднести з розміром прямих фінансових втрат. Однак встановлення такої відповідності вимагає додаткових досліджень у кожному конкретному випадку і залежить від багатьох факторів для систем, що розглядаються.

Таблиця 2.1 – Оцінка величини шкоди та її семантична характеристика

Рівень збитків	**Значення терм-множини T_L**	**Семантична характеристика значення показника величини шкоди (збитку)**
1	Дуже низький	Можна знехтувати збитками.
2	Низький	Збитки легко усунути, витрати на ліквідацію наслідків реалізації загрози невеликі.
3	Середній	Ліквідація наслідків реалізації загрози не пов'язана з великими витратами.
4	Високий	Ліквідація наслідків реалізації загрози пов'язана із значними фінансовими втратами.
5	Дуже високий	Організація припиняє існування.

Можливі незалежні шкали (приклади) оцінки прямих фінансових втрат та їх відносних значень ($r_{fl}^{rel} = z_{fl} / z_{fl}^{per}$, де z_{fl} – прямі фінансові втрати, z_{fl}^{per} – допустимі прямі фінансові втрати) наведено у таблиці 2.2. Все залежить від завдань, що вирішуються організацією, галузі, характеру та масштабів її діяльності, форми власності, вартості активів, тяжкості наслідків порушення їхньої безпеки та низки інших факторів.

Таблиця 2.2 – Шкали оцінки фінансових збитків

Рівень збитків	**Значення терм-множини T_L**	**Фінансові втрати (z_{fl})**	**Відносні значення фінансових втрат (r_{fl}^{rel})**
1	Дуже низький	менше 100 $	≤ 0.1
2	Низький	(100-1000) $	(0.1,0.3]
3	Середній	(1000-10 000) $	(0.3,0.6]
4	Високий	(10 000-100 000) $	(0.6,0.9]
5	Дуже високий	понад 100 000 $	> 0.9

Таким чином, при розробці методу оцінювання захищеності бази даних на підставі узагальнення рекомендацій експертів було визначено: кількість рівнів для аналізованих лінгвістичних змінних з відповідними для них діапазонами, а також функції належності та можливий варіант оцінки чисельного значення відповідної ймовірності або шкоди. Маючи в своєму розпорядженні відповідні дані, скориставшись виразом (2.3), можна визначити величину захищеності аналізованої бази даних.

Слід зазначити, що запропонованому методу, на відміну від деяких відомих, властива певна гнучкість. Це виявляється у здатності адаптуватися до нових умов функціонування і враховувати нові актуальні загрози, вразливості, заходи безпеки, які можуть об'єднуватися в деякі спільні групи. У тому числі є можливість вибору кількості рівнів відповідних лінгвістичних змінних. При цьому використання введеного інтегрального показника безпеки дозволяє кількісно оцінювати величину захищеності досліджуваної бази даних.

2.5. Висновки за розділом

1. Проаналізувавши та узагальнивши різні підходи та досягнення в галузі оцінки безпеки інформаційних систем в цілому та баз даних, зокрема, було

прийнято рішення про доцільність застосування моделі Клементса–Гофмана, що спирається на теорію графів, нечітких множин, ймовірностей, як науково-методологічної основи побудови системи захисту та оцінки безпеки БД.

2. Для більш адекватного оцінювання ймовірності небажаного інциденту (реалізації загрози) та захищеності БД у цілому модель Клементса–Гофмана була розширена до 6-мірного кортежу (шістки). Розширення було здійснено за рахунок доповнення моделі множиною вразливостей (слабких місць) об'єктів як окремо об'єктивно існуючої категорії. Це означає, що ймовірність небажаного інциденту (реалізації загрози) розглядається як двофакторна складова, в якій один із факторів відображає мотиваційну складову виникнення загрози, а другий враховує існуючі вразливості. Крім цього, у процесі розробки вдосконаленої моделі були конкретизовані деякі значущі її компоненти. А саме:

– виявлено основні значущі антропогенні загрози безпеці баз даних;

– визначено основні об'єкти захисту реляційних БД (з урахуванням двоїстої природи системи БД та різного ступеня деталізації її компонентів);

– визначено перелік основних загальних слабких місць як деяких типів вразливостей БД.

В якості інтегрального показника безпеки бази даних було визначено величину зворотну сумарному залишковому ризику, який по суті є мірою незахищеності активу. Це дозволило кількісно оцінювати безпеку баз даних. Складові компоненти, що визначають залишковий ризик та характеризують стійкість деякого бар'єру безпеки, представляються у вигляді певних лінгвістичних змінних.

3. Розроблений метод оцінювання основних компонент бар'єрів безпеки та захищеності бази даних в цілому є результатом синтезу вдосконаленої моделі Клементса–Гофмана, інтегрального показника безпеки, положень теорії нечітких множин та ризику. Даний метод, на відміну від відомих, маючи певну гнучкість, дозволяє досить просто, комплексно і кількісно оцінювати безпеку БД з різними моделями даних.

Основні питання, розглянуті у розділі, опубліковані у роботах [124, 151, 160, 182].

3. РОЗРОБКА МЕТОДІВ МАСКУВАННЯ ДАНИХ

3.1. Метод маскування даних на основі обчислення операцій за модулем

Сьогодні, як зазначалося у розділі 1 (п. 1.2), незважаючи на проведені дослідження та наявні рішення, існує потреба знаходження деякого нового підходу, який дозволяв би певною мірою зменшити ймовірність реалізації загрози логічного висновку (англ. inference). Тому, проаналізувавши можливості відомих методів [50-55, 183, 184], а також кращі практики приховування інформації від провідних вендорів на ринку маскування [185], спочатку було вирішено скористатися математичними перетвореннями на основі обчислення операцій за модулем не тільки для числових значень, подібно до методу MOBAT [52, 55, 184], але і для значень такого поширеного для БД типу даних, як рядок символів. Це робилося з метою розширення охоплення даних різного типу, що зберігаються в БД, для яких були потрібні відповідні перетворення, що ускладнюють реалізацію зловмисником загрози логічного висновку.

Суть методу полягає в наступному. З поля j-го атрибута i-го кортежу (рядка) деякого відношення (таблиці) R БД витягується елемент даних типу "рядок символів". Наприклад, для певності нехай це буде `'Abc123-Ю'`. Ці символи рядка спочатку перетворюються на шістнадцятковий рядок (подібне конвертування символів робиться й в методі шифрування із збереженням формату [59]). Для прикладу це `416263313233 2DD0AE`. Потім цей шістнадцятковий рядок наводиться до числового коду в десятковій системі числення (в даному випадку це `120612792912120 8914094`), над яким потім здійснюється перетворення, подібне до техніки MOBAT [53-55]:

$$R'_{ij} = R_{ij} - ((K_3^i \bmod K_1^R) \bmod K_2^j) + K_2^j, \qquad (3.1)$$

де R_{ij} – вихідне значення поля j-го атрибута i-го кортежу відношення R; R'_{ij} – замасковане значення відповідного елемента даних поля; K_1^R – 128-бітне

випадкове згенероване значення (закритий ключ – ЗК), постійне для таблиці R; K_2^j – 128-бітне випадкове згенероване значення (ЗК), постійне для атрибута j таблиці R; K_3^i – значення поля i-го рядка, одного з вибраних атрибутів цілого типу таблиці R (відкритий ключ – ВК). Як ВК найкраще використовувати значення стовпця цілого ідентифікатора (ID) первинного ключа таблиці R.

Вираз (3.1) – це пряме перетворення маскування конкретного значення поля R_{ij} кортежу i заданого атрибута j деякого відношення (таблиці) R. Зворотне перетворення виконується для отримання вихідного (не замаскованого) значення відповідного елемента даних кортежу заданого атрибута таблиці R:

$$R_{ij} = R'_{ij} + ((K_3^i \bmod K_1^R) \bmod K_2^j) - K_2^j. \qquad (3.2)$$

Для виконання математичних перетворень (3.1), (3.2) над отриманими довгими числами був запропонований спосіб і відповідна його реалізація. В основу алгоритмів, пов'язаних з виконанням обчислювальних операцій із перетворення довгих чисел у різні системи числення, без яких у практичній реалізації неможливо було обійтися, була покладена схема (правило) Горнера, що дозволяє скоротити обчислення та пам'ять. При використанні схеми Горнера немає необхідності зберігати проміжні результати, у зв'язку з тим, що кожне наступне число записується на місце попереднього в попередньо створений масив [186]. Проте після обчислення операцій з модулю над конвертованими символами, у загальному випадку, результат містив як звичайні (друковані), а й управляючі символи, що утруднювало їх уявлення в читабельному вигляді. Тому результат доводилося представляти у вигляді шістнадцяткового рядка, який зручно зберігати та обробляти в полі таблиці з типом символьного рядка, або перетворити до десяткового виду, що зручно для числової обробки. Для кращого розуміння суті перетворень і ускладнень, що виникають при цьому, розглянемо кілька прикладів.

Приклад 1. Нехай у певній послідовності рядків одного зі стовпців таблиці, дані якого необхідно замаскувати, зберігається наступний рядок символів

кирилиці: `'ГРС-56'`. Для наочності, нехай це будуть поруч розташовані рядки (рядки таблиці з нумерацією, що послідовно збільшується (інкремент за ID)).

Підсумком застосування виразу (3.1) до значень (рядку символів 'ГРС-56') стовпця (визначимо його як `DATA_X`) відповідних рядків деякої таблиці є результат, наведений нижче на рисунку 3.1.

ID	DATA_X
19746	E144A5509211952AEBE3C470A25F8C0
19747	CF518B3AC5B88CB41835AE555349764
19748	8BDC0D5DE66F27BF9280344028AB827
19749	8F48799ABEC3709CE2D45656532205E
19750	506741F22A4834CDEA6DEE65EACD651
19751	E5CBFC700E29E9374A81A902FF7BD9F
19752	C70DA272B74C3303321CA6E4FCF0A20
19753	1600D8CE183AEE9AE073EA796A06275D
19754	1649CD15E479D33CADD0193E19894C1D
19755	14D66A41674FFD516D9CC884BD8A6530
19756	C39366DA0C9E841C5A7516A42A41EDF
19757	FF1C5FA372FBA9337CE685174794AC7
19758	90A9BDB60B7BFDD79602050C8E8B1E9
19759	11DA56F36CB5A2611031BAD1BE2CC97F
19760	15091E645FCB319C8B7BD1ED0E660FFB
19761	F4FF490B43E6D894FBF15FDE133C93B
19762	E869C89417388E8B003299A376D326A
19763	A3EB56EAB14C77790ED7620F8EAFA14
19764	A2CA6EEA8F68627BD3690F743E081B3
19765	CE8F48D829CC53EE2EEC5B9000A8B53

Рисунок 3.1. Результат перетворення (Приклад 1)

Де значення стовпця ID є відкритими ключами K_3^i для i-го рядка таблиці. У цьому випадку ці значення лежать в діапазоні [19746, 19765] ($K_3^i \in [19746, 19765]$).

Як видно з наведеного результату, всі значення одного і того ж рядка символів `'ГРС-56'` мають різні значення, як було задумано, щоб протистояти загрозі логічного виведення. Однак таке уявлення не завжди може підходити для даних, тип (розмірність) яких не можна змінювати. До того ж, при довгих символьних рядках мало того, що збільшується час реалізації, старші байти практично не піддаються перетворенню, крім конвертації з рядкового типу в числовий, через недостатню довжину ключа. Про що свідчить приклад 2, наведений нижче.

Приклад 2. Нехай у деяких рядках того ж стовпця `DATA_X` зберігається символьний рядок `'Фосфоритний рудник'`.

Застосування виразу (3.1) до відповідного стовпця даних призводить до наступного результату – виду перетвореного символьного рядка у різних рядках таблиці (рисунок 3.2).

ID	DATA_MASKING
20659	D0A4D0BED181D184D0BED180D0B8D182D0BDD19EC501E1077C3320E176F0F1B28A3505
20660	D0A4D0BED181D184D0BED180D0B8D182D0BDD193555D8DA87CF439DBA3A015B0ACC2C7
20661	D0A4D0BED181D184D0BED180D0B8D182D0BDD196A3BCDB859EA0ED94B53DB4421A399B
20662	D0A4D0BED181D184D0BED180D0B8D182D0BDD1957238A723EF5F6E3989C1D314106159
20663	D0A4D0BED181D184D0BED180D0B8D182D0BDD1913798672CA2D33A572138BA761E3594
20664	D0A4D0BED181D184D0BED180D0B8D182D0BDD193224CFBA0E0590C3D70A2EACD6F05BE
20665	D0A4D0BED181D184D0BED180D0B8D182D0BDD1A1F3333683E2691BA17399EC724353CD
20666	D0A4D0BED181D184D0BED180D0B8D182D0BDD193A795A0B46246692D738F4A63F4B648
20667	D0A4D0BED181D184D0BED180D0B8D182D0BDD19F44BF9C5416208F6C5E33D44F9B811C
20668	D0A4D0BED181D184D0BED180D0B8D182D0BDD190AEE542469687FCE3FF8DD1C18DEA7C
20669	D0A4D0BED181D184D0BED180D0B8D182D0BDD19D68DAD6C0BAA7FC24FAE62B2526F3D1
20670	D0A4D0BED181D184D0BED180D0B8D182D0BDD19BF2CB316D1D2908F7FDC4F142E73CD0
20671	D0A4D0BED181D184D0BED180D0B8D182D0BDD196FA6E18263F66DA86C3793598F6B2FD
20672	D0A4D0BED181D184D0BED180D0B8D182D0BDD1A0892F4B6BE930B3E3D0D7CFFEE311CA
20673	D0A4D0BED181D184D0BED180D0B8D182D0BDD19EBF7F8AAC53585510F40366E236CF5A
20674	D0A4D0BED181D184D0BED180D0B8D182D0BDD19C1A189D7A0A289CB6F5AC214962308A
20675	D0A4D0BED181D184D0BED180D0B8D182D0BDD1994A517765D424A97C4514798C36CE54

Рисунок 3.2. Результат перетворення (Приклад 2)

Як видно з останнього прикладу, частина перетворених даних (виділені кольором) для одного і того ж вихідного рядка однакова. Для усунення цього недоліку виникла потреба у їх перемішуванні. Як один із доцільних варіантів бачилося очевидне рішення у випадковій перестановці відповідних байт отриманого коду рядка з можливістю зворотного їх відновлення.

Як відомо, більшість криптографічних алгоритмів досі комбінує підстановки та перестановки [187, 188], про що ще помітив у своїй роботі [189] К. Шеннон, узагальнюючи накопичений до нього досвід розробки шифрів. Як виявилося, що навіть у складних шифрах як типові компоненти можна виділити прості шифри, такі як шифри заміни, перестановки або їх поєднання. Підстановка та перестановка продовжують лежати в основі сучасних симетричних алгоритмів шифрування [188]. Відомі фахівці з комп'ютерної безпеки, криптографії Н. Фергюсон, Б. Шнайєр у монографії [190], розмірковуючи про те, як має виглядати ідеальний блоковий шифр, зазначають, що це має бути випадкова перестановка варіантів відкритого тексту. Уточнюючи при цьому, що для кожного значення ключа блоковий шифр повинен бути випадковою перестановкою варіантів відкритого тексту, причому перестановки для різних варіантів ключа

повинні вибиратися незалежно один від одного. Таке прийняте рішення надалі переросло в окремий метод.

Таким чином, даний метод привів до нового більш ефективного і менш обчислювально-затратного методу – методу випадкової перестановки елементів даних поля рядка. Хоча в деяких випадках, наприклад, при невеликих розмірностях рядків або збільшенні розмірності ключа і під час розпаралелювання процесів обчислень, а також за умови можливості зміни типу (розмірності) даних, цей метод також може бути затребуваним.

3.2. Метод маскування даних поля рядка таблиці бази даних

Як відомо, перестановкою n об'єктів називається спосіб послідовного розташування n різних об'єктів з урахуванням порядку [191]. Якщо пронумерувати місця цих об'єктів зліва направо (1,2,…,n), можна сформулювати таке визначення: взаємно-однозначне відображення $p: A \to A$ кінцевого впорядкованої множини $A = \{a_1, a_2, ..., a_n\}$ з n елементів на себе називається перестановкою елементів множини A. У загальному випадку для n елементної множини A із зафіксованим порядком елементів $a_1, a_2, ..., a_n$ перестановка – це довільна послідовність довжини n із різних елементів множини A. Перестановки з n елементів множини A відрізняються один від одного лише порядком елементів, що в них входять. Перестановку p можна записати як матриці з двох рядків. Наприклад, перестановку $\pi: A \to A$ множини $A = \{a, b, c, d, e\}$ таку, що $\pi(a) = e$, $\pi(b) = d$, $\pi(c) = a$, $\pi(d) = b$, $\pi(e) = c$, можна записати наступним чином:

$$\pi = \begin{pmatrix} a & b & c & d & e \\ e & d & a & b & c \end{pmatrix}. \qquad (3.3)$$

Зазвичай природа елементів множини A несуттєва, тому без зменшення спільності можна вважати, що $A = \{1, 2, ..., n\}$ (інакше необхідно перейти до номерів елементів (a_i, де $i \in \{1, ..., n\}$)). Тоді кожну перестановку π цих елементів можна записати як матрицю з наступних двох рядків:

$$\pi = \begin{pmatrix} 1 & 2 & 3 \ldots & n \\ a_1 & a_2 & a_3 \ldots & a_n \end{pmatrix}, \tag{3.4}$$

де $\{a_1, a_2, a_3, \ldots, a_n\} = \{1,2,3,\ldots,n\}$, $\pi(i) = a_i$ для всіх $i \in \{1,\ldots,n\}$.

Число всіх перестановок π з n різних елементів дорівнює $\pi_n = n!$.

Для кожної перестановки π існує зворотна перестановка π^{-1}, яка скасовує дію π. Добуток $\pi \cdot \pi^{-1}$ дорівнює тотожній перестановці $\pi_e = \pi \cdot \pi^{-1}$.

Зворотна перестановка $a'_1, a'_2, a'_3, \ldots, a'_n$ виходить, якщо (3.4) поміняти місцями рядки матриці, а потім упорядкувати стовпці в порядку зростання по верхнім елементам:

$$\begin{pmatrix} a_1 & a_2 & a_3 \ldots & a_n \\ 1 & 2 & 3 \ldots & n \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \ldots & n \\ a'_1 & a'_2 & a'_3 \ldots & a'_n \end{pmatrix}. \tag{3.5}$$

Застосуємо викладені вище теоретичні відомості з комбінаторики до розв'язання нашого завдання – маскування конкретного значення поля R_{ij} кортежу i заданого атрибута j деякої таблиці R за рахунок випадкової перестановки елементів критичних даних поля рядка. Випадковість у цьому випадку означає рівноймовірність отримання будь-якої з $n!$ можливих перестановок множини A.

В основу запропонованого методу випадкової перестановки елементів (байт, символів) даних конкретного поля різного типу (числового, символьних рядків, великих бінарних (Binary Large Object – BLOB), довгих символьних (Character Large Object – CLOB) об'єктів) рядки таблиці було покладено сучасний варіант алгоритму тасування Фішера-Йейтса (Fisher–Yates) [192], представлений Річардом Дуршенфельдом (Richard Durstenfeld) у [193].

Основними причинами вибору алгоритму тасування Фішера-Йейтса стали такі його переваги:

– незначна кількість кроків виконуваних операцій – асимптотична обчислювальна складність сучасного варіанта алгоритму – $O(n)$, де n – число елементів множини (у аналізованому разі це кількість елементів (символів, байт) даних конкретного поля);

– при використанні високоякісного незміщеного генератора випадкових чисел алгоритм гарантує незміщений результат;

– його ефективність та простота досі витримали випробування часом [194].

Алгоритм Фішера-Йейтса використовує вибірку рівномірно розподілених випадкових чисел із різних діапазонів. Тому важливо, щоб можна було б скористатися перевагами даного алгоритму, необхідно використовувати генератори псевдовипадкових чисел (ГПВЧ), які формують саме не зміщені в якусь частину інтервалу випадкові числа. З іншого боку, як зазначалося [186], алгоритм Фішера-Йейтса не має можливості генерувати більше m різних перестановок, тобто не може створити більше перестановок, ніж число внутрішніх станів генератора. І навіть коли кількість можливих станів генератора перевищує кількість перестановок, деякі з них можуть з'являтися частіше, ніж інші. Щоб уникнути появи нерівномірності розподілу, зазвичай рекомендується, щоб число внутрішніх станів генератора випадкових чисел перевищувало число перестановок на кілька порядків, якщо це можливо. Хоча в більшості випадків насправді не потрібно отримувати всі перестановки [186]. Тому, виходячи з вищевикладеного в запропонованому методі, передбачена можливість використання залежно від ситуації (в основному це пов'язано з необхідністю виконання процедури перемішування (перестановки) елементів даних полів різного типу з мінімальними часовими витратами) різних генераторів ПСЧ, що задовольняють зазначеним вище вимогам (криптографічно стійкий ГПВЧ в даному випадку не вимагається). Нам необхідно виконати заздалегідь такі перетворення – перестановки, щоб зловмисник не зміг, використовуючи складні, а також послідовності простих логічно пов'язаних запитів, отримати дані, доступ до яких безпосередньо йому закрито, на підставі умовиводу. Тобто реалізувати загрозу inference за прийнятний для нього час. При цьому законний користувач, навпаки, міг би отримати необхідні конфіденційні (чутливі) дані досить швидко і просто. Для визначеності в методі, що розглядається, будуть використані такі ГПВЧ:

1) лінійний конгруентний генератор (ЛКГ) випадкових чисел, популяризований у роботі [195]:

$$X_{j+1} = (aX_j + c) \bmod m \tag{3.6}$$

з константами (множником $a = 1664525$ та інкрементом $c = 1013904223$), обраними Д. Е. Кнутом і Х. В. Льюїсом, де $m = 2^{32}$;

2) генератор випадкових чисел, вбудованого пакета `DBMS_RANDOM` для СКБД Oracle, а саме `DBMS_RANDOM.VALUE`, що генерує числа з плаваючою точкою з 38 знаками після коми, з можливістю завдання їх різного діапазону;

3) генератор псевдовипадкових чисел Джорджа Марсальї Xorshift [196] з періодом 2^{128}-1. У генераторі Xorshift задається деяка початкова послідовність, до якої застосовуються операції виключне АБО (XOR) і логічного зсуву. Цей ГПВЧ був обраний, виходячи з рекомендацій, наведених у [197]. У принципі, виходячи з цих рекомендацій, можна вибрати будь-який інший ГПВЧ, у тому числі наведені у тій самій роботі [197]. При цьому слід зазначити, що у програмній реалізації всі вищезгадані ГПВЧ повинні бути перевірені на відповідність випадкових чисел, що видаються, в заданому діапазоні для рівномірного закону розподілу.

3.2.1. Характеристика основних етапів процесу маскування

У запропонованому рішенні використовується універсальна схема приховування даних полів різного типу рядка кортежу деякої таблиці бази даних, що спирається на застосування ключів K_1^R, K_2^j, K_3^i. K_1^R – унікальне 128-бітне випадкове значення (закритий ключ), згенероване криптографічно сильним ГПВЧ для кожної таблиці R, постійне для всіх значень, які маскуватимуться в цій таблиці; K_2^j – унікальне 128-бітне випадкове значення (закритий ключ), згенероване криптографічно сильним ГПВЧ для кожного атрибуту j деякої таблиці R, постійне для всіх значень, які потрібно маскувати в цьому стовпці; K_3^i – значення (відкритий ключ) цілісного ідентифікатора первинного ключа i-го

рядка таблиці R, постійне для всіх значень у стовпцях, які маскуватимуться в цьому рядку.

Перш ніж розпочати процес маскування чутливих даних (відповідно, як і до інверсного процесу), необхідно виконати певні підготовчі операції. А саме:

1) визначити які дані (яких стовпців, таблиць) мають бути перетворені;

2) згенерувати для таких даних відповідні закриті ключі K_1^R, K_2^j (конфіденційність ключів залежить від того, де вони зберігаються, і хто має до них доступ);

3) сформувати відношення (таблицю) R^{secret}, що містить інформацію, необхідну для приховування та відновлення критичних даних:

$$R^{secret}(n_R, n_j, t, k_1, k_2, h, p, z_{per} \mid n_R \in Nm^{table} \wedge n_j \in Nm^j \wedge t \in T \wedge k_1 \in K_1 \wedge k_2 \in K_2 \wedge h \in Nm^{hash} \wedge p \in Nm^{PRNG} \wedge z_{per} \in \mathbb{N}^*), \quad (3.7)$$

де Nm^{table} – множина імен таблиць, що маскуються (n_R – ім'я деякої таблиці R, $n_R \in Nm^{table}$); Nm^j – множина імен стовпців, що маскуються (збережених програм), n_j – ім'я j-го стовпця (збереженої програми) таблиці R, $n_j \in Nm^j$; T – множина типів об'єктів БД ($T = \{table, procedure, function, package\}$); K_1 – множина унікальних 128-бітних випадкових значень, згенерованих криптографічно сильним ГПВЧ для відповідних таблиць з Nm^{table}; K_2 – множина унікальних 128-бітних випадкових значень, що згенеровані криптографічно сильним ГПВЧ для відповідних стовпців таблиць з Nm^{table}; Nm^{hash} – множина використовуваних геш-функцій ($Nm^{hash} = \{MD4, MD5, SHA-1, SHA-256, SHA-384, SHA-512, SHA-3\}$); Nm^{PRNG} – множина можливих використовуваних ГПВЧ; $\mathbb{N}^*$ – множина натуральних позитивних чисел; z_{per} – кількість повторів перестановок (скільки послідовно раз виконуються операції перестановки). Слід зазначити, що це відношення R^{secret} в подальшому буде розширено (див. п. 3.3.1).

4) згенерувати ключ для таблиці R^{secret} та зашифрувати всі значення її рядків та стовпців одним із криптографічно стійких алгоритмів;

5) створити легітимним користувачам файли-контейнери (стегоконтейнери), що зберігають ключ розшифрування даних таблиці R^{secret}.

Нижче розглянемо основні етапи маскування.

1. Визначення типу об'єкта БД (*type*), імені таблиці ($name^{table}$) та її стовпця ($name^{j}$), елементи даних рядків якого мають маскуватися.

2. Витяг даних (A), що підлягають маскуванню, та ідентифікатора первинного ключа i-го рядка K_3^i з таблиці $name^{table}$.

3. Витяг необхідних для процесу маскування даних із таблиці R^{secret}:

$$Decrypt(R^{\sec ret}[name^{table},\ name^{j}, type]) \to (K_1^R, K_2^j, hash, PRNG, z_{per}).$$

Опосередкований доступ (у реалізації – це спеціальне програмне забезпечення – ПЗ) до закритих ключів K_1^R, K_2^j (а також інших даних таблиці R^{secret}) має авторизований користувач з відповідними привілеями, який пред'явить у відкритому сеансі правильний ключ для розшифрування даних таблиці R^{secret}. При цьому значення цього ключа у відкритому вигляді ніде не повинно відображатися. У реалізації, наприклад, його відстеження не допустимо через наявні засоби документування виконаних запитів (історії запитів). Воно має витягуватись спеціальним ПЗ сервера СКБД зі стегоконтейнера, який пред'явить під час відкриття сеансу автентифікований користувач з відповідними привілеями. Всі інші користувачі, навіть привілейовані, без знання цього ключа, отримати закриті ключі та інші важливі для процесів приховування та відновлення дані з таблиці R^{secret} не зможуть, а, отже, не зможуть провести операції з маскування даних та зворотної до неї операції.

4. Формування початкових значень ГПВЧ – $X_{R_{ij}}^0$.

Для кожного рядка i відповідного стовпця j обраної таблиці R це значення визначається так:

$$X_{R_{ij}}^0 = hash(K_1^R + K_2^j - K_3^i), \qquad (3.8)$$

де $hash()$ – одна з криптографічних геш-функцій (таких як: MD4, MD5, SHA-1, SHA-256, SHA-384, SHA-512, SHA-3). Мета використання геш-функції – перемішування (не ін'єктивне перетворення) закритих та відкритого ключів для неможливості їх відновлення за підсумковим результатом та отримання значно відмінних один від одного сформованих початкових значень $X_{R_{ij}}^0$ для ГПВЧ, навіть у разі зміни хоча б одного з цих ключів на один знак (одиницю). При цьому слід зазначити, що через різну точність подання чисел (для виключення переповнення розрядної сітки) в реалізаціях відповідного методу допускається наступна модифікація виразу (3.8): $X_{R_{ij}}^0 = hash(K_1^R + K_2^j - K_3^i) \bmod(N_{\max})$, тобто застосування додаткової операції (не ін'єктивної) для отриманого значення – взяття за модулем $N_{\max}$, де $N_{\max}$ – максимально допустиме ціле число у відповідної реалізації.

5. Виконання дій відповідно до алгоритму Фішера-Йейтса z_{per} разів:

– генерування (за допомогою одного з вибраних ГПВЧ (*PRNG*), на вхід якого подається відповідне сформоване початкове значення $X_{R_{ij}}^0$) випадкового цілого числа;

– виконання процедури перестановки елементів (байт, символів) початкового рядка A довжиною n.

В результаті маємо перетворене (замасковане) значення деякого поля A кожного рядка i стовпця j відповідного типу обраної таблиці R.

Загальна схема процесу маскування представлена нижче (рисунок 3.3) на псевдокоді. Виконавши відповідні перетворення з маскування вихідних даних, можна розраховувати на досить ефективне їх приховування від зловмисників. По-перше, представлення однакових даних, що зберігаються в БД (у різних рядках таблиці / таблиць), у різному вигляді (правдоподібному, але, у загальному випадку, що не є істинними / справжніми даними) вводять в оману зловмисника. Це обмежує можливості зловмисника щодо використання ним різного набору запитів для логічного висновку (умовиводу). По-друге, без знання початкового значення $X_{R_{ij}}^0$, особливо при великих розмірностях A (великих n), достатньо

складно визначити послідовність випадкових чисел, що генеруються, для перестановки (число яких дорівнює $n!$) елементів даних (різної довжини) кожного рядка (як правило, з дуже великою їх кількістю) таблиці БД, щоб згодом відновити вихідні дані. А неточне відновлення у разі використання, наприклад, частотного аналізу також не завжди прийнятне, тому що для деяких предметних областей навіть один символ у певній специфікації може відігравати принципове значення та відмінність. Все це в цілому сприяє протидії загрозі inference.

Masking process 1 (MP-1)

Input: $name^{table}$, $name^{j}$, $type$
Output: masked value A

```
 1: Select A, K_3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K_1^R, K_2^j, hash, PRNG, z_per)
 3: X_{R_j}^0 = hash(K_1^R + K_2^j − K_3^i)
 4: switch(PRNG)
 5: {case 1: linear congruential generator (LCG)
 6:  case 2: built-in random number generator (package DBMS_RANDOM)
 7:  case 3: Xorshift pseudo random number generator
 8:  ...
 9:  case Θ:...}
10: for k = 1 to z_per          /* number of repetitions of permutations */
11:  for i = n downto 1
12:   j=random_PRNG(1..i) /* a random number is generated in the range [1,i] */
13:   swap(A[i], A[j])      /* exchange */
14:  end for
15: end for
```

Рисунок 3.3. Схема процесу маскування (MP-1)

3.2.2. Характеристика основних етапів процесу зворотного до маскування

У запропонованому рішенні для відновлення замаскованих даних поля рядка використовується підхід до зворотної перестановки, подібний до викладеного в [191], з тією особливістю, що в ньому не обмежуються елементи, що переставляються тільки числами $\{1, 2, 3, ..., n\}$, а можуть використовуватися будь-які символи національних алфавітів, чисел, представлених у шістнадцятковій або інший системі числення. А саме, маючи початкову перестановку:

$$\pi = (\pi(1), \pi(2), ..., \pi(n)) \tag{3.9}$$

та результат її застосування:

$$(y_1, y_2, ..., y_n) = (x_{\pi(1)}, x_{\pi(2)}, ..., x_{\pi(n)}), \tag{3.10}$$

зворотну перестановку можна отримати, використовуючи вираз [189, 191]:

$$\pi^{-1}(\pi(i)) = i, \tag{3.11}$$

як:

$$(x_1, x_2, ..., x_n) = (y_{\pi^{-1}(1)}, y_{\pi^{-1}(2)}, ..., y_{\pi^{-1}(n)}), \tag{3.12}$$

або в матричному вигляді, як:

$$X[\pi(i)] = Y(i). \tag{3.13}$$

Якщо позначити перестановку $\pi_{ch} = \begin{pmatrix} x_1 & x_2 & x_3 & \dots & x_n \\ y_1 & y_2 & y_3 & \dots & y_n \end{pmatrix}$, де $y_i = \pi(x_i)$, $i \in \{1, ..., n\}$, а через $\pi_{num} = \begin{pmatrix} 1 & 2 & 3 & \dots & n \\ a_1 & a_2 & a_3 & \dots & a_n \end{pmatrix}$ її відповідну перестановку, як відображення нумерації на відповідні елементи множини, де $\{a_1, a_2, a_3, ..., a_n\} = \{1, 2, 3, ..., n\}$, тоді вираз (3.12) для зворотної перестановки можна записати наступним чином:

$$\pi_{ch}^{-1}(\pi_{num}(i)) = \pi_{ch}(i). \tag{3.14}$$

Однак спочатку необхідно отримати цю початкову перестановку, перш ніж шукати для неї зворотну. Тому послідовність етапів, що виконуються, буде наступною.

1. Визначення типу об'єкта БД (*type*), імені таблиці ($name^{table}$) та її стовпця ($name^{j}$), елементи даних рядків якого було замасковано.

2. Витяг замаскованих даних (Y) та ідентифікатора первинного ключа i-го рядка K_3^i з таблиці $name^{table}$.

3. Витяг необхідних даних із таблиці R^{secret} для відновлення замаскованих даних.

4. Формування початкових значень $X_{R_{ij}}^0$ для ГПВЧ.

Для кожного i -го рядка j -го стовпця обраної таблиці R початкове значення визначається відповідно до виразу (3.8).

Слід зазначити, що зловмиснику для отримання початкового значення (щоб сформувати початкову перестановку, яка була сформована при маскуванні – МР-1) необхідно знати як мінімум значення двох закритих ключів (K_1^R, K_2^j), а також тип застосованої криптографічної геш-функції та ГПВЧ, що використовується. До того ж кількість перестановок (кількість разів проведених операцій з перестановки символів рядка) для кожного конкретного рядка таблиці, що захищається, може бути різним. Перебір же «грубою силою» лише двох 128-бітних випадкових чисел, згенерованих криптографічно сильним ГПВЧ, дуже ресурсомістка задача, щоб її реалізувати за прийнятний час. Кількість комбінацій (без урахування визначення використовуваної геш-функції, ГПВЧ та кількості операцій з перестановки символьного рядка, що проводяться), які необхідно перебрати для j стовпців таблиці R дорівнює як мінімум $\frac{1}{2}\cdot j\cdot 2^{128}\cdot 2^{128} = j\cdot 2^{255} \approx 5.8\cdot j\cdot 10^{76}$ при Brute-force attack з 50% шансом (половиною від можливих комбінацій).

5. Визначення початкових перестановок елементів даних кожного рядка відповідної таблиці БД.

При цьому, природно, як ГПВЧ вибирається саме той генератор, який використовувався при маскуванні даних. Завдяки можливості повторення послідовності чисел, що формуються ГПВЧ, з одного і того ж початкового значення, виконуючи дії відповідно до алгоритму Фішера-Йейтса, отримаємо початкову перестановку: $\pi = (\pi(1), \pi(2), ..., \pi(n))$ (або в інших позначеннях $\pi_{num} = \begin{pmatrix} 1 & 2 & 3 & ... & n \\ a_1 & a_2 & a_3 & ... & a_n \end{pmatrix}$ – як відображення нумерації на відповідні елементи множини), аналогічну, отриманої при маскуванні (МА-1) для відповідного елемента даних кожного i -го рядка j -го стовпця таблиці R.

6. Здійснення перетворення зворотного до маскування.

Визначивши початкову перестановку $\pi = (\pi(1), \pi(2), ..., \pi(n))$, і, маючи вхідний рядок замаскованих даних $Y(i)$, відповідно до виразу (3.13) можна визначити вихідне (не замасковане) значення поля рядка $X(i)$, де $i \in \{1, ..., n\}$.

Загальна схема процесу зворотного маскування (IMP-1) представлена рисунку 3.4.

Inverse masking process 1 (IMP-1)

Input: $name^{table}$, $name^{j}$, $type$
Output: source (not masked) value - X

```
 1: Select A, K_3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K_1^R, K_2^j, hash, PRNG, z_per)
 3: X_{R_j}^0 = hash(K_1^R + K_2^j − K_3^i)
 4:   for i = 1 to n
 5:     π_num[i] = i      /* array preparation */
 6:   end for
 7: switch(PRNG)
 8: {case 1: linear congruential generator (LCG)
 9:  case 2: built-in random number generator (package DBMS_RANDOM)
10:  case 3: Xorshift pseudo random number generator
11:  ...
12:  case Θ:...}
13:  for k = 1 to z_per
14:    for i = n downto 1   /* getting the initial permutation */
15:      j=random_PRNG(1..i)
16:        swap(π_num[i], π_num[j])
17:    end for
18:    for i = 1 to l       /* inverse permutation */
19:      X[π_num[i]] = Y(i)
20:    end for
21:  end for
```

Рисунок 3.4. Схема процесу зворотного маскування (IMP-1)

3.2.3. Важливі модифікації методу маскування даних

Досить часто виникає необхідність маскування та інверсної (зворотної до маскування) процедури частини значення поля рядка таблиці. Наприклад, для маскування телефонних номерів, дисконтних карт, серійних номерів техніки, автомобільних номерів, даних, що подаються у вигляді великих двійкових об'єктів тощо. Тому представлені вище схеми (MP-1, IMP-1), які реалізують методи маскування та його інверсії, доцільно незначним чином модифікувати. Нижче наведено відповідні модифікації (MP-2, IMP-2) схем цих методів (рисунки 3.5,

3.6). Відмінності від МА-1 та ІМА-1 виділені кольором, жирним шрифтом та підкресленням відповідних параметрів.

Masking process 2 (MP-2)

Input: $name^{table}$, $name^{j}$, $type$, **iend**, **ibeg**
Output: masked value A

```
 1: Select A, K3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K1^R, K2^j, hash, PRNG, z_per)
 3: X_Rj^0 = hash(K1^R + K2^j − K3^i)
 4: switch(PRNG)
 5: {case 1: linear congruential generator (LCG)
 6:  case 2: built-in random number generator (package DBMS_RANDOM)
 7:  case 3: Xorshift pseudo random number generator
 8:  ...
 9:  case Θ:...}
10: for k = 1 to z_per          /* number of repetitions of permutations */
11:  for i = n-iend downto ibeg
12:   j=random_PRNG(ibeg..i) /* a random number is generated in the range [ibeg,i] */
13:   swap(A[i],A[j])      /* exchange */
14:  end for
15: end for
```

Рисунок 3.5. Схема процесу маскування (MP-2)

Inverse masking process 2 (IMP-2)

Input: $name^{table}$, $name^{j}$, $type$, **iend**, **ibeg**
Output: source (not masked) value - X

```
 1: Select A, K3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K1^R, K2^j, hash, PRNG, z_per)
 3: X_Rj^0 = hash(K1^R + K2^j − K3^i)
 4:   for i = 1 to n
 5:     π_num[i] = i       /* array preparation */
 6:   end for
 7: switch(PRNG)
 8: {case 1: linear congruential generator (LCG)
 9:  case 2: built-in random number generator (package DBMS_RANDOM)
10:  case 3: Xorshift pseudo random number generator
11:  ...
12:  case Θ:...}
13:  for k = 1 to z_per
14:    for i = n-iend downto ibeg    /* getting the initial permutation */
15:      j=random_PRNG(ibeg..i)
16:        swap(π_num[i],π_num[j])
17:    end for
18:    for i = 1 to l         /* inverse permutation */
19:      X[π_num[i]] = Y(i)
20:    end for
21:  end for
```

Рисунок 3.6. Схема процесу зворотного маскування (IMP-2)

Де параметри ibeg, iend задають межі інтервалу перетворення, а саме: ibeg – з якої позиції від початку елемента даних, що перетворюється, слід проводити перестановку; iend – до якої позиції з кінця елемента даних, що перетворюється, слід проводити перестановку.

Крім того, наприклад, щоб правдоподібним чином замаскувати номер банківської картки (з урахуванням правил їх формування), необхідно не тільки виконати відповідні перетворення над частиною її номера, але також необхідно обчислити останню контрольну цифру, використовуючи алгоритм Луна [198], відповідно до стандарту ISO/IEC 7812. Код функції, що реалізує алгоритм Луна мовою PL/SQL представлений у Додатку В.1. Тоді схеми MP-2, IMP-2 можна модифікувати, додавши в кінці (після отримання замаскованого або вихідного значення) відповідне перетворення, яке буде виконуватися при необхідності (див. схеми MP-3, IMP-3; рисунки 3.7, 3.8).

Masking process 3 (MP-3)

Input: $name^{table}$, $name^{j}$, *type*, *iend*, *ibeg*, **PRLuhn**
Output: masked value *A*

```
 1: Select A, K3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K1^R, K2^j, hash, PRNG, z_per)
 3: X^0_Rj = hash(K1^R + K2^j − K3^i)
 4: switch(PRNG)
 5: {case 1: linear congruential generator (LCG)
 6:  case 2: built-in random number generator (package DBMS_RANDOM)
 7:  case 3: Xorshift pseudo random number generator
 8:  ...
 9:  case Θ:...}
10: for k = 1 to z_per          /* number of repetitions of permutations */
11:  for i = n-iend downto ibeg
12:   j=random_PRNG(ibeg..i) /* a random number is generated in the range [ibeg,i] */
13:   swap(A[i],A[j])      /* exchange */
14:  end for
15: end for
16: if PRLuhn then
17:    A[n]= Luhn(A)     /* functions that implement the Luhn algorithm */
18:  end if
```

Рисунок 3.7. Схема процесу маскування (MP-3)

У таблицях 3.1-3.3 наведено результати маскування (із збереженням типу картки) номерів банківських карток: Visa – номер '4454102135347018', Master Card – номер '5167135104128196', картки платіжної системи American Express – '378282246310005', що зберігаються у відповідних послідовностях поруч

розташованих рядків деякої таблиці, за допомогою запропонованого методу, використовуючи три розглянуті вище (п. 3.2) ГПВЧ (стовпець DATA_MASKING_1 Таблиці 3.1 – для ГПВЧ-1; стовпець DATA_MASKING_2 – для ГПВЧ-2; стовпець DATA_MASKING_3 – для ГПВЧ-3).

Inverse masking process 3 (IMP-3)

Input: $name^{table}$, $name^{j}$, *type*, **iend**, **ibeg**, **PRLuhn**
Output: source (not masked) value - X

```
 1: Select A, K3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K1^R, K2^j, hash, PRNG, z_per)
 3: X^0_Rj = hash(K1^R + K2^j − K3^i)
 4:   for i = 1 to n
 5:     π_num[i] = i        /* array preparation */
 6:   end for
 7: switch(PRNG)
 8: {case 1: linear congruential generator (LCG)
 9:  case 2: built-in random number generator (package DBMS_RANDOM)
10:  case 3: Xorshift pseudo random number generator
11:  ...
12:  case Θ :...}
13:  for k = 1 to z_per
14:    for i = n-iend downto ibeg    /* getting the initial permutation */
15:      j=random_PRNG(ibeg..i)
16:        swap(π_num[i], π_num[j])
17:    end for
18:    for i = 1 to l      /* inverse permutation */
19:      X[π_num[i]] = Y(i)
20:    end for
21:  end for
16: if PRLuhn then
17:    A[n]= Luhn(A)     /* functions that implement the Luhn algorithm */
18:  end if
```

Рисунок 3.8. Схема процесу зворотного маскування (IMP-3)

Таблиця 3.1 – Результати маскування номера банківської картки Visa

ID	DATA_MASKING_1	DATA_MASKING_2	DATA_MASKING_3
35359	4411443305170521	4401235043741510	4400451172135348
35360	4413021740415533	4437312451510402	4413535021740418
35361	4434230511514078	4451124347013504	4433412071450155
35362	4441031715425306	4454317254110308	4445701534021132
35363	4410203415715438	4401473215035144	4401112347355406
35364	4471310435041525	4431424310075158	4403441103715257
35365	4442031413755102	4453113442705013	4450104341275318
35366	4434032105174511	4470310431215454	4432740405113516
35367	4442714501531309	4413274510410354	4401351344017250
35368	4411233574040152	4401354023541715	4470315450124131

Таблиця 3.2 – Результати маскування номера банківської картки Master Card

ID	DATA_MASKING 1	DATA_MASKING 2	DATA_MASKING 3
35369	5174311028615912	5121381165147098	5173192416150188
35370	5193751482161100	5116471112850931	5111091716482355
35371	5193711261150849	5173051612198419	5131072681541199
35372	5174513918161028	5124311678019515	5171350492111688
35373	5110341258911670	5176018951231413	5162081179141356
35374	5111501217936485	5114811259061374	5193110548211674
35375	5105311179468123	5168143975012118	5143196250811174
35376	5113517462018918	5129713164051180	5171591241013689
35377	5101512147198361	5164113082517196	5192304161118750
35378	5141351291186076	5118107163219455	5171084112195363

Таблиця 3.3 – Результати маскування номера картки платіжної системи American Express

ID	DATA_MASKING 1	DATA_MASKING 2	DATA_MASKING 3
35379	372800263182043	370020288642317	371288240036022
35380	378022631084025	370816438022205	372206321048087
35381	372006822180433	370010822348265	372142063008825
35382	373206140282809	372382004682011	376482020308215
35383	372860104823205	371242806280300	372200346821081
35384	370280463102285	376182320802400	372302880420161
35385	372843260180024	376048820221039	372042318028604
35386	378231264800025	376082412008322	371862302802040
35387	372012648080237	370102628084325	373802460082124
35388	372020023868415	378023801262045	372822460183004

У разі потреби проведення відповідних перетворень над чутливими даними, що зберігаються у БД у вигляді BLOB-об'єктів (наприклад, важливі документи форматів: doc, docx, rtf, xls, xlsx, xps, pdf, ppt, bmp, gif, tif, mp3, avi і т. д.), можна також скористатися запропонованим методом. Для цього необхідно врахувати деякі особливості, пов'язані з сигнатурами формату (як набору байтів, що забезпечує ідентифікацію типу файлу) даних, що перетворюються (див. Додаток В.2), байтовим представленням елемента даних, над яким виконується перетворення, зменшенням кількості байт, що переставляються, BLOB, які зберігають зображення (за рахунок знаходження деякої частини об'єкта, перестановка елементів якої дозволить приховати дані в цілому).

3.2.4. Порівняльний аналіз можливостей запропонованого методу та методу шифрування щодо приховування даних

Враховуючи, що захист конфіденційності може забезпечити широкий спектр заходів безпеки, а кожна організація має оцінити нюанси

конфіденційності, які вона хоче забезпечити у кожному конкретному випадку, оскільки відомо [60], що інструменти та технології, що реалізують одну форму конфіденційності, можуть не підтримувати інші форми, порівняльний аналіз проведемо з урахуванням поставленого основного завдання – ефективного приховування даних, що становлять інтерес для зловмисника, який прагне реалізувати загрозу логічного висновку.

Аналіз якісних характеристик

Для кращого представлення та розуміння відповідного перетворення даних (якісних характеристик) у таблиці 3.4 наведено результати маскування за допомогою запропонованого методу при використанні трьох різних ГПВЧ (див. п. 3.2; відповідні стовпці `DATA_MASKING_1`, `DATA_MASKING_2`, `DATA_MASKING_3`) та шифру AES-128 (у реалізації пакету `DBMS_CRYPTO` СКБД Oracle) рядки символів – `'КП173478'` (номер паспорта деякого суб'єкта). Для наочності були обрані поруч розташовані рядки (рядки таблиці з нумерацією, що послідовно збільшується). Хоча насправді у реальних таблицях БД вони, зазвичай, розташовуються у довільних рядках відповідного атрибута таблиці, що для зловмисника, який намагається реалізувати загрозу логічного висновку, є ускладнюючою обставиною.

При шифруванні символьних рядків за допомогою алгоритму AES (результати представлені в стовпці `DATA_ENCRYPT` таблиці 3.4) використовувався той же механізм вилучення та застосування ключів з таблиці R^{secret}, що у запропонованому алгоритмі маскування. При цьому результат виразу (3.8) використовувався для визначення вектора ініціалізації в алгоритмі шифрування.

Як видно з таблиці 3.4, усі отримані рядки відрізняються від вихідного, що дуже важливо. При цьому перетворені за допомогою пропонованого методу рядки на відміну від класичного шифрування, залишаючись правдоподібними, дозволяють до того ж виконувати перетворення над частиною даних, зберігаючи формат і не збільшуючи розмір рядка. Що у певних застосунках має важливе значення, оскільки подібні зміни формату, розміру для них є неприпустимим. У цьому відношенні звичайний алгоритм шифрування програє запропонованому методу. Можна, звичайно, скористатися так званим шифруванням зі збереженням

формату (англ. format-preserving encryption – FPE), але реалізація подібних алгоритмів, наприклад, в Oracle, як зазначається у відповідному посібнику [51], вимагає значної обробки.

Таблиця 3.4 – Результати перетворених даних

ID	DATA_MASKING_1	DATA_MASKING_2	DATA_MASKING_3	DATA_ENCRYPT
21	КП847137	КП137478	КП478317	EEA84E9F7E39AB03F2B91CA1AA5201A6
22	КП381477	КП374781	КП738471	E473C54DBF1D0268FF095E9A42CA947B
23	КП847713	КП837741	КП734871	E042944B236A2DD825CCB216560794A7
24	КП347871	КП477138	КП734871	CEC630EEC6214C4DC8C592A8AE1561F7
25	КП748371	КП314787	КП478317	CF4E4713110AF34D24E15FD96BC4C92A
26	КП734871	КП378741	КП377814	D05850D99C6EDB391E089B2D6BC0D207
27	КП347781	КП143877	КП477381	D5622782D852C4496DBA4C75C51CAECD
28	КП487371	КП713847	КП747831	9B7FBDA4B93B771C4D438A3B5DE7569B
29	КП743871	КП371847	КП734178	61EE40DA6BC15A94962BDFD49C21CFF5
30	КП317874	КП341787	КП734781	23F869F251803C499AE7867BCE56B91B
31	КП371748	КП147387	КП774381	FF7EAC80B98EA7FD4EA8AF2828DC191E
32	КП781347	КП378174	КП384771	573DE7DBFFD4A910B75BB4AF12D88609
33	КП837714	КП873471	КП877143	E1E51CBB388BFAC57FA9928C68B67FBA
34	КП743871	КП418377	КП371874	1C0C90C6B4F8C29BC90BDCBF2DA1A856
35	КП731784	КП147738	КП837714	D8A197854B41D54385310F3163E03479
36	КП714873	КП143787	КП371874	7E7D9F3F5D0F864B8EE9C112EB5982BD
37	КП787143	КП871374	КП478713	098920EC125DE02FBFA55935D5DC7C51
38	КП478731	КП348177	КП813747	9FC95C39A56520E2D87322BF4EC764FB
39	КП747813	КП431787	КП734718	63C8F6B1B9EC3B47F57D665E2322A86E
40	КП743187	КП774138	КП473817	3AD54FD4E9218153D1B35513D40DBF24
41	КП478137	КП438177	КП841377	C134F5F7594ED028BB7EA40C9282A6CE
42	КП774381	КП381774	КП747813	0633AD01AFE70AC3F0D957994338EC15
43	КП747831	КП178734	КП737481	F910D6AFB157CF55E70036CE6CB5DD7F
44	КП734871	КП417837	КП874317	3A22B18F0E0C40210BFB53B8BA40BBD7
45	КП743781	КП187473	КП837417	0F3DA9F05B5C3B46A471052ED82D39E2

Аналізуючи результати таблиці 3.4, можна зробити висновок, що зловмисник з дуже малою ймовірністю за наявною інформацією зможе ідентифікувати, що всі ці значення пов'язані з одним і тим самим символьним рядком `'КП173478'` (тобто з одним і тим самим реальним об'єктом – конкретною особистістю, наприклад, що здійснює деякі операції з пред'явленням свого паспорта: укладання контрактів, угод, купівля/продаж валюти тощо). Хоча кількість перестановок не така і велика 6!=720 (перетворенню підлягає тільки частина номера паспорта – цифри), але в даному випадку утруднюється навіть статистичний криптоаналіз (який можна ще більше ускладнити, застосувавши до результату перетворень метод заміни; більш докладно про цей підхід викладається у п. 3.3.1), оскільки насправді після відповідної псевдонімізації [17] виявляється, що з отриманими номерами будуть пов'язані зовсім інші та різні суб'єкти. Це заплутує реальні дані (дані не можуть бути співвіднесені з конкретним суб'єктом даних без використання додаткової інформації, що

рекомендується європейським Регламентом GDPR) та суттєво ускладнює реалізацію зловмисником загрози inference. Отже, пропонований метод приховування даних не дозволить зловмиснику отримати за прийнятний для нього час дані, доступ до яких безпосередньо йому закритий.

Аналіз кількісних характеристик

Нижче наведені результати були отримані при відповідних перетвореннях даних таблиць бази даних, реалізованої на різних комп'ютерах на платформі СКБД Oracle 12.2 c, встановленій на операційну систему Windows 10 (x64).

Варіант А – використовувався комп'ютер із процесором Intel(R) Core(TM)2 Duo CPU 2.16 GHz; RAM – 4 Гб; HDD – 320 Гб.

Варіант В – використовувався комп'ютер із процесором Intel(R) Core (TM) i3-7100 CPU 3.90 GHz; RAM – 4 Гб; HDD – 500 Гб. Перетворення (маскування, шифрування) із записом в БД піддавалися близько 20 000 рядків даних деякого стовпця типу varchar2(255) таблиці реальної БД.

На рисунках 3.9, 3.10 представлені результати середніх часів (у секундах) маскування та зворотного до маскування процесу, шифрування та розшифрування рядків символів із записом у БД всіх рядків даних із застосуванням запропонованого методу (використовувалися конгруентний ГПВЧ (ГПВЧ-1) та генератор псевдовипадкових чисел Xorshift (ГПВЧ-3)) та алгоритму звичайного шифрування AES-128 для варіантів А, В відповідно.

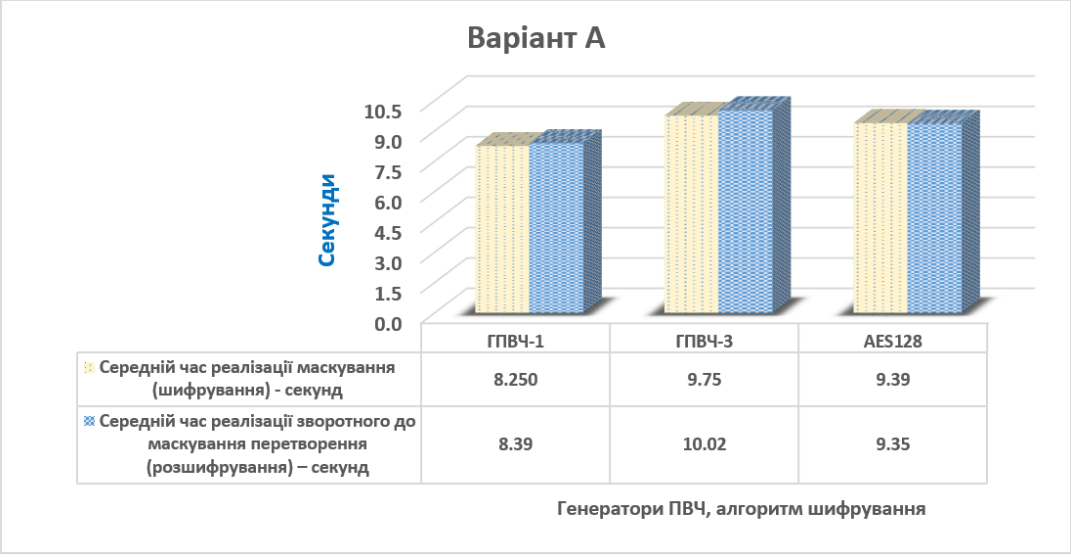


Рисунок 3.9. Середній час перетворення рядків символів для варіанта А

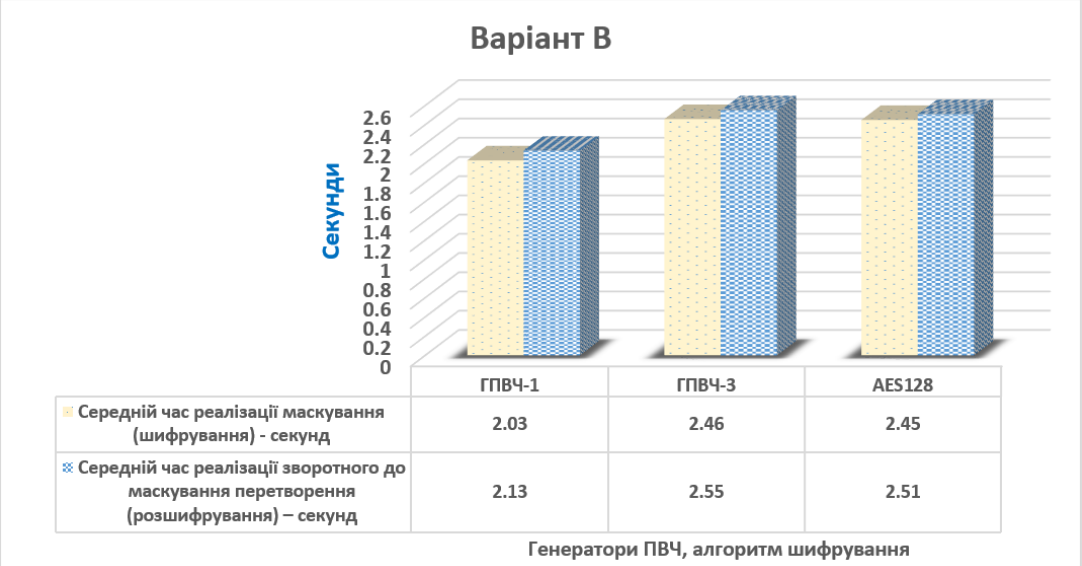


Рисунок 3.10. Середній час перетворення рядків символів для варіанта В

Як випливає з проведених досліджень, використання звичайного шифрування програє пропонованим методом не тільки в якісних характеристиках (див. вище), але і кількісних – через збільшення довжини даних, що зберігаються, та часу перетворення з приховування даних. Якщо при використанні ГПВЧ Xorshift порівнювані часи відповідних перетворень для обох методів практично однакові, що пояснюється не оптимізованою на момент експерименту реалізацією ГПВЧ Xorshift мовою PL/SQL (дана мова не підтримує реалізацію логічних зсувів, операцій виключне АБО, які доводилося вирішувати алгебраїчно), то вже при використанні ГПВЧ-1 реалізація відповідних перетворень з використанням шифрування програє запропонованому методу (11-12) % для варіанта А та (15-17) % для варіанту В відповідно.

3.2.5. Рекомендації щодо застосування методів маскування даних

При довжині рядка менше 3 символів або, якщо всі символи однакові, маскування із застосуванням методу перестановки елементів даних поля рядка деякої таблиці безпосередньо недоцільно. У цьому випадку розумніше використовувати:

– метод підстановки з елементами пропонованого підходу, а саме шляхом створення таблиці зіставлення з випадковим вибором з неї або символів, або їх комбінацій, використовуючи для цієї мети наявні ГПВЧ і механізм тасування запропонованого методу перестановки (з можливістю згодом відновити послідовність чисел, що генеруються з початкового значення $X_{R_{ij}}^{0}$, необхідну для зворотного перетворення);

– метод маскування даних на основі обчислення операцій за модулем (якщо допускається зміна типу або розмірності) шляхом перетворення короткого рядка до чисельного вигляду, якщо до нього входять символи літер, спеціальні символи та цифри;

– або модифікацію даного методу перестановки шляхом доповнення вихідного рядка даних (спочатку або наприкінці неї) деякою кількістю випадково вибраних символів, які при зворотній до маскування операції цього методу повинні бути безсумнівно видалені.

Для утруднення проведення статистичного криптоаналізу можливе використання спільно з методом перестановки методу заміни, що передбачає використання, наприклад, як алфавіт заміни символи з стандарту *Unicode* (даний підхід докладніше буде розглянуто нижче). Також ускладнити зловмиснику задачу розкриття чутливої інформації можна шляхом збільшення розмірності елементів, що маскуються, даних поля деяких рядків таблиці до заданої величини (якщо це допускається) за рахунок додавання випадкових символів з обраного алфавіту (або отриманих на основі семантичного аналізу аналізованої ПрО), до яких застосовуються аналогічні розглянутим вище перетворення (див. приклади у Додатку В.3). Природно, кількість символів, що доповнюються, повинна бути визначена (наприклад, щодо R^{secret}), і вони після зворотного до маскування перетворення повинні бути видалені.

У пропонованих методах маскувати передбачається лише неключові поля обраних таблиць. На даний момент для полів таблиць типу BLOB, що мають велику розмірність, застосування методу маскування на основі перестановки всіх

байт BLOB призводить до значних витрат часу. Щоб скоротити час реалізації даного методу можна розпаралелити процеси формування масивів образів значень полів великих двійкових об'єктів, що маскуються, і використовувати деякі сучасні алгоритми перестановки, які з появою великих базових кластерів і графічних процесорів мають можливість використовувати переваги розподіленого обчислювального середовища. Наприклад, такі як версії з розпаралелюванням алгоритму Rao-Sandelius та алгоритму Merge Shuffle [194, 199]. Інший передбачуваний спосіб скорочення часу перетворення – використання шаблонів перемішування, подібним до тих, що пропонуються в роботі [200], але формування більшої їх кількості. Найбільш простим рішенням бачиться варіант удосконалення пропонованого методу маскування BLOB шляхом зменшення кількості байт, що переставляються, об'єктів типу BLOB, за рахунок знаходження деякої частини об'єкта, перестановка елементів якої дозволить приховати дані в цілому. З цією метою можна скористатися як емпіричним методом, так і дослідженнями у сфері матричного маскування.

3.3. Метод приховування вихідного коду збережених програм

3.3.1. Характеристика основних етапів процесу маскування

Як уже зазначалося в розділі 1 (п. 1.1) сьогодні, незважаючи на проведені дослідження та наявні рішення, існує потреба в певному перегляді підходу до приховування коду програм, що зберігаються, результатом якого були б певні методи, прийоми, засоби, актуальні як у теоретичному, так і в прикладному аспекті. В основу запропонованого методу покладена та ж, що і наведена в п. 3.2.1, універсальна схема приховування даних, що спирається на застосування ключів K_1^R, K_2^j, K_3^i. З урахуванням особливостей призначення модулів (PSM), що постійно зберігаються, важливості їх цілісності і справжності схема МР-2 була дещо змінена (пов'язано з додаванням атрибутів k_Σ, l у відношення R^{secret}). В результаті схему методу маскування коду збережених програм можна представити наступним чином (рисунок 3.11).

Masking process 2a (MP-2a)

Input: $name^{table}$, $name^{j}$, *type*, *iend*, *ibeg*
Output: masked value A

```
 1: Select A, K3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K1^R, K2^j, hash, PRNG, z_per, k_Σ, l)
 3: X^0_Rj = hash(K1^R + K2^j − K3^i)
 4: switch(PRNG)
 5: {case 1: linear congruential generator (LCG)
 6:  case 2: built-in random number generator (package DBMS_RANDOM)
 7:  case 3: Xorshift pseudo random number generator
 8:  ...
 9:  case Θ:...}
10: for k = 1 to z_per          /* number of repetitions of permutations */
11:  for i = l-iend downto ibeg
12:   j=random_PRNG(ibeg..i) /* a random number is generated in the range [ibeg,i] */
13:   swap(A[i],A[j])          /* exchange */
14:  end for
15: end for
```

Рисунок 3.11. Схема процесу маскування (MP-2a)

Де $name^{table}$ – ім'я таблиці R, що маскується (наприклад, при створенні збережених програм у словнику бази даних СКБД Oracle зберігаються і байт-код програм, та їх вихідний код, який доступний для рядкового перегляду в таких системних уявленнях, як: `DBA_SOURCE`, `ALL_SOURCE`, `USER_SOURCE`, `INT$DBA_SOURCE`, `CDB_SOURCE` і системної таблиці `SYS.SOURCE$` (у контексті аналізованого підходу це і є таблиця R)); $name^{j}$ – ім'я j-й маскованої програми, що зберігається в таблиці R; *type* – тип об'єкта БД (процедура, функція, пакет тощо); A – рядки вихідного коду (X) / замаскованого коду (Y) програми, що зберігається; *hash*() – одна з криптографічних геш-функцій (наприклад, MD4, MD5, SHA-1, SHA-256, SHA-384, SHA-512, SHA-3 [201]); *PRNG* – використовуваний ГПВЧ (зі списку наявних); z_{per} – кількість повторів перестановок; k_{Σ} – еталонна контрольна сума вихідного коду збереженої програми; l – довжина коду збереженої програми в символах.

Як і в аналізованому вище методі, перед початком процесу маскування критичних програм, що зберігаються, необхідно виконати певні підготовчі операції: визначити які PSM повинні бути перетворені; згенерувати для вибраних

PSM відповідні ключі K_1^R, K_2^j; сформувати відношення (таблицю) R^{secret}, що містить інформацію, необхідну для приховування та відновлення критичних даних:

$$R^{secret}(n_R, n_j, t, k_1, k_2, h, p, z_{per}, k_\Sigma, l \mid n_R \in Nm^{table} \wedge n_j \in Nm^j \wedge t \in T \wedge k_1 \in K_1 \wedge k_2 \in K_2 \wedge h \in Nm^{hash} \wedge p \in Nm^{PRNG} \wedge z_{per} \in \mathbb{N}^* \wedge k_\Sigma \in \mathbb{N}^* \wedge l \in \mathbb{N}^*), \quad (3.15)$$

а також згенерувати (якщо він ще не був згенерований або потрібно його перестворення) ключ для таблиці R^{secret} та зашифрувати всі значення її рядків та стовпців одним із криптографічно стійких алгоритмів; створити (якщо вони ще не були створені) легітимним користувачам стегоконтейнери, що зберігають ключі розшифрування даних таблиці R^{secret}.

Виконавши відповідно до схеми МР-2а дії маскування вихідного коду PSM, можна розраховувати на досить ефективне його приховування від зловмисників. Зважаючи на те, що при великій довжині коду l (великих розмірностях A) без знання $X_{R_j}^0$ дуже складно визначити послідовність випадкових чисел, що генеруються, для перестановки. Якщо наприклад скористатися методом простого перебору, то число можливих варіантів послідовностей випадкових чисел, що генеруються для перестановки складе $l!$.

Наприклад, для знаходження методом перебору (brute-force attack) послідовності генерованих випадкових чисел для перестановки коду деякої збереженої процедури довжиною l=230 символів (число можливих перестановок: $230! \approx 7.76 \cdot 10^{444}$) у деякій конкретної реалізації, коли:

– початкове значення послідовності $X_{R_j}^0$, використовуване в ГПВЧ, обчислюється відповідно до виразу: $X_{R_j}^0 = hash(K_1^R + K_2^j - K_3^i) \bmod(N_{\max})$ і для запобігання переповнення розрядної сітки в наступних обчисленнях $N_{\max}$ визначається з діапазону цілих чисел, що не перевищують, наприклад, 4294967295;

– відомий використовуваний ГПВЧ (у випадку, що розглядається, це ЛКГ), геш-функція і $z_{per}=1$,

може знадобитися більше двох місяців при використанні комп'ютера з процесором Intel(R) Core (TM) i3-7100 CPU 3.90 GHz; RAM – 4 Гб; HDD – 500 Гб із встановленою на операційну систему Windows 10 (x64) СКБД Oracle 12.2 с.

Природно, якщо невідомі використовувані геш-функція, ГПВЧ (і цей генератор має більше внутрішніх станів), $z_{per}>1$ (шифр перестановки стає значно безпечнішим [47]), то час визначення послідовності генерованих випадкових чисел для перестановки коду деякого PSM, а отже, і $X_{R_j}^0$ шляхом перебору зростає. У загальному випадку, при більшому $N_{\max}$, більшій довжині коду вихідної програми, що підлягає маскуванню, цей час зростає дуже швидко, у зв'язку з тим, що факторіал зростає швидше, ніж будь-яка показникова функція або будь-яка степенева функція, а також швидше, ніж будь-яка сума добутків цих функцій.

І навіть при обмеженнях сучасного варіанта алгоритму тасування Фішера-Йейтса [199], який не має можливості створити більше перестановок, ніж кількість внутрішніх станів генератора *m*, кількість перестановок є досить великою. Так, наприклад, для генератора псевдовипадкових чисел G. Marsaglia Xorshift [18] з $m=2^{128}-1$ число перестановок може сягати $3.4\cdot10^{38}$, а ГПВЧ, рекомендованого у роботі [197], приблизно до $3.138\cdot10^{57}$. До того ж у більшості випадків не потрібно отримувати всі перестановки [186]. Проте, задля справедливості, слід зазначити, що перестановочний шифр вразливий для методів частотного аналізу [188]. Тому, швидше за все, метод грубої сили при великих значеннях l, $z_{per}>1$, невідомих геш-функціях, ГПВЧ буде не ефективним. Найбільш ефективним методом розкриття замаскованих збережених програм в цьому випадку може стати метод статистичного аналізу. Хоча і його застосування теж може не досягти бажаного ефекту для зловмисника через те, що кожен символ і значення збережених програм відіграють значну роль для виконуваних команд. Неточне відновлення їхнього коду неприйнятне. Оскільки навіть незначні зміни

символів і значень (констант) у коді програми, що зберігається, можуть призвести до абсолютно протилежних дій. Навіть перебравши всі допустимі варіанти перестановок не завжди ясно, який із цих варіантів істинний. Правдоподібними варіантами будуть, наприклад: $a1 > b2$, $a2 \le b1$, $ab = 21$, $x = 1$, $x = 0$ тощо. При цьому можна здійснювати маскування не всього коду програми, що зберігається, а тільки деякої його частини (без заголовка збереженої програми і її закінчення), щоб у зловмисника не було певної апріорної інформації для спрощення криптоаналізу. Тому не знаючи ключа $X_{R_j}^0$ для ГПВЧ, коректно відновити початковий код вихідної процедури дуже складно. Принаймні, це зробити набагато складніше, ніж за допомогою існуючих сучасних інструментальних засобів (див. Додаток В.4), які дозволяють легко відновлювати вихідні коди програм, що зберігаються за допомогою вбудованих в СКБД (причому не в усі) засобів.

Щоб ще ускладнити проведення статистичного аналізу, нижче пропонується підхід комбінованого використання методів перестановки та заміни (поліалфавітний шифр). Слід зазначити, що операції методу, що виконуються відповідно до схеми процесу маскування МР-2а, не призводять до зміни формату вихідних даних (рядків PSM коду) і довжини коду. Це дуже важливо для реалізації можливості виконання необхідних наступних дій, пов'язаних із збереженням коду програми на сервері БД та первинною контрольованістю цілісності коду замаскованої програми, а саме, можливості оцінити чи була змінена довжина коду (незалежно від того, зроблено це було помилково чи навмисно зловмисником, шкідливою програмою). В подальшому для контролю цілісності коду PSM використовується процедура порівняння контрольної суми відновленого коду програми з еталонною, що зберігається у зашифрованому вигляді у таблиці R^{secret}. Це дозволяє переконатися в незмінності отриманого в результаті зворотного маскування перетворення коду PSM і гарантує можливість його використання без ризику виконання ним будь-яких недокументованих (шкідливих) дій.

Виконати перетворення відповідно до МР-2а це ще тільки частина процесу маскування коду збереженої програми, на відміну від процесу маскування даних відповідних атрибутів таблиць. Отриманий перетворений код збереженої програми необхідно зберегти на сервері. І зробити це можна лише після виконання відповідних операторів мови SQL. А саме, для того щоб перетворений код процедури, функції, пакета, що маскується, зберігся на сервері (навіть з помилками), необхідне застосування оператора створення відповідного об'єкта (`CREATE ...`). Тому отриманий результат перестановки необхідно об'єднати шляхом конкатенації з конструкцією `CREATE`:

$$Y' = f_1(Y) = 'CREATE...' \| chr(10) \| Y, \qquad (3.16)$$

де $chr(10)$ – оператор (керуючий символ) зміни рядка (LF, line feed).

Наприклад, для деякої процедури `DEMO`, що зберігається на сервері СКБД Oracle, для якої виконано перетворення з маскування її коду, це буде наступний рядок з конструкцією CREATE:

```
create or replace procedure DEMO (salary IN NUMBER) as
```

з якою конкатенується перетворений код процедури. Ось чому важливо не змінювати формат вихідних даних у процесі операції маскування.

Після компілювання цей код буде збережено на сервері, хоча сам об'єкт (збережена процедура) матиме статус недійсного (`INVALID`). Але при цьому авторизований ктаористувач може в будь-який момент часу, використовуючи процедуру зворотного перетворення, відновити його до початкового вигляду та після відповідної компіляції призвести до статусу дійсного об'єкта (`VALID`). При цьому відновлений код програми після виконання дій авторизованим користувачем або процесом може бути знову замаскований. Конкретний вид рядка з конструкцією `CREATE` (з точною вказівкою параметрів), яка об'єднується шляхом конкатенації з перетвореним кодом процедури маскування, великого значення не має. Головна роль рядка з конструкцією `CREATE` – це можливість зберегти після відповідної процедури компіляції отриманого перетвореного коду

на сервері. Приклади перетворення (маскування) деякої збереженої процедури відповідно до МР-2а наведені в Додатку В.5.

Щоб створений об'єкт (збережена процедура, функція) залишався дійсним (`VALID`), хоча по суті, своїх справжніх функцій він не зможе виконувати, можна дещо модифікувати запропонований вище спосіб конкатенації конструкції `CREATE` з маскованим кодом програми, що зберігається. Наприклад, як початкова конструкція, яка конкатенується з маскованим кодом PSM, можна використовувати за певних умов (враховуючи, що вкладення багаторядкових коментарів один в одного не допускається) наступні рядки мови PL/SQL:

```
create or replace procedure DEMO (salary IN NUMBER) as
begin
null;
      /*
```

а в якості заключної конструкції, що конкатенується, – наступні рядки коду:

```
*/
end;.
```

Або використати директиви умовної компіляції. Для цього слід визначити початкову конструкцію:

```
create or replace procedure DEMO (salary IN NUMBER) as
begin
null;
$IF false $THEN /*
```

та заключну конкатеновану конструкцію:

```
*/$END
end;.
```

Тоді вираз для Y' набуде наступного вигляду:

$$Y' = f_2(Y) = 'CREATE...' \| chr(10) \| 'begin' \| chr(10) \| 'null;' \| chr(10) \| \| '\$IF\ false\ \$THEN\ /*' \| Y \| chr(10) \| '*/\$END' \| 'end;'. \quad (3.17)$$

Щоб підвищити здатність протистояти атакам повного перебору, частотного аналізу, пропонується декілька модифікувати запропонований метод маскування (схема МР-2а). А саме, для маскування вихідних кодів програм, що зберігаються, використовувати не тільки метод перестановки, а й метод заміни (поліалфавітний шифр). При цьому до алфавіту символів заміни пропонується включити символи зі стандарту *Unicode* (наприклад, з областей від U+0000 до U+007F (символи набору ASCII) та від U+0400 до U+052F).

У вдосконаленому методі після відповідної випадкової перестановки символів коду програми, що зберігається, пропонується кожен отриманий символ замінювати на відповідний символ, який вибирається випадковим чином з обраного алфавіту. З цією метою використовується ГПВЧ, що формує послідовності випадкових чисел з урахуванням початкового значення $X_{R_j}^0$. У загальному випадку це може бути інший ГПВЧ ($PRNG_{sub}$), ніж той, який використовується для перестановки ($PRNG$). В цьому випадку до відношення R^{secret} слід додати ще два атрибути: $PRNG_{sub}$ – номер вибраного зі списку ГПВЧ для процедури заміни та s_s – ознака необхідності використовувати метод заміни для маскування вихідних кодів визначених PSM.

Для виконання заміни достатньо скористатися перетворенням такого виду:

$$index(Y^i) = (index(A^i) + r^i) \bmod n, \tag{3.18}$$

де $index(A^i)$ – порядковий номер у таблиці підстановки R_{sub} A^i-го символу коду збереженої програми, отриманого після відповідної випадкової перестановки ($A^i = R_{sub}[index(A^i)]$, $i = 1..n$); n – кількість елементів таблиці R_{sub} (довжина (потужність) алфавіту, що включає символи стандарту *Unicode* із зазначених вище областей); r^i – i-й елемент послідовності випадкових чисел ($r^i \in \mathbb{N}^*_{<n} = \{m \in \mathbb{N}^* \mid m < n\}$), що формується відповідним ГПВЧ, з урахуванням початкового значення $X_{R_j}^0$; $index(Y^i)$ – порядковий номер у таблиці підстановки перетвореного ($Y^i = R_{sub}[index(Y^i)]$) після відповідної заміни символу.

Перш ніж перейти до розгляду загальної схеми процесу маскування коду PSM, доречно відзначити, що дане рішення - використовувати метод заміни на додаток до методу перестановки, можна використовувати і для маскування елементів даних (якщо перетворення підлягають поля, що не відносяться до типу даних BLOB) в розглянутому вище (п. 3.2) методі. При цьому до алфавіту символів заміни пропонується включити символи залежно від типу даних.

Таким чином, з урахуванням викладеного вище, загальну схему процесу маскування коду програм, що зберігаються, можна представити наступним чином (рисунок 3.12). Де як функція $f(Y)$ може виступати функція $f_1(Y)$ або $f_2(Y)$.

У Додатку В.6 наведено приклади отриманого перетвореного коду деякої збереженої процедури DEMO у разі використання при маскуванні генератора випадкових чисел Xorshift з періодом 2^{128}-1 без заміни символів вихідного коду після їх перестановки (приклад В.6.1, Додаток В.6) і із заміною символів (приклад В.6.2, Додаток В.6).

Masking process P (MP-P)

Input: $name^{table}$, $name^{j}$, *type*, *iend*, *ibeg*
Output: transformed value of SP code – Y'

```
 1: Select A, K3^i From name^table where Nm^j = name^j
 2: Decrypt(R^secret[name^table, name^j, type]) → (K1^R, K2^j, hash, PRNG, PRNG_sub, z_per, k_Σ, l, s_s)
 3: X^0_Rj = hash(K1^R + K2^j − K3^i)
 4: switch(PRNG)
 5: {case 1: linear congruential generator (LCG)
 6:  case 2: built-in random number generator (package DBMS_RANDOM)
 7:  case 3: Xorshift pseudo random number generator
 8:  ...
 9:  case Θ:...}
10: for k = 1 to z_per          /* number of repetitions of permutations */
11:  for i = l-iend downto ibeg
12:   j=random_PRNG(ibeg..i) /* a random number is generated in the range [ibeg,i]*/
13:   swap(A[i], A[j])      /* exchange */
14:  end for
15: end for
16: if s_s then          /* sign of substitution */
17: switch(PRNG_sub)
18: {case 1: LCG
19:  case 2: built-in random number generator (package DBMS_RANDOM)
20:  case 3: Xorshift pseudo random number generator
21:  ...
22:  case Θ:...}
23:  for i = 1 to n
24:  r^i =random_PRNG_sub (1..n)
25:   index(Y^i) = (index(A^i) + r^i) mod n,
26:   Y^i = R_sub[index(Y^i)]
27:  end for
28: else Y = A
29: end if
30: Y' = f(Y)
```

Рисунок 3.12. Схема процесу маскування коду PSM (MP-P)

3.3.2. Характеристика основних етапів зворотного до маскування процесу

Щоб відновити вихідний код замаскованої програми законний користувач (або процес) залежно від дій, виконаних при маскуванні, спочатку повинен або виключити один доданий рядок (застосування функції $f_1(Y)$ – вираз (2.16)), або чотири перші рядки та два останні (застосування функції $f_2(Y)$ – вираз) (3.17) – виділені кольором у прикладах В.6.1, В.6.2 Додатку В.6). Потім, якщо виконувалася заміна символів, потрібно виконати процедуру зворотну до неї. Для знаходження зворотної перестановки необхідно визначити початкову перестановку, що використовувалася при маскуванні, і далі відновити код вихідної програми, що зберігається.

В результаті, послідовність виконуваних операцій буде наступною.

1. Витягуються з таблиці БД, де зберігаються PSM потрібна збережена програма (її код):

SELECT A, K_3^i FROM $name^{table}$ WHERE $Nm^j = name^j$.

2. Виключаються додані у процесі конкатенації «зайві» рядки перетвореного коду збереженої програми Y': $Y = F(Y')$.

3. Витягуються та розшифровуються відповідні дані з таблиці R^{secret}:

$$Decrypt(R^{\sec ret}[name^{table},\ name^{j}, type]) \to (K_1^R, K_2^j, hash, PRNG, PRNG_{sub}, z_{per}, k_\Sigma, l, s_s).$$

4. Обчислюється довжина коду збереженої програми Y ($length(Y)$). Якщо $length(Y) \neq l$ то подальші дії припиняються.

5. Формується початкове значення $X_{R_j}^0$ (таке саме як у МР-Р).

6. Підготовляється масив π_{num}.

7. Якщо виконувалася процедура заміни Y, виконується зворотне до неї перетворення для відновлення символів, що входять у вихідну збережену програму:

$$index(A^i) = (index(Y^i) - r^i) \bmod n, \qquad (3.19)$$

$$A^i = R_{sub}[index(A^i)]. \quad (3.20)$$

8. Визначається первісна перестановка.

Завдяки можливості повторення послідовності чисел, що формуються ГПВЧ, з одного й того самого початкового значення, виконуючи дії відповідно до алгоритму Фішера-Йейтса, отримаємо первісну перестановку: $\pi_{num} = (\pi_{num}(1), \pi_{num}(2), ..., \pi_{num}(l))$, аналогічну, отриманої при маскуванні (MP-P).

9. На підставі отриманої первісної перестановки $\pi_{num} = (\pi_{num}(1), \pi_{num}(2), ..., \pi_{num}(l))$ та рядка даних $Y(i)$ (якщо виконувалася процедура заміни: $Y(i) = A(i)$), виконується зворотна перестановка відповідно до виразу:

$$X[\pi_{num}(i)] = Y(i),\ i \in \{1, ..., l\} \quad (3.21)$$

10. Обчислюється контрольна сума ($hash(X)$) відновленого PSM коду, яка порівнюється з еталонним значенням k_Σ з таблиці R^{secret}. Якщо $hash(X) = k_\Sigma$, тоді здійснюється компілювання відновленого PSM коду, і його подальше використання. В іншому випадку видається повідомлення про неможливість використання відновленої збереженої програми.

Загальну схему процесу зворотного до маскування коду програм, що зберігаються, можна представити наступним чином (рисунок 3.13).

Як показує практика, відновити код вихідної збереженої програми без знання ключів та решти інформації, що зберігається в таблиці R^{secret}, задача важко розв'язна (про що згадувалося в п. 3.3.1). Авторизований же користувач з відповідними привілеями відновити код замаскованого PSM може дуже швидко. Залежно від довжини PSM та продуктивності сервера, цей час становить, як правило, частки або одиниці мілісекунд (у найгірших результатах він не перевищує одиниць секунд).

Inverse masking process P (IMP-P)

Input: $name^{table}$, $name^{j}$, *type*, *iend*, *ibeg*
Output: source (not masked) value - X

1: Select A, K_3^i From $name^{table}$ where $Nm^j = name^j$
2: Deletion of unnecessary lines from Y' ($Y = F(Y')$)
3: Decrypt($R^{secret}[name^{table}, name^j, type]$) $\rightarrow$ (K_1^R, K_2^j, *hash*, *PRNG*, $PRNG_{sub}$, z_{per}, k_Σ, l, s_s)
4: **if** *length*(Y) $\neq l$
5: notice of termination of further operations
6: *exit*
7: **end if**
8: $X_{R_j}^0 = hash(K_1^R + K_2^j - K_3^i)$
9: **for** i = 1 **to** l
10: $\pi_{num}[i] = i$ /* array preparation */
11: **end for**
12: **if** s_s **then** /* sign of substitution */
13: **switch**($PRNG_{sub}$)
14: **{case** 1: LCG
15: **case** 2: built-in random number generator (package DBMS_RANDOM)
16: **case** 3: Xorshift pseudo random number generator
17: ...
18: **case** Θ:...**}**
19: **for** i = 1 **to** n
20: r^i=random_$PRNG_{sub}$(1..n)
21: $index(A^i) = (index(Y^i) - r^i) \bmod n$,
22: $A^i = R_{sub}[index(A^i)]$
23: **end for**
24: $Y = A$
25: **end if**
26: **switch**(PRNG)
27: **{case** 1: linear congruential generator
28: **case** 2: built-in random number generator (package DBMS_RANDOM)
29: **case** 3: Xorshift pseudo random number generator
30: ...
31: **case** Θ:...**}**
32: **for** k = 1 **to** z_{per}
33: **for** i = l-iend **downto** ibeg /* getting the initial permutation */
34: j=random_PRNG(ibeg..i)
35: swap($\pi_{num}[i]$, $\pi_{num}[j]$)
36: **end for**
37: **for** i = 1 **to** l /* inverse permutation */
38: $X[\pi_{num}[i]] = Y(i)$
39: **end for**
40: **end for**
41: **if** $hash(X) = k_\Sigma$
42: compiling a restored stored program
43: **else**
44: give a message about the impossibility of using the restored PSM
45: **end if**

Рисунок 3.13. Схема процесу зворотного до маскування коду PSM (IMP-P)

3.3.3. Оцінка можливості використання шифрування для приховування вихідного коду збережених програм

Для приховування коду програм можна використовувати і шифрування. Наприклад, при реалізації БД та відповідних PSM в СКБД Oracle можна скористатися вбудованими пакетами звичайного (без збереження формату) шифрування. Але як зазначалося вище, використання стандартного шифрування призводить до зміни формату та збільшення розмірності вихідних даних. Наприклад, для деякої збереженої процедури, розмір якої до шифрування становив 92662 символи, після процедури шифрування збільшився до 218676 символів, на відміну від методу маскування, викладеного вище, при використанні якого розмір перетвореного коду збереженої процедури залишався незмінним (92662 символи). Аналогічно для іншої збереженої процедури, розмір якої до шифрування становив 3989 символів, після процедури шифрування збільшився до 8054 символів і т. д. Тобто є збільшення числа символів більш ніж у 2 рази.

Крім того, як вказувалося в п. 3.3.1, отриманий після перетворення (шифрування) код збереженої програми необхідно зберегти на сервері. І зробити це можна лише після виконання відповідних операторів мови SQL. Тому отриманий результат шифрування необхідно об'єднати шляхом конкатенації з конструкцією `CREATE` (вирази (3.16), (3.17)). Але попередньо необхідно виконати нетривіальне перетворення рядка байт у рядок символів – формування CLOB (при розшифруванні – навпаки). До того ж, якщо розмір зашифрованої процедури перевищує 4000 символів у СКБД Oracle, при її компіляції видається помилка і автоматично додається символ зміни / переведення рядка (рисунки 3.14, 3.15), а отже, змінюється розмір зашифрованих даних, що призводить до помилки розшифрування. Щоб такого не відбувалося, потрібна додаткова обробка отриманого перетвореного коду. Все це призводить до збільшення часу на процедури зашифрування / розшифрування.

```
select line, TEXT, DBMS_LOB.getLength(TEXT) from USER_SOURCE
where type = 'PROCEDURE' and name = 'TEST_SET_OUT_UMD_VARIANT'
```

	LINE	TEXT	DBMS_LOB.GETLENGTH(TEXT)
1	1	procedure TEST_SET_OUT_UMD_VARIANT (a varchar2) as	51
2	2	CCC2D005149C40C56021DF7F759AAE2B1EA1B3FCF4C8827D659EC518 A3AD0AFD521F7974E33B864121A6F472E05700367F497DBAD80B79C7	4000
3	3	3436CD00420D172A45A13B2AFAEA12F230DF91FB6AB5CB65F5DBE24C FED0999D7950FF5871296603C92B4D410DC2DEE60CEFA32867FD63994	4000
4	4	314E0618F33AB862D1DBB1E4DA8F9B19F98217C5268E6466B05660E43 E382539DDAB5BFA91A6F24608C8B85375318970F3C1AA41841F0451B9	4000
5	5	387265E83A39970C22B8A40FA721066665FD32259A359E0641F65D177 C22B2AD1122BE601DC1D162DD2089EFF0261D2A5E95C882EA0E81493	4000
6	6	809194C216C454E36DF6277D6189F2DC8302B4FB538BDCF17ECD47AA 1A30F7C8FE1ED4F78B9CACFB4255D5065D63CFFA83BC91FFBB57267A7	4000
7	7	9D259C88B351B03B83CF4696DEC0104EB1D4D34CAB997CDFBA0E4583 33F6AE0D1FCDF5D302CCAB60685D9FBA57E931C62370A67BD6F691BE	4000
8	8	912A6A8A7B1FED3B6DFD5DBCBB4859326895982355B9471442D013D6 882BC1BC41117FFA85E1339278866BA8CF47700D762DEB2ED13D9B836	4000
9	9	33DA3A59CA6A1B4EFCB07DB868534534FD7232F3C6202D39AFF8D766 0E2E2B3F57224EB763966BDAF0CF43DD0BD856433AFE86635C00AA76	4000
10	10	137EEBA3799B790DFBD1FF65E6EEC5725F1ED14EACA6E4FAFFB6837F5 D84B8350DA343C05359B5E0AA6A4B366B8E5CE3085017298932F38736	4000
11	11	F22DE1EF03C88BF87F73EDD072645411EF43F994CB3737CC49601B8D1E 3F296D62A67FD781504CC218DB38C406A222EDB59F7A9E24124D5D85	4000
12	12	24594F47C79EE094646CF41E927BD1BE5622F77131E5750D6476FE692C B4A988DDA1E122364D2E2138C5DDEDA7ABBF652F25B79EA5630978E	4000

Рисунок 3.14. Вид (уявлення) перетвореної після шифрування збереженої процедури

```
select line, TEXT, DBMS_LOB.getLength(TEXT) from USER_SOURCE
where type = 'PROCEDURE' and name = 'TEST_SET_OUT_UMD_VARIANT'
```

	LINE	TEXT	DBMS_LOB.GETLENGTH(TEXT)
1	1	procedure TEST_SET_OUT_UMD_VARIANT (a varchar2) as	51
2	2	CCC2D005149C40C56021DF7F759AAE2B1EA1B3FCF4C8827D659EC518 A3AD0AFD521F7974E33B864121A6F472E05700367F497DBAD80B79C7	4000
3	3		1
4	4	3436CD00420D172A45A13B2AFAEA12F230DF91FB6AB5CB65F5DBE24C FED0999D7950FF5871296603C92B4D410DC2DEE60CEFA32867FD63994	4000
5	5		1
6	6	314E0618F33AB862D1DBB1E4DA8F9B19F98217C5268E6466B05660E43 E382539DDAB5BFA91A6F24608C8B85375318970F3C1AA41841F0451B9	4000
7	7		1
8	8	387265E83A39970C22B8A40FA721066665FD32259A359E0641F65D177 C22B2AD1122BE601DC1D162DD2089EFF0261D2A5E95C882EA0E81493	4000
9	9		1
10	10	809194C216C454E36DF6277D6189F2DC8302B4FB538BDCF17ECD47AA 1A30F7C8FE1ED4F78B9CACFB4255D5065D63CFFA83BC91FFBB57267A7	4000
11	11		1
12	12	9D259C88B351B03B83CF4696DEC0104EB1D4D34CAB997CDFBA0E4583 33F6AE0D1FCDF5D302CCAB60685D9FBA57E931C62370A67BD6F691BE	4000
13	13		1
14	14	912A6A8A7B1FED3B6DFD5DBCBB4859326895982355B9471442D013D6 882BC1BC41117FFA85E1339278866BA8CF47700D762DEB2ED13D9B836	4000
15	15		1
16	16	33DA3A59CA6A1B4EFCB07DB868534534FD7232F3C6202D39AFF8D766	4000

Рисунок 3.15. Вид (уявлення) перетвореної після шифрування та компіляції збереженої процедури

3.4. Висновки за розділом

1. В результаті проведених досліджень було запропоновано два методи, що ускладнюють реалізацію зловмисником загрози логічного висновку, в основу яких було покладено принципи динамічного маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних. З тією лише особливістю пропонованого підходу динамічного маскування, що у виробничій базі даних проводиться попередня фізична зміна чутливих (конфіденційних) даних, але при цьому є можливість у разі необхідності навести всі зміни у вихідний стан користувачем, що має на це відповідні права. Це вигідно відрізняє пропонований механізм від більшості типових комерційних інструментів маскування критичних даних. При цьому в першому методі для маскування відповідних чутливих елементів даних використовуються математичні перетворення на основі обчислення операцій за модулем, а в другому – перетворення на основі принципів випадкової перестановки елементів даних різного типу, що спирається на сучасний варіант алгоритму тасування Фішера-Йейтса, з можливістю використання методу заміни (на додаток до методу перестановки) з метою ускладнення статистичного аналізу. Другий метод дозволяє при мінімальних часових, обчислювальних витратах і необхідних при цьому даних перетворювати критичні дані до правдоподібного виду з можливістю їх відновлення.

При реалізації даних методів автентифікований користувач з відповідними правами отримує доступ до чутливих (конфіденційних) даних за рахунок можливості здійснити перетворення (перезапис) запиту «на льоту», а зловмисник навіть, використовуючи складні чи послідовності простих логічно пов'язаних запитів, обмежений у реалізації загрози логічного висновку за прийнятний йому час, у зв'язку з тим, що йому доступні лише збережені в базі замасковані дані.

2. Метод маскування, який використовує математичні перетворення на основі обчислення операцій за модулем, доцільно використовувати при невеликих розмірностях чутливих елементів даних.

3. Метод маскування на основі принципів випадкової перестановки елементів даних різного типу, з можливістю використання методу заміни може бути використаний для приховування чутливих елементів даних різного типу (числових, символьних рядків, великих бінарних, довгих символьних об'єктів), причому з можливістю виконувати відповідні перетворення як над повним значенням, так і над його часткою. Це є актуальним з одного боку для маскування таких даних, як банківські, дисконтні картки, телефонні номери, серійні номери техніки, автомобільні номери тощо, а з іншого боку дозволяє скоротити кількість операцій перетворення великих бінарних об'єктів, а отже, і час реалізації їх маскування практично без втрати ефективності приховування. Відмінною особливістю даного методу є підхід до процесу перемішування даних, а саме перемішування елементів значення даних всередині потрібного поля рядка. При цьому основні операції запропонованого методу можуть знайти застосування для вертикального перемішування – перестановки значень всередині стовпця обраної таблиці. Пропонований метод може бути використаний як у виробничих, так і в невиробничих базах даних, розширюючи можливості так званого статичного маскування даних. Проведені дослідження показали, що використання запропонованого методу для вирішення задачі протидії загрозі логічного виводу є кращим ніж класичне шифрування, так як його застосування не призводить до зміни формату та збільшення розмірності даних, що зберігаються, а час, що витрачається на саме перетворення (маскування) даних приблизно на (10-17) % менше необхідного часу на шифрування цих самих даних.

4. При довжині рядка менше трьох символів або, якщо всі символи однакові, маскування із застосуванням методу перестановки елементів даних поля рядка деякої таблиці є недоцільним. У цьому випадку розумніше використовувати або метод підстановки, або метод маскування даних на основі обчислення операцій за модулем (якщо допускається зміна типу або розмірності даних). У деяких випадках можливе збільшення розмірності елементів, що маскуються, даних поля до заданої величини шляхом додавання випадкових символів з обраного алфавіту (або отриманих на основі семантичного аналізу ПрО, що розглядається), до яких застосовуються аналогічні перетворення (природно, кількість доповнюваних

символів має бути визначена, і вони після зворотного до маскування перетворення повинні бути видалені).

5. Завдяки розробленому методу маскування коду програм, що зберігаються у відповідних кортежах певного атрибуту деякої системної таблиці бази даних, в основу якого покладено принцип випадкової перестановки символів коду з можливою заміною кожного такого символу на інший випадково вибраний символ із стандарту Unicode, забезпечується більш ефективне приховування коду постійних збережених модулів, тобто таке приховування, яке вимагає значно більших обчислювальних, а отже, і часових витрат, ніж при використанні існуючих способів, що надаються розробниками деяких сучасних СКБД. Для гарантованості цілісності відновленого коду збережених програм (виявлення можливих несанкціонованих дій щодо його зміни, незалежно від того, зроблено це було помилково або навмисно зловмисником, шкідливою програмою) у запропонованому методі використовуються процедури порівняння його довжини та контрольної суми з еталонними значеннями, що зберігаються в зашифрованому вигляді у R^{secret}. Це дозволяє переконатися в незмінності отриманого в результаті зворотного маскування перетворення коду конкретного PSM і гарантує можливість його використання без ризику виконання ним будь-яких недокументованих (шкідливих) дій.

6. Щоб не допустити / знизити ймовірність розкриття (компрометації) закритих ключів K_1^R, K_2^j, в реалізації методу передбачається їх (як і деяких інших даних відношення R^{secret}) шифрування криптостійким алгоритмом. При цьому значення ключа, яким зашифровується таблиця R^{secret} у відкритому вигляді, ніде не показується і не допускається його відстеження через засоби документування виконаних запитів (історії запитів). Витягується це значення спеціальним ПЗ сервера СКБД із стегоконтейнера, який пред'явить під час відкриття сеансу автентифікований користувач із відповідними привілеями. Всі інші користувачі, навіть привілейовані, без знання цього ключа, витягти закриті ключі та інші важливі для процесів приховування та відновлення дані з R^{secret} не зможуть, а, отже, не зможуть відновити вихідні дані.

Результати, представлені у розділі, опубліковані у роботах [47, 59, 202].

4. РОЗРОБКА МЕТОДУ КОНТРОЛЮ ЦІЛІСНОСТІ ТА СПРАВЖНОСТІ ЗБЕРЕЖЕНИХ ПРОГРАМ, ЗАСНОВАНОГО НА МОЖЛИВОСТЯХ ТЕХНОЛОГІЇ БЛОКЧЕЙН

У попередньому розділі для гарантованості цілісності відновленого після маскування коду збереженої програми (виявлення можливих несанкціонованих дій щодо його зміни) використовувалася процедура порівняння його довжини та контрольної суми з еталонними значеннями. При цьому дуже важливим є і постійний моніторинг цих об'єктів бази даних, оскільки деякі з атак на БД і не тільки на неї (наприклад, можна атакувати операційну систему через вразливості сервера БД) можуть бути виявлені саме на основі аналізу зміни (умисної чи випадкової) PSM (порушення їх цілісності, автентичності), або їхнього набору (збільшення чи зменшення їхньої кількості) на сервері БД.

Далі в роботі для моніторингу змін, що відбуваються (відрізняючи поточні та несанкціоновані) в критичних для користувача і системи кодах всіх PSM сервера БД буде розглянутий метод, заснований на можливості парадигми блокчейн (англ. blockchain).

Блокчейн заснований на перевіреній криптографічній технології, що гарантує цілісність та доступність даних. Блокчейн забезпечує й інші параметри безпеки, такі, як невідмовність (англ. non-repudiation) та автентичність, оскільки всі операції захищені закритими ключами і цифровими підписами. Незмінність, ще одна найважливіша риса блокчейна. Як тільки запис внесено до блокчейну, змінити його не можна. З технологічної та бізнес-прикладної точки зору існують різні типи блокчейнів, наприклад, такі, як [203]: публічні блокчейни (англ. public blockchains); приватні блокчейни (англ. private blockchains); напівприватні блокчейни (англ. semiprivate blockchains); закритий розподілений реєстр (англ. permissioned ledger); реєстр, що розділяється (англ. shared ledger); повністю приватні та пропрієтарні блокчейни (англ. fully private and proprietary blockchains); токенізовані блокчейни (англ. tokenized blockchains); нетокенізовані блокчейни (англ. tokenless blockchains) та деякі інші.

В випадку, що розглядається, більшою мірою інтерес представляє *закритий розподілений реєстр* – блокчейн, де учасники мережі вже знають один одного і довіряють один одному. Тобто блокчейн із фіксованим списком учасників з обмеженими правами доступу. Такий тип блокчейнів прагнуть використати великі компанії, що підтримують складні бізнес-процеси. У закритих розподілених реєстрах можна обійтися без механізму розподіленого консенсусу. Замість нього використовується протокол угоди, за допомогою якого фіксується загальновизнана істина про стан записів у блокчейні. В даному випадку вважається, що всі верифікатори вже визначені центральним органом для верифікації транзакцій у ланцюжку, і, як правило, механізм майнінгу не потрібний.

4.1. Визначення структури первинного блока

Відповідно до принципів організації блокчейна, як збудованого за певними правилами безперервного послідовного ланцюжка блоків, кожна мережа блокчейнів повинна мати перший, так званий *первинний* опублікований незмінний надалі *блок* (*genesis block*), до якого згодом додається наступний блок на основі узгодженої моделі консенсусу [204]. Як правило, *genesis block* реалізуються програмним способом. І оскільки він позбавлений фундаментальної основи безпеки блокчейна, що полягає у пов'язаності з попередніми блоками, він повинен бути опрацьований особливо ретельно. Тому, проаналізувавши структури подібних блоків для різних систем блокчейнів [203, 205] та способи їх реалізації, враховуючи все вищесказане, було запропоновано первинний блок наступної структури з описом його компонентів, наведеної в таблиці 4.1. Даний блок найдоцільніше формувати відразу ж після інсталяції обраної СКБД на задану платформу з прив'язкою до відповідного часу.

Приклад genesis block та лістинг програми для визначення значень його полів наведено у Додатку Г.1.

Таблиця 4.1 – Структура genesis block

Поле	*Опис*
nonce	128-бітове випадкове число, згенероване криптографічно сильним генератором псевдовипадкових чисел.
i_{id}	Число, отримане в результаті перетворення i_{id} =*nonce*(mod 2^{32}).
ver_db	Назва компонента та номер версії СКБД (номери версій основних компонентів бібліотеки у базі даних).
pl_name	Платформа, на якій встановлена СКБД.
h_name	Ім'я хост-машини, на якій встановлена СКБД.
d_name	Доменне ім'я бази даних.
timestamp	Час створення (мітка часу) блоку - інсталяції СКБД. Як час використовується Всесвітній координований час (Coordinated Universal Time – UTC).
h_{gen}	Геш-значення, що обчислюється відповідно до виразу (4.1).

$$h_{gen} = hash(hash(nonce) \,||\, hash(timestamp) \,||\, hash(ver\_db) \,||\, hash(pl\_name) \,||\, hash(h\_name) \,||\, hash(d\_name)), \quad (4.1)$$

де $hash$ – одна з криптографічних геш-функцій (таких як: MD5, SHA-1, SHA-256, SHA-384, SHA-512, SHA-3), яка може застосовуватися підряд кілька разів (наприклад, $hash(h_{gen}) = SHA256(SHA256(h_{gen}))$).

4.2. Визначення структури блоків

На підставі аналізу структур блоків блокчейнових ланцюжків відомих систем [203-206], взявши за основу контейнерну структуру даних, що об'єднує транзакції для включення їх до загальнодоступного реєстру системи Bitcoin, з урахуванням специфіки розглянутого рішення була визначена структура інших блоків блокчейнового ланцюжка, наведена у таблиці 4.2.

Таблиця 4.2 – Структура блоків

Поле	**Опис**
Заголовок	
h_{p_block}	Геш заголовка попереднього блоку*.
h_{block}	Геш заголовка поточного блоку (b_header=($h_{p\_block}, i_{id}, timestamp, h_{root}$)), що обчислюється відповідно до виразу (4.2).
h_{root}	Геш кореня дерева Меркла для всіх PSM у блоці.
i_{id}	Одноразовий номер, інкремент від i_{id} попереднього блоку: $i_{id} = i_{id} + 1$ (через те, що не потрібен майнінг).
timestamp	Час створення (мітка часу) блоку. В якості часу використовується Coordinated Universal Time (UTC).

Тіло блоку	
d_{DB}	Доменне ім'я бази даних.
n_{DB}	Ім'я бази даних
n_{sh}	Ім'я схеми (колекції логічних структур даних чи об'єктів) бази даних.
α_k	Ім'я k-го модуля, що постійно зберігається (SO_k, $k = 1..n_{so}$).
p_k	Тип k-го модуля, що постійно зберігається (PSM): процедура, функція, пакет, тіло пакету, тригер, власний тип даних, вихідний код Java-об'єктів, уявлення ($p_k \in type_{sp} = \{package, package\ body, procedure, function, trigger, type, java\ source, view\}$).
h_k	Значення геша k-го PSM, яке обчислюється відповідно до деякої функції.
n_{so}	Кількість контрольованих PSM.
w	Підпис даних (геша від конкатенації h_{root} та геш значень *timestamp*, n_{so}) одним із сучасних алгоритмів цифрового підпису ($Sing$) творцем j-ої схеми БД (вираз (4.3)).

* – для блоку наступного за *genesis block*: $h_{p\_block} = h_{gen}$.

$$h_{block} = hash(b\_header) = hash(h_{p\_block} \| i_{id} \| timestamp \| h_{root}). \quad (4.2)$$

$$w \leftarrow Sing_{sk^j}\left(hash(h_{root} \| hash(timestamp) \| hash(n_{so}))\right), \quad (4.3)$$

де sk^j – особистий ключ творця j-ої схеми БД.

У загальному вигляді структуру блокчейна для пропонованого методу можна представити так (рисунок 4.1).

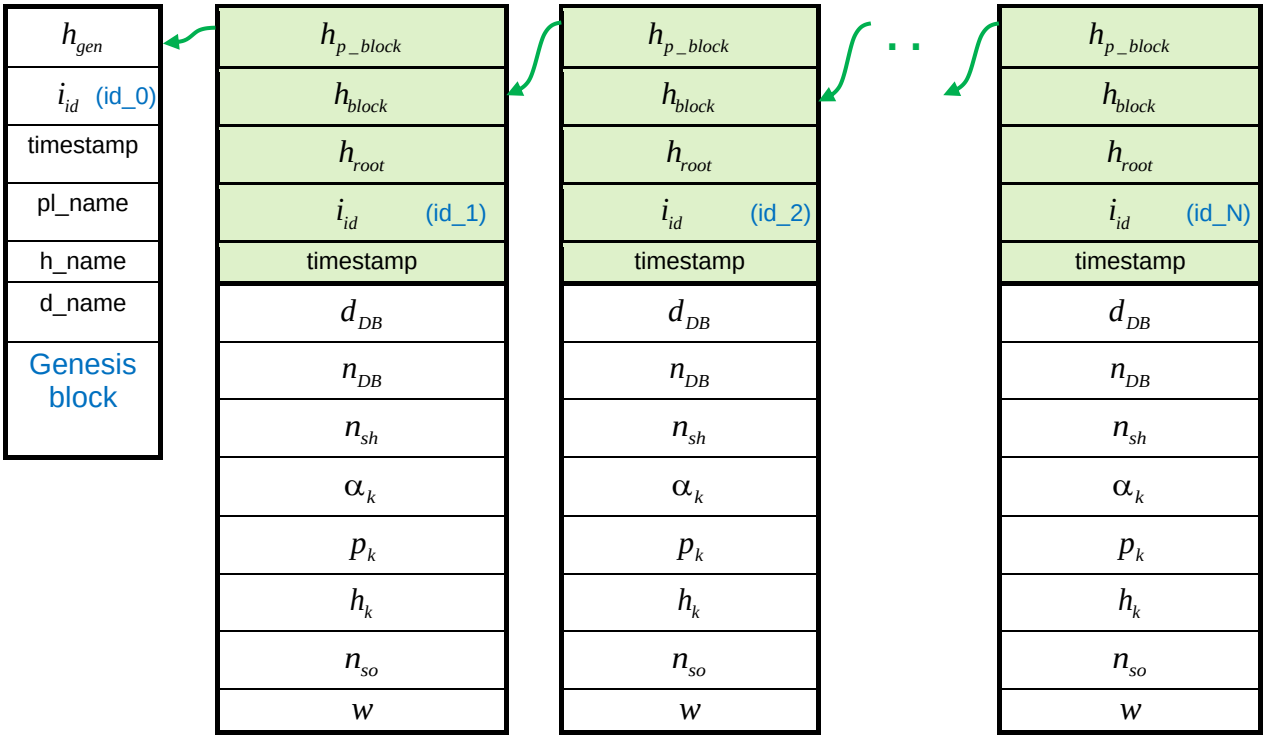


Рисунок 4.1. Структура блокчейна

Приклад вмісту блока id_1 наведено у Додатку Г.2.

4.3 Методи знаходження кореневого значення геш-дерева

Доцільність використання структури геш-дерева, кореня Меркла, механізму цифрового підпису для контролю цілісності та справжності збережених у конкретній схемі БД (n_{sh}) об'єктів у запропонованому методі контролю цілісності та справжності збережених програм обумовлена об'єктивною потребою раціонального витрачання ресурсів, що веде до економії обсягу збережених для цього даних та обчислювального ресурсу процесора. А саме, для того щоб здійснити контроль цілісності та автентичності конкретного PSM у конкретній схемі БД звичайним способом потрібно було б виконати процедури гешування та цифрового підпису із збереженням відповідних даних для кожного з них. До того ж, щоб перевірити цілісність усіх цих даних, верифікатор повинен мати (зберігати) всі їх геш-значення. Використання ж структури геш дерева дозволяє забезпечити контроль цілісності не тільки конкретного PSM, що перевіряється, але і всіх інших збережених PSM обраної схеми БД. Так як цей один фрагмент даних входить у загальну структуру і зміна хоча б одного біта в ньому спричинить повну зміну значення кореня дерева Меркла. Крім того, у багатьох випадках кінцевий користувач, володіючи обмеженими обчислювальними ресурсами, за наявною в його розпорядженні інформацією (склад якої буде показаний нижче) не зможе самостійно перевірити цілісність коду. Тому було прийнято рішення щодо розробки більш прийнятного способу верифікації. Концепція геш-дерев була запропонована та запатентована [207] Р. Мерклом. Меркл запропонував використовувати повні бінарні дерева (англ. complete binary trees) для створення множинних одноразових підписів [208]. Сьогодні у багатьох застосунках, що використовують ідею безпечної перевірки вмісту великих структур даних, застосовуються різні модифікації бінарних дерев (двійкове / бінарне дерево – дерево з не більше двома дітьми для кожного вузла [209]), відповідно до яких визначаються правила обчислення кореневого значення геш-дерева. У

запропонованому методі контролю цілісності і справжності збережених програм розглядаються два можливі підходи до обчислення кореня дерева Меркла, що формується на основі двох типів бінарних дерев, а саме:

– *незбалансованого бінарного дерева*, тобто дерева, у якого кількість листя не є числом ступенем 2 [210]. Про подібні дерева згадується, наприклад, в інформаційному документі Інтернету RFC 6962 [211], що описує експериментальний протокол публічної реєстрації сертифікатів безпеки транспортного рівня, який у певному сенсі дозволяє будь-кому здійснювати аудит діяльності центру сертифікації та помічати видачу підозрілих сертифікатів, а також проводити аудит самостійно;

– *повного бінарного дерева* (англ. *full binary tree*) – такого бінарного дерева, в якому кожен вузол має рівно нуль або двох дітей (іншими словами, кожен вузол є або листом, або має два дочірні елементи [209]). Таке дерево також відомо, як *правильне двійкове дерево* (англ. *proper binary tree*) [212].

Перш ніж розглянути конкретні методи формування дерева Меркла та знаходження його кореневого значення, слід звернути увагу на деякі особливості, що відіграють важливу роль у цьому процесі. По-перше, сила конструкції геш-дерева залежить від сили основної використовуваної хеш-функції ($hash$). Тому рекомендується використовувати одну з криптографічних хеш-функцій (таких як: SHA-1, SHA-256, SHA-384, SHA-512, SHA-3) в якості основи геш-дерева. По-друге, для зменшення колізій (щоб протистояти атаці знаходження другого прообразу [211]) між листовими гешами та внутрішніми гешами доцільно використовувати різні конструкції для хешування листових вузлів та внутрішніх вузлів [210]. А саме, для гешей внутрішнього вузла додається один байт «1» (0x01) перед гешами, що конкатенуються, а для гешей листового вузла додається один байт «0» (0x00) перед кодом збереженого об'єкта БД:

$$h_k = X_k^H = LH(SO_k) = hash(0\text{x}00 \,||\, SO_k),\ k = 1,...,n_{so},, \tag{4.4}$$

$$\begin{gathered} X_i^l = IH(X_{j\_left}^{l+1}, X_{j\_rigth}^{l+1}) = hash(0\text{x}01 \,||\, X_{j\_left}^{l+1} \,||\, X_{j\_rigth}^{l+1})); \\ l = H-1, H-2,...,0;\ \ i = 1,2,...,N_l;\ \ \ j = 1,2,...,\lfloor N_{l+1}/2 \rfloor; N_l \le 2^l; N_{l+1} \le 2^{l+1}; \end{gathered} \tag{4.5}$$

де h_k – значення геша k-го PSM деякої схеми БД (n_{sh}); SO_k – код k-го PSM ($SO_k \in SO$); $IH(...)$ – геш-функція для внутрішнього вузла; $LH(...)$ – геш-функція для листового вузла; H – висота дерева (в даному випадку $H = \lceil \log_2 n_{so} \rceil$ – найменше ціле число, яке більше або дорівнює дійсному числу $\log_2 n_{so}$); $\lfloor N_{l+1} / 2 \rfloor$ – найбільше ціле число, яке менше або дорівнює дійсному числу $N_{l+1} / 2$; l – рівень (глибина) дерева; N_l – кількість вузлів на l-ом рівні; N_{l+1} –кількість вузлів на $(l+1)$-ом рівні; $X^{l+1}_{j\_left}$ – геш значення лівого вузла деякого піддерева на глибині $l+1$, коренем якого є i-й вузол (X^l_i) на глибині l; $X^{l+1}_{j\_rigth}$ – геш значення правого вузла деякого піддерева на глибині $l+1$, коренем якого є відповідний i-й вузол на глибині l.

Якщо $X^{l+1}_{j\_rigth} = null$, тобто $X^{l+1}_{j\_left}$ не має брата на відповідному рівні, то можливі два варіанти дій, або вузол $X^{l+1}_{j\_left}$ просувається вгору по дереву без гешування доти, поки для нього не буде знайдений брат, або геш останнього лівого вузла на відповідному рівні дублюється ($X^{l+1}_{j\_rigth} = X^{l+1}_{j\_left}$) і геш значення внутрішнього вузла обчислюється відповідно до виразу (4.5).

На рисунку 4.2 з урахуванням зроблених зауважень схематично представлений зв'язок між тілом блока, деревом Меркла та заголовком блока.

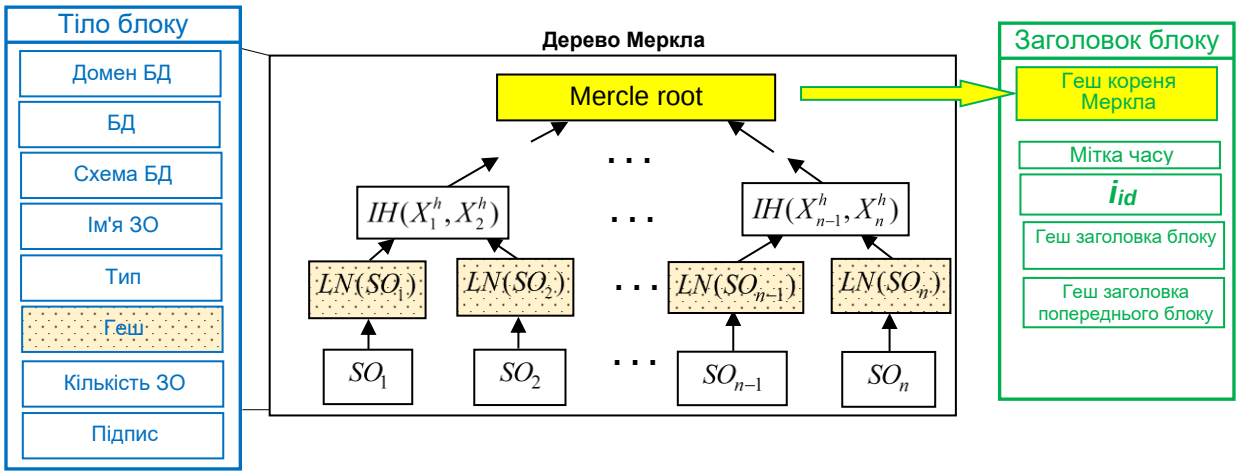


Рисунок 4.2. Зв'язок між тілом блока, деревом Меркла та заголовком блока

4.3.1. Метод обчислення кореня дерева Меркла, що формується на основі незбалансованого бінарного дерева

Оскільки число PSM схеми БД ($SO_k, k=1,...,n_{so}$) у загальному випадку довільне, геш-дерево, що формується, може бути незбалансованим бінарним деревом. В цьому випадку дерево будується в такий спосіб. Спочатку впорядковуються за алфавітом (залежно від імені та типу) PSM (SO), що зберігаються в конкретній схемі, , що розглядається, БД (n_{sh}). Потім відповідно до виразу (4.4) обчислюються геші цих упорядкованих PSM. Геш останнього непарного (якщо $n_{so} \bmod 2 = 1$) листа просувається вгору без змін (без гешування) по дереву доти, доки для нього не буде знайдено брата (лівий вузол). Аналогічні дії виконуються і для проміжних значень геш-функції внутрішніх вузлів, які не мають значення брата (правого вузла), з яким вони можуть бути конкатеновані. Для решти внутрішніх вузлів значення гешів розраховуються відповідно до виразу (4.5). Процес повторюється доти, доки не буде отримано єдиний геш – корінь дерева Меркла (h_{root}). Як приклад розглянемо геш-дерево, сформоване з урахуванням п'яти PSM: $X_1^3 = LN(SO_1)$, $X_2^3 = LN(SO_2)$, $X_3^3 = LN(SO_3)$, $X_4^3 = LN(SO_4)$, $X_5^3 = LN(SO_5)$ (рисунок 4.3).

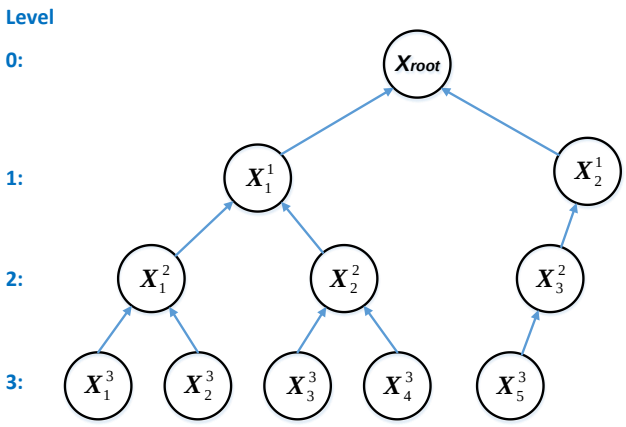


Рисунок 4.3. Незбалансоване геш-дерево

У наведеному прикладі X_5^3 (листовий вузол) не має брата, з яким можна було б виконати операцію конкатенації. Тому X_5^3 просувається (проштовхується)

вгору по дереву без відповідного хешування ($X_3^2 = X_5^3;\ X_2^1 = X_3^2 = X_5^3$) доти, доки воно не буде зчеплене (конкатеноване) зі значенням $X_1^1 = IN(X_1^2, X_2^2) = hash(0x01 \| X_1^2 \| X_2^2)$, де $X_1^2 = IN(X_1^3, X_2^3);\ X_2^2 = IN(X_3^3, X_4^3)$. Після чого для отримання кореневого гешу виконується операція конкатенації значень 0x01, X_1^1, $X_2^1 = X_3^2 = X_5^3$ та їх гешування відповідно до виразу (4.5): $X_{root} = IN(X_1^1, X_2^1)$. Послідовність операцій процесу обчислення кореня дерева Меркла у цьому варіанті представлена нижче (рисунок 4.4).

Merkle Root Calculation (MRC-1)

Input: $SO = \{SO_1, ..., SO_k, ..., SO_{n_{sp}}\}$, $hash$, z, n_{sh}

Output: Mercle root - h_{root}

1: Сортування PSM (SO_k) за алфавітом (в залежності від їх імені та типу)
2: $h_k = X_k^{H-1} = LH(SO_k) = hash(0x00 \| SO_k),\ k = 1,...,n_{so}$ // Розрахунок геш-значень для SO_k
3: $cur\_hashes = A_{hash} = [h_1, \ldots, h_k, \ldots, h_{n_{so}}]$ // копіювання геш-значень в масив
4: *parents_hashes* = [] // створення масиву для зберігання батьківських гешів
5: while *cur_hashes*.size()!=1 // цикл: поки масив не буде складатися з 1 елементу
6: { for (i=1; i<=*cur_hashes*.size();i=i+2) // усі геші поточного рівня
7: { *left*=*cur_hashes*[i]
8: $right = \begin{cases} cur\_hashes[i+1], \text{ if } \exists\, cur\_hashes[i+1]; \\ null. \end{cases}$
9: if (right==null) then
10: $X = left$
11: else
12: $X = IH(left, right) = hash(0x01 \| left \| right)$
13: *parents_hashes*.add(X) // додати геш-значення до батьківського масиву
14: }
15: *cur_hashes*=*parents_hashes* // перевизначення масиву
16: *parents_hashes* = [] // очистити масив
17: }
18: h_{root} = *cur_hashes*[1]
19: return h_{root}

Рисунок 4.4. Схема процесу обчислення кореня дерева Меркла (MRC-1)

Де z – кількість (кратність) застосувань поспіль геш-функції; A_{hash} – масив геш-значень кодів PSM (SO_k);*cur_hashes* и *parents_hashes* – масиви хеш-значень $(l+1)$-го (поточного) та l-го рівнів (батьківських вузлів для *cur_hashes*) дерева Меркла відповідно; *size*() – функція (метод), що повертає кількість елементів у масиві; *left* – геш значення $X_{j\_left}^{l+1}$ лівого вузла деякого піддерева на відповідному

рівні $(l+1)$ дерева; *right* – геш значення $X_{j\_rigth}^{l+1}$ правого вузла деякого піддерева на глибині $l+1$.

Середній час розрахунку кореня дерева Меркла для 10 000 PSM відповідно до алгоритму MRC-1 для СКБД Oracle 12.2 с, встановленої на 64-розрядну операційну систему Windows 10 на сервері з процесором Intel (R) Core (TM) i5-8300H CPU 2.30GHz с ОЗУ – 8 ГБ, становить близько 4.5 с. Для прискорення надалі процесу верифікації цілісності збережених у базі даних PSM є можливість збереження відповідної версії дерева як послідовності бітів (у форматі серіалізації). У практичній реалізації для цього була створена таблиця T_{ts}, що є деяким спеціальним журналом, що зберігає дані про вузли дерев для відповідних схем БД, а також про різні посилання на ці вузли, мітки часу додавання цих даних та деяку іншу інформацію. Структура таблиці T_{ts} та характеристика її атрибутів наведені у Додатку Г.3. У разі використання збереженої версії дерева Меркла у форматі серіалізації для 10 000 PSM, середній час верифікації цілісності вибраного PSM становить близько 0.1 с.

4.3.2. Метод обчислення кореня дерева Меркла, що формується на основі повного бінарного дерева

Геш-дерево можна формувати і трохи інакше, а саме, будуючи завжди повне (англ. *full*) бінарне дерево. Якщо кількість SO_k відповідної схеми бази даних непарна, то значення геша останнього з упорядкованого списку PSM, що є крайнім правим листовим вузлом дерева, що формується, дублюється (як це робиться, наприклад, в Bitcoin). Аналогічні дії виконуються і для геш значень внутрішніх вузлів.

Отже, спочатку відповідно до виразу (4.4) обчислюються хеші PSM. Якщо блок (схема БД) містить непарну кількість PSM, то значення геша останньої збереженої програми дублюється. Потім обчислюються геші внутрішніх вузлів відповідно до виразу (4.5). Знову ж таки, якщо кількість отриманих внутрішніх вузлів на відповідному рівні непарна, то значення геша останнього вузла на цьому

рівні дублюється. У формалізованому вигляді значення правого вузла можна визначати так:

$$X_{j\_rigth}^{l+1} = \begin{cases} (X_{2j}^{l+1} \mid j = 1,2,...,\lfloor N_{l+1}/2 \rfloor : N_{l+1} \le 2^{l+1}, N_{l+1} \bmod 2 = 0); \\ (X_{j\_left}^{l+1} \mid N_{l+1} \bmod 2 = 1 \wedge \exists X_{\tau}^{l+1} : \tau = N_{l+1}). \end{cases} \quad (4.6)$$

Отже, правий вузол хеш-дерева у разі завжди присутній.

Таким чином, правий вузол геш-дерева в даному випадку завжди присутній. Викладений вище процес повторюється, поки не буде отримано корінь дерева Меркла.

На рисунку 4.5 показано сформоване повне бінарне геш-дерево з кореневим значенням X_{root} для дев'яти PSM, хеш значення яких відповідають листовим вузлам X_1^4, X_2^4, …, X_9^4, X_{10}^4.

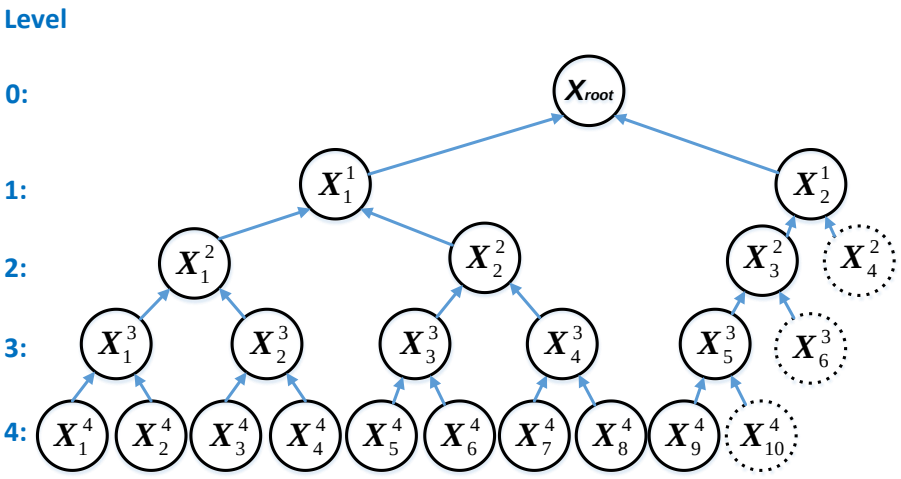


Рисунок 4.5. Повне бінарне геш-дерево

Вузли, позначені пунктиром, це дублікати геш-значень відповідних листового вузла та внутрішніх вузлів геш-дерева, а саме: X_{10}^4 – дублікат геш-значення листового вузла X_9^4; X_6^3 – дублікат геш-значення внутрішнього вузла X_5^3; X_4^2 – дублікат геш-значення внутрішнього вузла X_3^2.

Послідовність операцій процесу обчислення кореня дерева Меркла у цьому варіанті представлена рисунку 4.6.

Merkle Root Calculation (MRC-2)

Input: $SO = \{SO_1, ..., SO_k, ..., SO_{n_{sp}}\}$, $hash$, z, n_{sh}

Output: Mercle root - h_{root}

1: Сортування PSM (SO_k) за алфавітом (в залежності від їх імені та типу)
2: $h_k = X_k^{H-1} = LH(SO_k) = hash(0x00 \| SO_k),\ k = 1,...,n_{so}$ // Розрахунок геш-значень для SO_k
3: $cur\_hashes = A_{hash} = [h_1, \ldots, h_k, \ldots, h_{n_{so}}]$ // копіювання геш-значень в масив
4: $parents\_hashes$ = [] // створення масиву для зберігання батьківських гешів
5: while cur_hashes.size()!=1 // цикл: поки масив не буде складатися з 1 елементу
6: { for (i=1; i<=cur_hashes.size();i=i+2) // усі геші поточного рівня
7: { $left = cur\_hashes[i]$
8: $right = \begin{cases} cur\_hashes[i+1], \text{ if } \exists\, cur\_hashes[i+1]; \\ null. \end{cases}$
9: if (right==null) then
10: $X = IH(left, right) = hash(0x01 \| left \| left)$
11: else
12: $X = IH(left, right) = hash(0x01 \| left \| right)$
13: $parents\_hashes.add(X)$ // додати геш-значення до батьківського масиву
14: }
15: $cur\_hashes = parents\_hashes$ // перевизначення масиву
16: $parents\_hashes$ = [] // очистити масив
17: }
18: $h_{root} = cur\_hashes[1]$
19: return h_{root}

Рисунок 4.6. Схема процесу обчислення кореня дерева Меркла (MRC-2)

У загальному випадку процес обчислення кореня дерева Меркла можна представити у вигляді наступної послідовності дій.

1. Операції над вхідними даними:

– упорядкування збережених у схемі БД (n_{sh}) PSM ($SO_k \in SO;\ k = 1,...,n_{so}$) за абеткою (залежно від імені та типу);

– обчислення відповідно до виразу (4.4) та обраної (з деякого наявного кінцевого набору) криптографічно сильною геш-функцією ($hash$), що застосовується z разів, значень гешей PSM.

2. Вибір стратегії та схеми формування дерева (MRC-1 або MRC-2), а також способу зберігання проміжних значень.

Дерево Меркла формується знизу нагору. Слід зазначити, що саме дерево може зберігатися у пам'яті. Використовуючи рекурсивний алгоритм обходу вузлів дерева, останні створюються та динамічно редукуються у процесі їх обробки. Але для прискорення в подальшому процесу верифікації цілісності об'єктів, що зберігаються в базі даних, є можливість збереження відповідної версії дерева, у тому числі і у вигляді послідовності бітів (у форматі серіалізації). Потрібна

пам'ять для зберігання проміжних значень пропорційна висоті дерева $H = \lceil \log_2 n_{so} \rceil$.

3. Розрахунок кореня геш-дерева.

Корінь дерева Меркла (h_{root}) є криптографічним доказом цілісності блока. Його значення фіксується у заголовку блока. Як вже зазначалося вище, при використанні геш-дерев набагато простіше (потрібно менше обчислювальних витрат) довести належність певного елемента даних (збереженої програми) до їх множини. Клієнти, які зберігають лише заголовки блоків, а не їх вміст, назвемо їх легкими (такі клієнти відомі, наприклад, користувачам біткойн-мережі, як спрощені клієнти (англ. lightweight client), клієнти зі спрощеною перевіркою платежів (англ. simple-payment-verification client), для перевірки інформації про PSM не перераховують всі геші, а запитують у сервера для доказу існування та цілісності цього PSM шлях автентифікації, оскільки такий клієнт може не потребувати інформації про інші PSM бази даних. Клієнт, обчисливши по отриманих гешах шляху автентифікації значення кореня Меркла, і, порівнявши його з наявним, може переконатися в цілісності PSM, що перевіряється. На рисунку 4.7 показаний приклад дерева Меркла, для якого клієнт по отриманому шляху автентифікації довжиною всього в чотири геш-значення (відзначені штрихуванням: H_B, H_{89}, H_{CDEF} и $H_{01234567}$) може перевірити цілісність збереженої програми A шляхом обчислення чотирьох додаткових парних гешей H_{AB}, H_{89AB}, $H_{89ABCDEF}$ та кореня дерева Меркла (обведено пунктиром на рисунку).

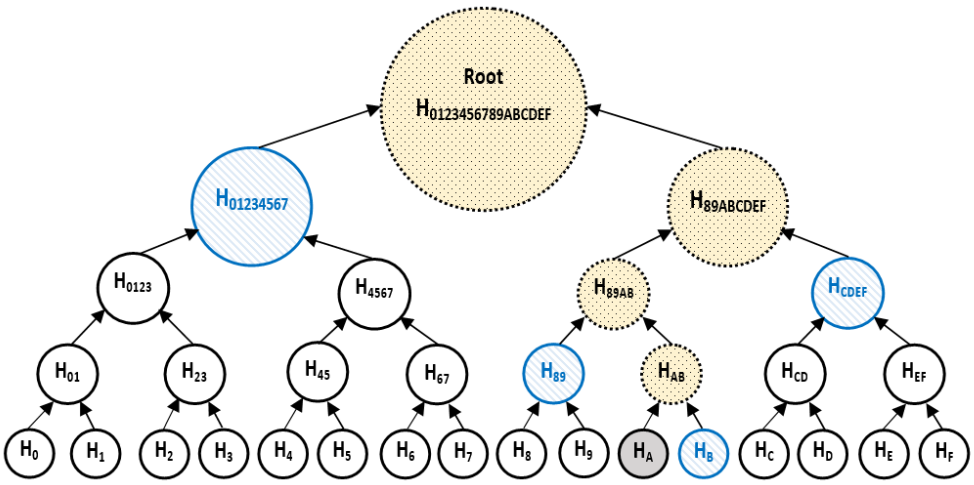


Рисунок 4.7. Приклад дерева Меркла

4.4. Відображення структури блокчейна у відношення реляційної моделі

Структуру блокчейна, зображену рисунку 4.1, можна досить просто представити у вигляді двох відношень, що є її відображенням. А саме, відношення *заголовків блоків блокчейна* та деякої іншої додаткової інформації – R_{bch} (вираз (4.7)) та відношення *збережених програм* – R_{SP} (вираз (4.8)), схеми яких містять фіксовану множину атрибутів, точніше пар <*ім'я-атрибута* : *ім'я-домена*> { $< A_1 : D_1 >, < A_2 : D_2 >, \ldots, < A_l : D_l >$ }.

$$R_{bch}(i_{id}, t, d_{DB}, n_{DB}, n_{sh}, h_{root}, h_{block}, h_{p\_block}, n_{so}, w \mid i_{id} \in \mathbb{N}^* \wedge t \in T \wedge d_{DB} \in Nm^{d_{DB}} \wedge \\ \wedge n_{DB} \in Nm^{DB} \wedge n_{sh} \in Nm^{sh_{DB}} \wedge h_{root} \in H_{Mr} \wedge h_{block} \in H_b \wedge \\ \wedge h_{p\_block} \in (H_b \bigcup \varnothing) \wedge n_{so} \in \mathbb{N}^* \wedge w \in W), \quad (4.7)$$

де i_{id} – номер i-го блока блокчейна; t – мітка часу створення блоку (T – *UTC*); $Nm^{d_{DB}}$ – множина імен доменів БД; d_{DB} – ім'я домену конкретної БД; Nm^{DB} – множина імен БД; n_{DB} – ім'я конкретної БД; $Nm^{sh_{DB}}$ – множина імен схем БД, n_{sh} – ім'я конкретної схеми БД (або '*genesis block*'); H_{Mr} – множина гешей коренів Меркла ($H_{Mr} = \{0,1\}^n$ – множина всіх слів довжини n в алфавіті $\{0,1\}$), h_{root} – геш кореня дерева Меркла i-го блоку ($i = 1..N_{bc}$, де N_{bc} – загальна кількість блоків блокчейна); h_{block} – геш заголовка поточного i-го блока; h_{p_block} – геш заголовка попереднього $(i-1)$-го блока; H_b – множина гешей блоків ($H_b = \{0,1\}^n$); n_{so} – кількість контрольованих PSM; $\mathbb{N}^*$ – множина натуральних чисел без нуля; W – множина цифрових підписів ($w \in W$, $W = \{0,1\}^l$).

Нижче наведено результат відображення відношення R_{bch} в базове відношення (таблицю) БД $\varphi_1 : R_{bch} \to T_{\text{BLOCK\_CH}}$, представлений у вигляді основних рядків коду мови визначення даних стандарту ISO SQL, що використовуються в операторах `CREATE (ALTER) TABLE`:

```
iid      NUMERIC(12) PRIMARY KEY,
t        TIMESTAMP (6) WITH TIME ZONE not null,
ddb      VARCHAR(255) not null,
ndb      VARCHAR(50) not null,
```

```
nsh       VARCHAR(50) not null,
hroot     RAW (128) not null,
hblock    RAW (128) not null,
hp_block RAW (128),
nso       NUMERIC(12) not null,
w         RAW (128) not null,
unique   (ddb, ndb, nsh, hroot, hp_block),
unique   (hblock);.
```

Включення атрибуту hp_block до складу унікального ключа (ddb, ndb, nsh, hroot, hp_block) пояснюється тим, що у разі необхідності повернення до попередньої версії збережених програм можна було б виконати збереження цих даних ланцюжка блокчейна у відповідному рядку таблиці $T_{\text{BLOCK\_CH}}$. Приклад частково заповненої таблиці наведено нижче (Таблиця 4.3).

Таблиця 4.3 – Приклад частково заповненої таблиці бази даних $T_{\text{BLOCK\_CH}}$

iid / i_{id}	**t /** t	**ddb /** d_{DB}	**ndb /** n_{DB}	**nsh /** n_{sh}	**hroot /** h_{root}	**hblock /** h_{block}	**hp_block /.** h_{p_block}	**nso/** n_{so}	**w /** w
29698 7922	21-APR-20 06.00.13.000000 PM +03:00	ua.xxx.com	WORKGR\ DESKTOP-QRRDTTA	genesis block	D420161F35294B0A647DD3E6253C57AE258EC417D1014EFC483A66E7B6A91CE1	D420161F35294B0A647DD3E6253C57AE258EC417D1014EFC483A66E7B6A91CE1		1	…
29698 7923	22-APR-20 02.34.01.575000 PM +03:00	ua.xxx.com	orcl	SYS	4DC69C6660AF511F08D3F89FE899D19396269676F6578832EBC452EA45F4AD56	442F64B40C2CBA0E4786DEC2FB9FA64C310C8555F8E6F1582E1651AEB7501CEB	D420161F35294B0A647DD3E6253C57AE258EC417D1014EFC483A66E7B6A91CE1	9799	…
29698 7924	22-APR-20 02.36.24.606000 PM +03:00	ua.xxx.com	orcl	user_1	3538FDE46591936C2FF53D069093231E9F72C316451629D44FAAE4AB221FE2D1	F5415080C68CE7E671F5262A968CE013B70C6B3BEC200C9E90192D5AA22ED6EC	442F64B40C2CBA0E4786DEC2FB9FA64C310C8555F8E6F1582E1651AEB7501CEB	326	…
29698 7925	22-APR-20 03.08.36.742000 PM +03:00	ua.xxx.com	orcl	user_2	74B846CCB97171B7DE1BB030A816323DEF55FDA790FA8542E7F827554F9832BD	E8F2615EC3C765F98EF717900D6F9A533137C9E577CF9476DB14E4F23F6C95D0	F5415080C68CE7E671F5262A968CE013B70C6B3BEC200C9E90192D5AA22ED6EC	254	…
29698 7926	22-APR-20 03.15.00.461000 PM +03:00	ua.xxx.com	orcl	user_3	7B9B563A38169B016726D3BAD00883224DA30C2D1DF6863CFB6A88D4EF167F0D	D9F3CDD236978C6B3F750B9BCD270C2C97D32AC1DF24BB85515D3A926F01B4B9	E8F2615EC3C765F98EF717900D6F9A533137C9E577CF9476DB14E4F23F6C95D0	5	…
29698 7927	22-APR-20 03.22.00.678000 PM +03:00	ua.xxx.com	orcl	user_1	B404B7901D760F3820D22442E9665485A5CFD9174C4B3F3D5DF6C92E00C0B584	9206A9C3D6D5115F1C880F112F702CA9E49FA84B172A202573685521B5EC2F8E	D9F3CDD236978C6B3F750B9BCD270C2C97D32AC1DF24BB85515D3A926F01B4B9	327	…
…	…	…	…	…	…		…	…	…

У перший рядок таблиці $T_{\text{BLOCK\_CH}}$ заносяться дані для genesis block. При цьому в стовпці h_{root}, h_{block} заноситься значення h_{gen} ($h_{root}^{genesis} = h_{block} = h_{gen}$), що обчислюється відповідно до виразу (4.1), в стовпець n_{so} – значення 1, в стовпець

n_{sh} – значення *genesis block*, у стовпець n_{DB} – ім'я хост-машини, де встановлено СКБД; цифровий підпис w, обчислений відповідно до формули (4.3) і записаний у перший рядок таблиці $T_{\text{BLOCK\_CH}}$, є гарантом того, що цей первинний блок був сформований саме тим учасником інформаційного обміну, кому всі зацікавлені особи довірили це право відповідно до протоколу угоди.

Значення атрибутів інших рядків таблиці $T_{\text{BLOCK\_CH}}$ визначаються відповідно до відповідних виразів та конкретних системних параметрів.

За такої структури $T_{\text{BLOCK\_CH}}$ в кінцевих її рядках будуть розміщуватись дані відповідних схем, актуальні на поточний момент часу. Якщо необхідно визначити інформацію про стан PSM конкретної схеми БД актуальну на довільний момент часу, необхідно звернутися до рядків таблиці $T_{\text{BLOCK\_CH}}$, для яких знайдений найближчий час не перевищує зазначене. З цією метою потенційно можна використати так званий ретроспективний запит. Однак за його використання слід враховувати, що можливості ретроспективи принципово обмежені. Ретроспективна інформація надходить із сегментів відкату, обсяг яких не безмежний. Це означає, що відкат можна виконувати лише до певного моменту часу. Спроба відкату за межі сегментів відкату призведе до генерації помилки з повідомленням про брак інформації в сегментах відкату для отримання відповідної версії даних. Наскільки далеко можна повернутися в часі, залежить від загальної інтенсивності виконання дій у базі даних та розміру буферів відкату. Слід пам'ятати, що буфери відкату фіксують кожну зміну у базі даних, зокрема у внутрішніх таблицях, які СКБД оновлює самостійно. Тому, мабуть, не вдасться повернутись у часі дуже далеко. Але з іншого боку більший інтерес, як правило, викликає інформація про цілісність збережених програм на поточний час, а застаріла інформація про PSM контрольованих схем БД у минулому, відображена в ланцюжку блоків, необхідна для моніторингу процесу зміни версій програм. Хоча ця інформація також може допомогти у виявленні несанкціонованих дій зловмисників або шкідливих програм.

Відношення збережених програм R_{SP} має такий вигляд:

$$R_{SP}(i_{id}, p_k, \alpha_k, h_k \mid i_{id} \in \mathbb{N}^* \wedge p_k \in type_{sp} \wedge \alpha_k \in Nm_{sp} \wedge h_k \in H_{sp}), \quad (4.8)$$

де Nm_{sp} – множина імен PSM, H_{sp} – множина гешей збережених програм ($H_{sp} = \{0,1\}^n$). Нижче наведено результат відображення R_{SP} в базове відношення БД $\varphi_2 : R_{SP} \to T_{\text{STOR\_P}}$, представлений у вигляді основних рядків коду мови визначення даних стандарту ISO SQL, що використовуються в операторах `CREATE (ALTER) TABLE`:

```
iid     NUMERIC(12) not null,
pk      VARCHAR(50)  not null,
alk     VARCHAR(100) not null,
hk      RAW (128) not null,
PRIMARY KEY (iid, pk, alk),
FOREIGN KEY (iid) references BLOCK_CH (iid);.
```

У таблиці 4.4 наведено приклад частково заповненої таблиці $T_{\text{STOR\_P}}$, сформованої на початковому етапі побудови дерева Меркла – етапі обчислення гешей PSM при одноразовому використанні геш-функції SHA-256.

Таблиця 4.4 – Приклад частково заповненої таблиці бази даних $T_{\text{STOR\_P}}$

iid / i_{id}	**pk /** p_k	**alk /** α_k	**hk /** h_k
296987923	FUNCTION	AQ$_GET_SUBSCRIBERS	05A85236D79D0FFB86DEB11B1F5D155C49B831A008C6E96F4A389C3896540107
…	…	…	…
296987923	PACKAGE BODY	DBMS_WARNING	07FBA88D5A80A22E46CFF984BBC68FC916E992AB44D37A5488F4AE75225AC8A5
…	…	…	…
296987924	PACKAGE	DAED_SESS	C9B0A9141391C3F004A5B9D56D7BB41AF36A3CD6730475C74CEC49CF8C837CFA
…	…	…	…
296987924	PROCEDURE	DOCS_P	DF83407E9C339372455C446C9553799ADE00DB636D85415A19ACBE6209047B00
296987924	TRIGGER	EVENTS_IUBR	1BC79E740AF61F9D246A7050EA7DFEE1937A8A274347A6383EB47FE3B4DDF181
…	…	…	…

Важливе рішення, яке потрібно приймати в будь-якій блокчейновій мережі, – визначити, який вузол буде додавати наступний блок до ланцюжка [203]. При цьому необхідно враховувати, що вузол у блокчейновій мережі може виконувати різні функції в залежності від ролі, що приймається. У запропонованому методі у процесі інформаційного обміну та управління даними передбачається участь як вузлів (під якими розуміються пристрої користувача –

«клієнти»), які відповідно до наявних привілеїв можуть визначати новий PSM БД у відповідній схемі (власники схеми), так і вузлів, які можуть тільки запитувати інформацію про існування деякого об'єкта та його цілісності (так звані легковажні вузли – інші легітимні користувачі – тонкі клієнти системи).

З метою підвищення безпеки даних, що зберігаються у вищезгаданих таблицях $T_{\text{BLOCK\_CH}}$ та $T_{\text{STOR\_P}}$, що є фізичним відображенням структури блокчейна, останні (таблиці) пропонується розміщувати у схемі привілейованого користувача. А доступ до них відповідно до певного протоколу угоди, за допомогою якої фіксується загальновизнана істина про стан записів у блокчейні, дозволити лише обмеженому колу користувачів (власникам схем) та лише на читання та запис (привілеї на операції SELECT, INSERT). Крім того, щоб захиститися від незаконних дій зловмисників, які використовуючи різні вразливості СКБД, змогли б отримати певні привілеї, у тому числі зуміли б їх підвищити до рівня привілейованого користувача, який може здійснювати практично будь-які маніпуляції з будь-якими даними та об'єктами схем БД, доцільно підписувати кожним творцем-власником конкретної схеми БД «свої» відповідні дані (результат конкатенації гешованих значень: геша кореня Меркла, timestamp, кількості об'єктів), що зберігаються в таблиці $T_{\text{BLOCK\_CH}}$, одним із сучасних алгоритмів цифрового підпису. При цьому деякі дані таблиць $T_{\text{BLOCK\_CH}}$ (значення атрибутів t, d_{DB}, n_{DB}, n_{sh}, h_{root}), $T_{\text{STOR\_P}}$ (значення атрибутів p_k, α_k) можна перетворити на формат JSON (як більш відповідний формат для серіалізації складних структур, що полегшує обмін такими структурованими даними між усіма мовами програмування), наприклад, у такому уявленні:


```
{"timestamp":"2020-04-22T11:34:01.575000Z",   "ddb":"ua.xxx.com",   "ndb":"orcl",
"nsh":"SYS","hroot":"4DC69C6660AF511F08D3F89FE899D19396269676F6578832EBC452EA45F4
AD56","pk":"FUNCTION","alk":"AQ$_GET_SUBSCRIBERS"}
{"timestamp":"2020-04-22T11:34:01.575000Z",   "ddb":"ua.xxx.com",   "ndb":"orcl",
"nsh":"SYS","hroot":"4DC69C6660AF511F08D3F89FE899D19396269676F6578832EBC452EA45F4
AD56","pk":"FUNCTION","alk":"CHECK_UPGRADE"}
…
{"timestamp":"2020-04-22T11:34:01.575000Z",   "ddb":"ua.xxx.com",   "ndb":"orcl",
"nsh":"SYS","hroot":"4DC69C6660AF511F08D3F89FE899D19396269676F6578832EBC452EA45F4
AD56","pk":"PACKAGE BODY","alk":"DBMS_WARNING"}
…
```

{"timestamp":"2020-04-22T12:08:36.742000Z", "ddb":"ua.xxx.com", "ndb":"orcl", "nsh":"user_2","hroot":"74B846CCB97171B7DE1BB030A816323DEF55FDA790FA8542E7F827554F9832BD","pk":"PACKAGE BODY","alk":"DAED_SESS"}
…
{"timestamp":"2020-04-22T12:22:00.678000Z", "ddb":"ua.xxx.com", "ndb":"orcl", "nsh":"user_1","hroot":"B404B7901D760F3820D22442E9665485A5CFD9174C4B3F3D5DF6C92E00C0B584","pk":"VIEW","alk":"V_TOP_CLASSES"}
…

Після цього сформувавши з цих даних певний файл – реєстр (спеціалізовану базу даних), який розповсюдити всім користувачам, включаючи тих, що не є власниками схем БД. По-перше, для можливості здійснення дублюючого контролю несанкціонованих змін у PSM бази даних, по-друге, для можливості легітимним користувачам легковажних вузлів формулювати коректні запити для отримання інформації про цілісність використовуваних в їх застосунках збережених програм. Використовуючи концепцію геш-дерев, і, маючи певні дані файлу-реєстру (заголовки блоків, імена та типи об'єктів) легітимний користувач зберігає можливість визначити факт наявності і цілісності PSM, що його цікавить за допомогою отримання невеликого за розміром обсягу даних (у вигляді шляху автентифікації в дереві Меркла) від повноцінного вузла (сервера БД) без необхідності зберігання чи передачі величезного обсягу даних блокчейна. Якщо припустити, що зловмисник (шкідливе програмне забезпечення) якимось чином змінив існуючий PSM або розмістив в одній із схем БД код нового нелегітимного PSM, то знаючи алгоритм гешування (геш функцію для знаходження h_k), він повинен внести відповідні зміни до даних таблиці $T_{\text{STOR\_P}}$. І навіть якщо йому вдасться це зробити, все одно подібні несанкціоновані дії будуть виявлені, оскільки дані таблиці $T_{\text{STOR\_P}}$ пов'язані з даними, що зберігаються в атрибутах h_{root}, h_{block} таблиці $T_{\text{BLOCK\_CH}}$, зміна яких вимагає певних додаткових привілеїв, а також знання особистого ключа власника відповідної схеми БД для підпису даних відповідно до виразу (4.4). До того ж змінити дані у всіх файлах формату JSON, що є відображенням даних таблиць $T_{\text{BLOCK\_CH}}$, $T_{\text{STOR\_P}}$ і що знаходяться у віданні безлічі автономних вузлів, практично нездійсненне завдання. Виявлення несанкціонованих змін відбудеться при черговій перевірці, яку може ініціювати будь-який із легітимних користувачів системи, звернувшись на сервер із

відповідним запитом для підтвердження (або спростування) цілісності певного PSM. При цьому обсяг інформації, що обмінюється між клієнтом і сервером СКБД для перевірки цілісності конкретного PSM – мінімальний. Він визначається розміром запису контрольованих на поточний момент часу даних (імені і типу об'єкта, що зберігається) в JSON форматі у вигляді відповідної множини пар ключ-значення. На сервері СКБД контроль цілісності збережених програм може задаватися з певною періодичністю в рамках аудиту із записом відповідної інформації в журнал аудиту з подальшим його аналізом і вживання дієвих заходів.

4.5. Висновки за розділом

1. В результаті проведених досліджень у напрямку вирішення важливого завдання забезпечення безпеки баз даних з точки зору цілісності і справжності програмних модулів, що зберігаються в них, був розроблений новий метод моніторингу, що ґрунтується на можливостях технології блокчейн.

2. Відповідно до запропонованого методу були:

а) визначено: структура та правила формування первинного та наступних блоків у блокчейновому ланцюжку. За основу структури блоків була взята контейнерна структура даних, що об'єднує транзакції для включення їх до загальнодоступного реєстру системи Bitcoin, з урахуванням специфіки розглянутого рішення; два методи обчислення кореня геш-дерева, що формується на основі бінарних дерев різних типів: незбалансованого бінарного дерева та повного бінарного дерева, та спосіб зберігання проміжних результатів. Доцільність використання структури геш-дерева, його кореня, механізму цифрового підпису для контролю цілісності та справжності PSM обумовлена об'єктивною потребою раціонального витрачання ресурсів, що веде до економії обсягу збережених для цього даних та необхідного процесорного часу;

б) запропоновані: підхід до зберігання структури блокчейна в рамках реляційної моделі даних, що полягає у відображенні цієї структури у два відношення: відношення заголовків блоків блокчейна – R_{bch} і відношення збережених програм бази даних – R_{SP}; рекомендації щодо розміщення таблиць

$T_{\text{BLOCK\_CH}}$ та $T_{\text{STOR\_P}}$, виходячи з обмеження можливості несанкціонованої зміни їх даних. Ці таблиці пропонується розміщувати у схемі привілейованого користувача, доступ до якого іншому обмеженому колу користувачів (власникам схем) дозволено лише на читання та запис. Крім того, деякі дані таблиць $T_{\text{BLOCK\_CH}}$, $T_{\text{STOR\_P}}$ доцільно перетворити на формат JSON (як більш відповідний формат для серіалізації складних структур, що полегшує обмін такими структурованими даними між усіма мовами програмування), після чого сформувавши з цих даних деякий файл – реєстр, який поширити всім легітимним користувачам множини автономних вузлів для додаткової можливості моніторингу цілісності та справжності PSM;

в) сформульовано вимогу про необхідність підпису творцем конкретної схеми БД «своїх» відповідних даних, представлених у таблиці $T_{\text{BLOCK\_CH}}$, одним із сучасних алгоритмів цифрового підпису, щоб захиститися від неправомочних дій привілейованого користувача, а також від нелегітимних дій зловмисників, які отримали незаконним способом привілеї власника тієї чи іншої схеми щодо відповідних PSM.

3. Розроблений метод дозволяє забезпечити суворий контроль набору збережених програм БД, їх цілісність, справжність при менших обсягах збережених для цього даних і необхідних часових ресурсів процесора, ніж при використанні відомого методу контрольних сум, що вимагає в даному випадку виконання процедур гешування та цифрового підпису із збереженням відповідних даних для кожного конкретного PSM у конкретній схемі БД.

4. Пропоноване рішення, в якому використовується закритий розподілений реєстр – блокчейн з фіксованим списком учасників та обмеженими правами доступу з управління інформацією, який прагнуть використовувати великі компанії, що підтримують складні бізнес-процеси, може бути використане адміністраторами СКБД для своєчасного виявлення несанкціонованої модифікації PSM, а також як науково-методологічна основа розробки нових інструментальних засобів у складі сучасних СКБД для забезпечення безпеки різних об'єктів, що зберігаються в базах даних.

Результати, представлені у розділі, опубліковані у роботах [123, 213].

5. РЕАЛІЗАЦІЯ МЕТОДІВ І ЗАСОБІВ ЗАХИСТУ В БАЗАХ ДАНИХ, ПОБУДОВАНИХ НА ОСНОВІ СХЕМИ З УНІВЕРСАЛЬНИМ БАЗИСОМ ВІДНОШЕНЬ

5.1. Методи та засоби забезпечення конфіденційності даних

Деякі аспекти забезпечення конфіденційності, пов'язані з запропонованими методами приховування чутливих даних, що становлять інтерес для зловмисника, були розглянуті в розділі 3 (в Додатку Д.1 наведено лістинги деяких збережених функцій спеціального пакета, що дозволяє реалізувати ці методи). Однак це лише частина відповідних методів та засобів, що використовуються для захисту баз даних, у тому числі БД з УББВ. Порушення конфіденційності можуть виникати з різних причин: через недогляд у політиці безпеки або через неправильно налаштований контроль безпеки, в результаті дій кінцевого користувача або системного адміністратора (привілейованого користувача) тощо. Тому численні контрзаходи, зокрема, шифрування, контроль доступу та стеганографію, можуть допомогти забезпечити конфіденційність від можливих загроз.

Для захисту важливої інформації, що зберігається в БД з УББВ, доступ до неї з одного боку слід обмежити, а з іншого – доцільно її зашифровувати. Шифрування даних є ключовим компонентом під час реалізації принципу багаторівневого захисту. Прагнення зменшити ризик втрати конфіденційності даних, у тому числі через інсайдерські загрози привілейованих користувачів, стало мотивованим початком для розробки механізмів, що забезпечують можливість ефективного використання для захисту криптографічних примітивів, вбудованих у СКБД, розробки власних засобів криптографічного захисту та їх комплексного застосування. У зв'язку з цим на прикладі СКБД Oracle був розроблений пакет відповідних підпрограм та методика його застосування, в якій передбачається комплексне використання як вбудованих у СКБД криптографічних примітивів (Таблиця 5.1), так і національних алгоритмів

шифрування (наприклад, «Калина» з національного стандарту України ДСТУ 7624:2014).

Таблиця 5.1 – Характеристика стандартного пакета програм dbms_crypto

Характеристика пакета	Стандартний пакет *dbms_crypto* (для версій старше Oracle 12*c*)
Алгоритми шифрування	DES, 3DES, AES, RC4, 3DES_2KEY
Типи доповнення	PKCS5, нулі
Режими шифрування	CBC, CFB, ECB, OFB
Криптографічні алгоритми гешування	MD5, SHA-1, SHA-2 (SHA-256, SHA-384, SHA-512), MD4
Алгоритми гешування за ключем (MAC)	HMAC_MD5, HMAC_SH1, HMAC_SH256, HMAC_SH384, HMAC_SH512
Криптографічний псевдовипадковий генератор чисел (тип)	RAW, NUMBER, BINARY_INTEGER
Типи зашифрованих даних	RAW, CLOB, BLOB

Лістинги деяких спеціальних функцій-оболонок, інкапсульованих у розроблений пакет і що дозволяють розширити можливості стандартних підпрограм вбудованих у СКБД пакетів та зробити процес шифрування більш простим та гнучким, а також приклади їх використання наведені у Додатку Д.2. Пропонований пакет підпрограм може бути ефективно використаний розробниками прикладного програмного забезпечення при створенні ними відповідних застосунків, що працюють з конфіденційними даними БД, забезпечуючи їхню безпеку на рівні сервера БД. Даний пакет, у тому числі, був використаний при захисті даних таблиці ключів R^{secret} (розділ 3) від неправомочних користувачів, включаючи привілейованих (шляхом застосування криптостійких алгоритмів), при формуванні початкових значень $X^0_{R_{ij}}$ для ГПВЧ, що значно відрізняються один від одного (за рахунок можливості застосування однієї з криптографічних геш-функцій: MD4, MD5, SHA-1, SHA-256, SHA-384, SHA-512). При цьому ключ розшифрування даних таблиці R^{secret}, використовуючи метод стеганографії вбудовування повідомлення, поміщається за допомогою спеціального програмного забезпечення в певний файл-контейнер (стегоконтейнер). Виймається значення ключа також спеціальним ПЗ сервера СКБД зі стегоконтейнера, який пред'явить під час відкриття сеансу автентифікований користувач з відповідними привілеями.

Крім того, для захисту вразливих даних БД, що знаходяться у відповідних файлах, що зберігаються на жорсткому диску та на резервних носіях, від можливого викрадення та розкриття, який необхідний відповідно до багатьох національних та / або міжнародних нормативних документів та правил (таких, наприклад, як GDPR, HIPAA, PCI DSS тощо) пропонується використовувати так званий метод «прозорого шифрування даних» (Transparent Data Encryption – TDE), у якому інформація автоматично розшифровується під час зчитування з носія (якщо було введено правильний ключ спеціального «гаманця») і автоматично зашифровується під час запису. Техніка TDE (захисту файлів БД від можливого викрадення за допомогою шифрування конфіденційної інформації з мінімальними зусиллями) притаманна різним СКБД (патент IBM 7426745 [214]). У розглянутій реалізації БД з УБВ було виконано відповідні умови можливості застосування цієї техніки: визначено місцезнаходження, пароль «гаманця», виконано його активування/відкриття.

В даний час багато користувачів вважають за краще передавати дані на сторонні хмарні сервери, щоб знизити навантаження на локальне сховище. Однак збереження конфіденційних даних на віддалених серверах створює проблеми з безпекою та є джерелом занепокоєння для власників даних. У зв'язку з проблемами безпеки і приватності, що постійно зростають, стає все більш актуальним шифрування даних, що зберігаються віддалено. Однак використання традиційного шифрування перешкоджає виконанню операції пошуку зашифрованих даних. Одним із підходів до вирішення цієї проблеми є шифрування з можливістю пошуку. Рішення для пошуку в захищених базах даних охоплюють широкий спектр криптографічних методів, хоча домінуючого рішення досі не існує [215, 216]. Цей напрямок залишається плідною областю досліджень [2]. У зв'язку з чим було запропоновано рішення, що дозволяє користувачеві та серверу, використовувати стандартну мову SQL для безпечного пошуку даних у віддаленій базі даних (у тому числі побудованої на основі схеми з універсальним базисом відношень), зашифрованою за допомогою надійної та ефективної криптосистеми, а також здійснювати безпечну вставку, модифікацію та видалення

необхідних конфіденційних даних за розумних накладних витрат [217]. Це досягається за рахунок організації оперативного автоматичного розшифрування спеціально розробленим захищеним програмним забезпеченням відповідних даних, необхідних для пошуку без можливості перегляду цих відкритих даних. Останнє стосується у тому числі й привілейованих користувачів (інсайдерів), що насамперед використовують засоби документування сервера СКБД. Пропонований підхід передбачає роздільну обробку SQL-запиту на стороні сервера СКБД і сервера застосунків (проксі бази даних). На сервері застосунків початковий запит попередньо перетворюється на необхідний вид, а потім передається на сервер БД для виконання. Після виконаного на сервері СКБД запиту запитані зашифровані дані передаються проксі-серверу бази даних для їхнього розшифрування і передачі користувачеві (клієнту). Сервер СКБД не може розшифрувати зашифровані криптостійким алгоритмом збережені дані, не запитані застосунком [217].

Як відомо [60, 61], конфіденційність і цілісність як основні концепції тріади CIA залежать один від одного. Без цілісності об'єкта (неможливість зміни об'єкта без дозволу) конфіденційність не може бути підтримана. До того ж, забезпечення інформаційної безпеки баз даних неможливе без розгляду аспектів забезпечення цілісності даних. Тому далі розглядаються питання забезпечення цілісності даних у БД із універсальним базисом відношень.

5.2. Забезпечення цілісності баз даних з універсальним базисом відношень відповідно до рекомендацій моделі Кларка-Вілсона

Багато, особливо комерційних організацій більше стурбовані цілісністю своїх даних, ніж їх конфіденційністю [60, 61]. Цілісність для них є більш важливою. Якщо ви публікуєте інформацію в Інтернеті на Web-сервері і вашою метою є зробити її доступною для найширшого кола людей, то конфіденційність у цьому випадку не потрібна. Зате істотно підвищується відповідальність за надання неспотвореної інформації, що отримується з БД, наприклад, про

відомості, що зберігаються в ній, з офіційних правових, нормативних, фінансових, медичних та інших документів організації, у тому числі і самих цих документів. Інформація має бути справжньою чи непідробною. Дані повинні бути правильними, правдивими, бути справжнім відображенням реальності. У цілому ж, і в комерційному, і у військовому середовищі важко уявити систему, для якої були б не важливі властивості цілісності [62]. Але при цьому слід пам'ятати і враховувати, що контроль цілісності вимагає додаткових ресурсів: часу та пам'яті. Наприклад, основною проблемою реалізації механізмів контролю цілісності файлових об'єктів є їхній досить сильний вплив на завантаження обчислювального ресурсу системи, що обумовлюється наступними причинами [119]: по-перше, може знадобитися контроль великих обсягів інформації, що пов'язано зі значною тривалістю виконання процедури контролю; по-друге, може знадобитися безперервне підтримання об'єкта в еталонному стані. У зв'язку з чим виникає закономірне питання: з якою періодичністю здійснювати контроль? Адже моніторинг цілісності файлів – це ефективний підхід до виявлення агресивної поведінки шляхом виявлення дій щодо модифікації відповідних критичних файлів [218]. Якщо виконувати його часто, це призведе до істотного зниження продуктивності системи, якщо рідко, ефективність такого контролю може бути низькою. Тому одним із основних завдань при реалізації механізмів контролю цілісності файлових об'єктів є вибір принципів та механізмів запуску процедури перевірки цілісності.

Для того, щоб більш суворо та науково обґрунтовано викласти результати прикладних досліджень, пов'язаних із забезпеченням цілісності у базах даних, побудованих на основі схеми з універсальним базисом відношень, доцільно скористатися деякою моделлю безпеки. Використання формальних моделей безпеки дозволяє формулювати вимоги до створюваних захищених систем (в даному випадку до БД) у чітко визначеній формі, що відповідає безпеці, прийнятій в організації. У загальному випадку модель безпеки може бути отримана з нуля, використовуючи математичну модель, або шляхом розширення існуючої. Хоча обидва ці підходи непрості, оскільки вимагають необхідної формалізації та

повторного доказу [219]. Тому, проаналізувавши з урахуванням особливостей аспектів, що розглядаються, широко відомі моделі цілісності: Біба [220], Кларка-Вілсона [116] та їх застосування [60-62, 117, 118, 127, 219, 221-223], а також менш відомі: модель Гогена-Мезегера (англ. Goguen-Meseguer) [224], Сазерлендську (англ. Sutherland) модель захисту [225], за основу була взята модель Кларка-Вілсона, що розглядається в п. 1.3.2.1, як найбільш підходяща, що використовує багатогранний підхід до забезпечення цілісності. Дана модель не вимагає використання решітчастої структури, і замість визначення формального кінцевого автомата визначає кожен елемент даних і допускає модифікації лише за допомогою невеликого набору програм [60, 61]. Цілісність даних, відповідно до моделі Кларка-Вілсона, досягається за рахунок [226]: автентифікації, аудиту, правильно сформованих транзакцій, поділу обов'язків.

Відомо, що отримати доступ до даних будь-якої сучасної БД можна лише через СКБД. Традиційна СКБД забезпечує автентифікацію, авторизацію, управління транзакціями та даними, ведення журналів тощо. Так для перевірки наявності у суб'єкта (користувача, процесу) необхідних повноважень для виконання необхідної операції в традиційних СКБД, так званому диспетчері БД [7], є спеціальний модуль контролю прав доступу. Тому, реалізація БД з УБВ на платформі деякої обраної реляційної СКБД, автоматично призводить до виконання вимог правила Е3 моделі Кларка-Вілсона, яке наказує системі автентифікувати всіх користувачів (кожен суб'єкт), які намагаються виконати будь-яку процедуру TP. Відповідно до правила Е4, права доступу суб'єктів (враховуючи їх функціональні обов'язки) до об'єктів БД з УБВ (оброблюваним елементам CDI) можуть бути призначені та змінені лише спеціально уповноваженими суб'єктами (адміністратором безпеки, адміністратором БД, власником схеми БД). Для цієї мети використовуються команди (оператори) GRANT / REVOKE стандарту SQL. Крім того, враховуючи особливості схеми та можливості застосування БД з УБВ [14, 15], був розроблений додатковий механізм надання привілеїв, що реалізується в рамках технології RLS, що зажадав введення деяких додаткових до наявної основної схеми БД з УБВ відношень:

– *відношення користувачів* U:

$$U = \{(u_1, u_2, u_3) \mid u_1 \in U_1 \wedge u_2 \in U_2 \wedge u_3 \in U_3 \wedge$$
$$(\ (\forall u_1 \forall u_2 \forall u_3 (\forall u_2' \in U_2)\ \ (U_{pr}(u_1, u_2, u_3) \wedge U_{pr}(u_1, u_2', u_3) \to u_2 = u_2')\ ) \wedge ; \quad (5.1)$$
$$(\forall u_1 \forall u_2 \forall u_3 (\forall u_1' \in U_1)\ \ (U_{pr}(u_1, u_2, u_3) \wedge U_{pr}(u_1', u_2, u_3) \to u_1 = u_1')\ )\ )\},$$

де U_1 – множина ідентифікаторів користувачів (суб'єктів), U_2 – множина імен користувачів; $U_{pr}(...)$ – предикатні символи, що співвідносяться відношенню U; U_3 – множина привілеїв, що надаються користувачам на виконання таких операцій, як видалення, вставка, зміна, зчитування, а також їх комбінації;

– *відношення розподілу прав доступу до даних інших користувачів* G:

$$G = \{(g_1, g_2, g_3) \mid g_1 \in U_1 \wedge g_2 \in U_1 \wedge g_3 \in U_3\}. \quad (5.2)$$

Екстенсіонал відношення G – це набір кортежів, кожен з яких співвідноситься з конкретним користувачем-власником даних (g_1), який передає права доступу (g_3), що належать йому, іншому авторизованому користувачеві (g_2).

Як правило, сьогодні, в реляційних СКБД, окремі записи (поля) спеціально не захищаються, хоча є приклади відомі з практики, коли це потрібно [227, 228]. Тому, з метою забезпечення такої функціональності, враховуючи інваріантність структури відношень R^{sh}, та виходячи з можливостей технології RLS, у рамках схеми БД з УБВ було також визначено спеціальне відношення – *відношення обмежень прав доступу до конкретного елемента даних* ПрО, що моделюється:

$$A = \{(a_1, a_2, a_3, a_4) \mid a_1 \in U_1 \wedge a_2 \in U_2 \wedge a_3 \in R^{sh}_{name} \wedge a_4 \in R^{sh}_{ID}\}, \quad (5.3)$$

де R^{sh}_{name} – множина імен відношень схеми R^{sh} (рисунок 5.1); $R^{sh}_{ID} = \bigcup\limits_i R^{sh}_i[K_{PK_i}]$– множина ідентифікаторів, які є первинними ключами (K_{PK_i}) у відповідних відношеннях R^{sh}, доступ до яких обмежується для користувача $a_1 \in U_1$ з ім'ям $a_2 \in U_2$.

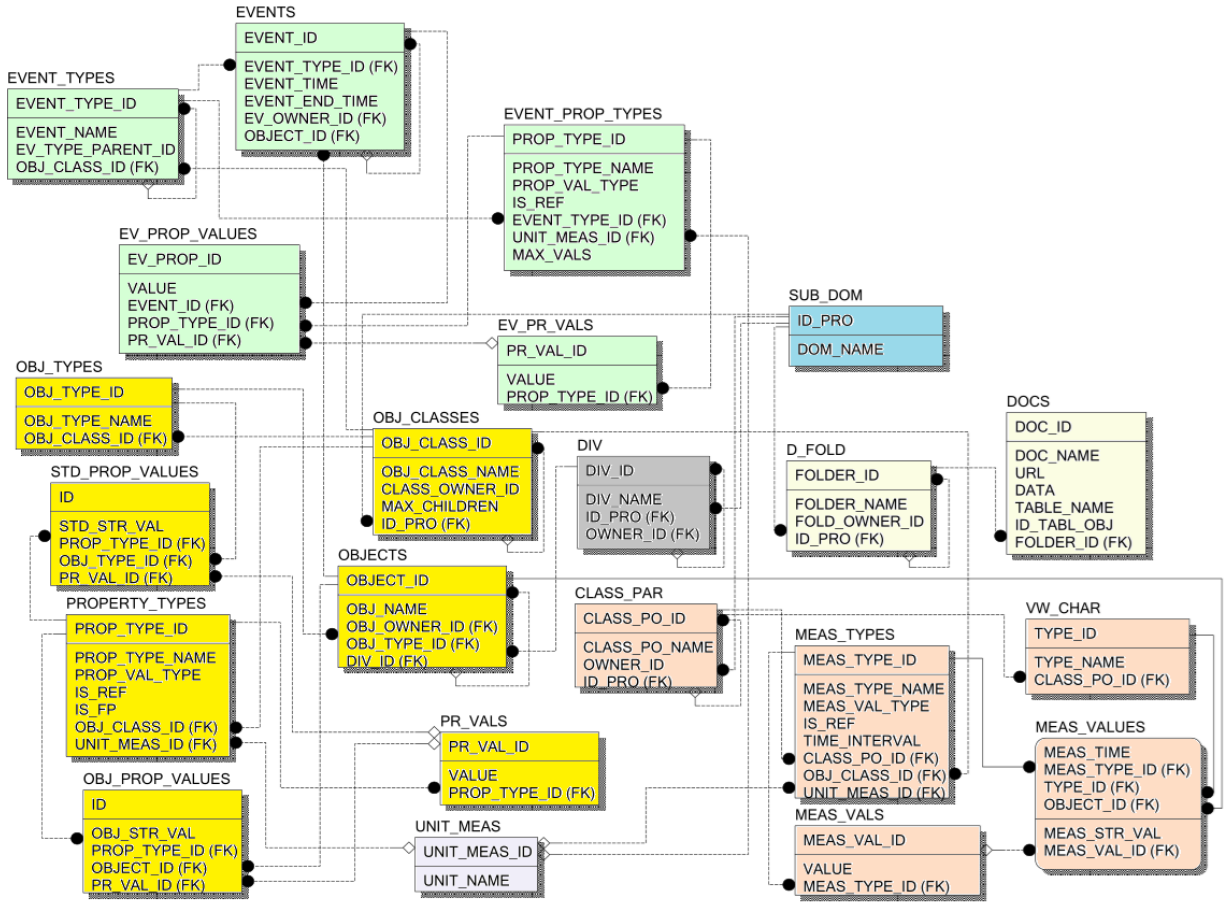


Рисунок 5.1. Діаграма основних базових відношень R^{sh} схеми БД із УБВ

Відповідно до технології RLS було визначено: набір декларативних команд (політик RLS), які визначають, як і коли слід застосовувати обмеження доступу користувачів (відповідно до їхніх функціональних обов'язків, згідно з правилом С3) до кортежів основних відношень схеми БД з УБВ R^{sh}; множина збережених функцій Ψ, які викликаються у випадку, коли виконуються умови, задані в політиці безпеки (політиці RLS); предикати, що формуються функціями Ψ, які СКБД автоматично приєднує в кінець конструкції WHERE, що виконуються користувачем операторів SQL. У сукупності це все можна уявити, як реалізацію правил, що регламентують контроль доступу до даних відношень R^{sh} схеми БД з УБВ:

$$Sr = \{R_i^{sh}, oper_i^{\,j}, policy_i^{k}, \Psi_i^{l}, attr_i^{\mu kl}, pat_{contr}^{R_i^{sh}}\}, \qquad (5.4)$$

де $oper_i^{\,j}$ – j-я комбінація (із значень: select, update, delete, insert) дозволених операцій доступу (процедур перетворення TP) до відношення $R_i^{sh} \in R^{sh}$ (як

одному з елементів CDI); $policy_i^k$ – ім'я (назва) k-ї політики RLS, яка застосовується для базового відношення R_i^{sh}; $\Psi_i^l \in \Psi$ – ім'я l-й функції, що генерує предикат для базового відношення R_i^{sh}; $attr_i^{\mu kl}$ – значення μ-ї ознаки (параметра) для k-ї політики RLS і l-ї функції; $pat_{contr}^{R_i^{sh}}$ – шаблон команд управління доступом до R_i^{sh} (приклад, одного з таких шаблонів наведено в Додатку Д.3 у вигляді елементів програмного коду).

Всі перераховані вище дії робилися для того, щоб СКБД могла контролювати допустимість застосування TP до елементів CDI та забезпечити підтримку списку необхідних конкретним користувачам процедур перетворення TP із зазначенням допустимого для кожної $TP_i \in TP$ і даного суб'єкта ($s_j \in S$) набору оброблюваних елементів CDI відповідно до вимог правил E1 та E2 моделі Кларка-Вілсона.

Для баз даних, що підтримують реляційну модель даних, обмеження цілісності забезпечуються способами декларативної та процедурної підтримки, кожен із яких, по суті, призводить до створення та / або використання деякого програмного коду, що реалізує обмеження. Різниця полягає лише в тому, як такий код створюється та де він зберігається. Причому обмеження цілісності даних мають бути попередньо формально визначені (оголошені), перш ніж СКБД зможе забезпечити їх виконання. У разі операцій, що змінюють вміст бази даних, у традиційних СКБД (у диспетчері БД), як правило, є спеціальний модуль контролю цілісності даних [7], який виконує перевірку того, чи задовольняє запитана операція всім встановленим обмеженням підтримки цілісності даних. У тому числі, даний модуль, приймаючи UDI як вхідні дані, активує TP, яка або конвертує їх в CDI, або відхиляє (згідно з правилом C5). Модуль контролю цілісності даних СКБД, контролюючи допустимість застосування процедур перетворення TP по відношенню до списку елементів CDI відповідно до правила E1, слідкує за коректністю реалізації всіх процедур перетворення TP (відповідно до правила C2), у сенсі того, що останні не повинні порушувати цілісність даних.

І все це з урахуванням того, що в системі повинні бути процедури IVP, здатні підтвердити цілісність будь-якого CDI (правило С1).

При розробці основних об'єктів схеми БД з УБВ, з метою захисту бази від порушення узгодженості даних, що зберігаються в ній, були використані можливості обох способів. Зокрема, у створюваній схемі з допомогою засобів підтримки цілісності, передбачених стандартом мови SQL, було визначено реалізації обмежень цілісності $\Pr^{sh}$, отримані у результаті відображення: $\gamma : \Pr \to \Pr^{sh}$ (де $\Pr$ – множина обмежень цілісності, які специфікуються у моделі даних із УБВ (M_{ubr}) [14]). Суть декларативної підтримки обмежень цілісності – визначення обмежень засобами DDL (*data definition language*) мови SQL. Засоби декларативної підтримки цілісності використовувалися при створенні базових відношень схеми БД з УБВ для визначення таких типів обмежень, як цілісність сутностей, посилальна цілісність, обов'язкові (*not null*) дані, обмеження доменів. А саме, як відомо [7, 27], цілісність сутності, пов'язується насамперед із унікальністю та нескоротністю первинного ключа. Ці вимоги цілісності були визначені для всіх базових відношень схеми – як результат відображення (застосування конструкцій *primary key*, *unique* відповідних операторів SQL): $\gamma_{PK} : \Pr_{PK} \to \Pr^{sh}_{constr_{\text{primary\_key}}}$; $\gamma_{UK} : \Pr_{UK} \to \Pr^{sh}_{constr_{\text{unique}}}$. Нижче наведено приклад такого відображення у вигляді основних рядків коду мови DDL:

```
alter table MEAS_VALUES
  add primary key (MEAS_TIME, MEAS_TYPE_ID, TYPE_ID, OBJECT_ID);).
```

В результаті відображення: (застосування конструкції *foreign key* операторів *create / alter table*), для забезпечення цілісності посилання були визначені зовнішні ключі відношень схеми та стратегії дій при видаленні даних. В результаті відображення: $\gamma_{not\_null} : \Pr_{not\_null} \to \Pr^{sh}_{constr_{\text{not\_null}}}$ (застосування специфікатора *not null* в операторах *create / alter table*) було встановлено обмеження, що забороняють присвоєння відповідним значенням атрибутів невизначеного значення *null*.

Здійснюючи відображення множини обмежень цілісності моделі даних з універсальним базисом відношень M_{ubr}, в інваріантній до предметних областей

схемі БД були визначені обмеження для доменів атрибутів ознак, типів даних характеристик об'єктів, подій, параметрів об'єктів та деякі інші (як результат відображення $\gamma_{dom} : \mathrm{Pr}_{dom} \to \mathrm{Pr}^{sh}_{constr_{\mathrm{check}}}$ – застосування конструкції *check* оператора *create / alter table*). Приклад результату такого відображення:

```
alter table EVENTS add check ((event_end_time is null) or ((event_end_time
is not null) and (event_end_time >= event_time)));.
```

Однак не всі обмеження цілісності можна було реалізувати (сприяючи тим самим забезпеченню виконання вимог правил С1, С2) за допомогою декларативної підтримки. Тому поряд із засобами цього способу реалізації обмежень цілісності широке застосування знайшли засоби процедурної підтримки, такі як: тригери, збережені процедури, функції (для простоти іноді об'єднувані загальною назвою SQL процедурами [157]), механізми яких протягом останніх років у багатьох комерційних СКБД були значно розширені [32, 157]. За допомогою засобів процедурної підтримки у схемі БД з УБВ були реалізовані такі обмеження цілісності ($\mathrm{Pr}^{sh}_{constr_{\mathrm{proc}}}$): обмеження на можливість: зміни занесених у відповідні відношення схеми метаданих ПрО (наприклад, максимальних значень $\mathrm{max\_vals} \in at(R^{sh}_{\mathrm{event\_prop\_types}})$), видалення спискових значень відповідних характеристик із відношень $R^{sh}_{pr\_vals}$, $R^{sh}_{ev\_pr\_vals}$, $R^{sh}_{meas\_vals}$, якщо вони присутні у відношеннях R^{sh}, що асоціюються з даними ПрО, що моделюється; обмеження на можливість введення нових даних, що суперечать введеним метаданим ПрО (для відношень R^{sh}, що асоціюються з даними ПрО); реалізація посилальної цілісності для відношень схеми R^{sh}, пов'язаних з відношенням R^{sh}_{docs} (конкретний документ із відношення R^{sh}_{docs} зв'язується з конкретним екземпляром відповідного відношення R^{sh} (рисунок 5.1)); обмеження максимальної кількості екземплярів об'єктів ($R^{sh}_{\mathrm{objects}}$) для певного класу об'єктів ($R^{sh}_{\mathrm{obj\_classes}}$); обмеження максимальної кількості значень ($R^{sh}_{\mathrm{ev\_prop\_values}}$), які можуть бути присвоєні певної характеристиці події ($R^{sh}_{\mathrm{event\_prop\_types}}$) для екземпляра події ($R^{sh}_{\mathrm{events}}$) заданого класу; обмеження на кількість подій (R^{sh}_{events}), що відбуваються з одним екземпляром об'єкта ($R^{sh}_{\mathrm{objects}}$): *а*) в

один і той же момент часу з одним екземпляром об'єкта не може відбуватися більше однієї події одного класу; *б*) в однієї події, що відбувається з одним екземпляром об'єкта, може бути кілька підпорядкованих подій з різними екземплярами об'єктів, що відбуваються в один момент часу, але у конкретного екземпляра події, що відбувається з екземпляром об'єкта певного класу, подія-власник може бути тільки одна; генерація унікальних значень первинних ключів для відношень схеми R^{sh} та деякі інші.

На рисунку 5.2 представлений підхід до забезпечення цілісності даних у БД з УБВ, що передбачає використання запропонованих способів декларативної та процедурної підтримки обмежень цілісності при розробці об'єктів інваріантної до предметних областей схеми бази даних.

Рисунок 5.2. Забезпечення цілісності даних у БД з УБВ

Крім того, беручи до уваги двоїсту природу систем баз даних, як інформаційного продукту з двома компонентами (активами): власне збереженими

в БД даними, доступними для використання, та програмними засобами СКБД, а також можливості шкідливого впливу на ці активи, доцільним є забезпечення безпеки їх обох. Тому нижче розглядаються деякі аспекти забезпечення цілісності таких важливих об'єктів бази даних, що здійснюють управління даними, як модулі, що постійно зберігаються. До них відносяться згадані вище збережені процедури, функції, які можуть об'єднуватися в пакети, тригери, як особливий вид процедурного коду, і деякі інші. Постійний моніторинг цих об'єктів бази даних (як елементів *CDI*) дуже важливий, оскільки деякі з атак на БД і не тільки на неї (наприклад, можна атакувати операційну систему через вразливість сервера БД) можуть бути виявлені саме на основі аналізу зміни (навмисного чи випадкового) цих об'єктів (порушення їх цілісності), чи його набору (збільшення чи зменшення їх числа) на сервері БД. Тому для забезпечення можливості проведення контролю цілісності таких збережених модулів, у тому числі, що відносяться до схеми БД з УБВ, використовуючи потенціал сучасної блокчейнової моделі, як це показано в розділі 4, були розроблені: структура; методи формування первинного (genesis) та наступних блоків; методи верифікації (у термінології моделі Кларка-Вілсона це *IVP*) цілісності PSM, а також розміщені в одній із схем БД привілейованого користувача два відношення, що є відображенням структури блоків у блокчейновому ланцюжку.

Як відомо, основний недолік, який зазвичай згадується для моделі Кларка-Вілсона, полягає в тому, що *IVP* і пов'язані з ними методи непросто реалізувати в реальних комп'ютерних системах, зокрема через те, що може знадобитися контроль великих обсягів інформації, що пов'язано зі значною тривалістю виконання процедури *IVP* [118]. Так, наприклад, для того щоб здійснити контроль цілісності конкретного модуля (як одного з елементів *CDI*) в конкретній схемі БД звичайним способом потрібно було б виконати процедури гешування і цифрового підпису зі збереженням відповідних даних для кожного з них. Використання ж структури геш дерева дозволяє забезпечити контроль цілісності не тільки конкретного PSM, що перевіряється, але і всіх інших збережених програм обраної схеми БД, у тому числі і процедури, яка забезпечує правильність формування

значень Таблиць 4.3, 4.4. Так як цей один фрагмент даних входить у загальну структуру і зміна хоча б одного біта в ньому спричинить повну зміну значення кореня дерева Меркла. Тому дерева Меркла знайшли широке застосування для безпечної та ефективної валідації (перевірки цілісності) великих структур даних [203, 205, 210, 229, 230]. На сервері СКБД контроль цілісності модулів, що постійно зберігаються, розглянутим вище способом, може задаватися з певною періодичністю в рамках проведеного аудиту із записом відповідної інформації в журнал аудиту з подальшим його аналізом і вживання дієвих заходів. При цьому перевірку цілісності певного PSM може ініціювати будь-який із легітимних користувачів системи, який звернеться на сервер із відповідним запитом (див. 4.4). Використовуючи концепцію геш-дерев, і, маючи певні дані файлу-реєстру легітимний користувач зберігає можливість визначити факт наявності об'єкта, що його цікавить, а також його цілісність за допомогою отримання невеликого за розміром обсягу даних (у вигляді шляху автентифікації в дереві Меркла) від сервера БД без необхідності зберігання чи передачі великого обсягу даних блокчейна.

Підхід використання потенціалу сучасної блокчейнової моделі можна застосовувати і для контролю цілісності даних відношення R_{docs}^{sh}, в якому можуть зберігатися різні документи предметної області, що моделюється, а також таблиці збережених програм $T_{\text{STOR\_P}}$ (цілісність таблиці $T_{\text{BLOCK\_CH}}$ забезпечується цифровим підписом кожним творцем-власником конкретної схеми БД «своїх» відповідних даних).

Не є таємницею, що для створення повноцінної системи безпеки бази даних не менш важливе значення має процедура аудиту. Відповідно до правила С4 моделі Кларка-Вілсона кожне застосування TP має реєструватися в спеціальному елементі CDI – журналі реєстрації, що містить інформацію достатню для відновлення повної картини кожного застосування цієї процедури перетворення, і доступному тільки для додавання до нього інформації. Тому для моніторингу стану, змін, що вносяться до БД, дій користувачів, крім залучення стандартних засобів аудиту СКБД, на платформі якої реалізується схема БД з УББВ,

використовуються розроблені спеціальні функції діагностування, реалізовані в інтерпретаторі мови моделі даних (ММД) [231, 232], здатні виявляти внесення некоректних даних. З цією метою використовуються також тригери, що підтримують протоколювання операцій, що виконуються у БД. Крім того, для контролю дій користувачів можуть використовуватися дані з таблиці-журналу змінених даних [233]. Завдяки інформації, що зберігається в таблиці-журналі, що формується автоматично при вказанні відповідного параметра збереженої процедури інтерпретатора ММД, спрощується процес відновлення неправильно змінених або втрачених даних, та полегшується процедура визначення користувачів, часів та характеру проведених ними змін. Таким чином, аналізуючи з позиції положень моделі Кларка-Вілсона можливості представлених вище розроблених та реалізованих, у тому числі в рамках схеми БД з УББ, методів та засобів, що забезпечують цілісність відповідних елементів *CDI* бази даних, можна зробити висновок, що вони повною мірою відповідають основній ідеї моделі. Основні теоретичні принципи політики контролю цілісності визначають, що потрібно зробити, а реалізовані механізми визначають, як ці принципи досягаються. Отже, бази даних, реалізовані з урахуванням схеми з УББ, вважатимуться відповідними потребам захищених з погляду цілісності БД.

5.3. Систематизація реалізованих заходів захисту

Вирішуючи проблему захисту корпоративних баз даних, побудованих на основі схеми БД з універсальним базисом відношень, від можливих загроз у процесі створення цієї інваріантної до предметних областей схеми БД були розроблені різні заходи забезпечення безпеки. Дані заходи, розглянуті у цьому та попередніх розділах, ґрунтуються як на загальних формальних моделях управління доступом, забезпечення цілісності даних, методах, засобах, механізмах, що підтримуються СКБД, на платформі якої запропонована схема реалізується, так і на власних, розроблених у рамках створення даної схеми. З метою уточнення слід зазначити, що на додаток до розглянутих у роботі заходів, наприклад, методів маскування, пропонується для деяких програм, що

зберігаються, залишити можливість використання вбудованих засобів приховування, а для приховування складу і структури базових і віртуальних відношень (таблиць і уявлень) схеми БД з УБО використовувати розроблені спеціальні табличні функції (pipelined-функції). Ці функції з параметром у вигляді рядка метаопису ММД можуть використовуватися в операторах мови SQL (конструкції FROM) при зверненні до даних БД замість базових та віртуальних відношень схеми БД з УБВ, наприклад,

```
SELECT '<КлассО>='||COLUMN_VALUE as name
FROM TABLE(get_spis_metadata('{<КлассО>=*.*;}'))
ORDER BY name;
```

де *get_spis_metadata* – pipelined-функція з параметром у вигляді рядка метаопису ММД.

Також доцільно звернути увагу на деякі не зазначені раніше особливості запропонованих рішень, які відіграють важливу роль у забезпеченні безпеки БД. Так, наприклад, використання можливостей захисту на рівні рядків (технологія RLS) дозволяє забезпечити доступ до ключів таблиці R^{secret} після її розшифрування лише певним користувачам. Використання можливостей динамічного SQL в пакеті, що реалізує запропонований метод маскування, дозволяє певною мірою не асоціювати даний пакет з його застосуванням для управління ключами. Зловмисник, звернувшись, наприклад, до системного представлення ALL_DEPENDENCIES СКБД Oracle, на платформі якої може реалізовуватись схема БД з УБВ, не зможе з'ясувати, на які таблиці статично посилається пакет. Крім того, у запропонованій реалізації рішення не створюються тимчасові таблиці, які часто становлять під загрозу безпеку в існуючих підходах до маскування.

У систематизованому вигляді реалізовані у схемі БД із універсальним базисом відношень заходи, що забезпечують безпеку баз даних, наведено на рисунку 5.3.

Рисунок 5.3. Заходи безпеки, що реалізовані в схемі БД з універсальним базисом відношень

5.4. Висновки за розділом

1. Для забезпечення безпеки корпоративних баз даних, побудованих на основі схеми БД з універсальним базисом відношень пропонується комплексне використання загальних формальних моделей управління доступом та забезпечення цілісності даних, методів, засобів, механізмів, що підтримуються СКБД, на платформі яких ця схема реалізується, та власних заходів забезпечення безпеки, розроблених у рамках створення інваріантної до ПрО схеми БД.

2. Вирішуючи проблему захисту баз даних, як найважливішого корпоративного ресурсу, у процесі створення схеми БД з УБВ було розроблено спеціальні заходи, у вигляді відповідних методів, реалізованих об'єктів схеми (тригерів, процедур, пакетів, таблиць, функцій) та правил їх використання, що забезпечують управління доступом до об'єктів схеми БД, конфіденційність, цілісність даних та об'єктів, відновлення неправильно змінених або втрачених даних, моніторинг стану, змін, що вносяться до БД, протоколювання дій користувачів.

3. Реалізовані методи та засоби, що ґрунтуються на рекомендаціях різних формальних моделей безпеки, положеннях теорії реляційних БД, технології RLS, потенціалі сучасної блокчейнової моделі, можливостях мов SQL та ММД, а також СКБД, на платформі якої реалізуються схема БД з УБВ, контролюючи зміни збережених елементів бази даних, запобігають їх несанкціонованій зміні уповноваженими суб'єктами та перешкоджають внесенню змін неавторизованими суб'єктами. В результаті збережені дані та програми залишаються правильними, незмінними, неспотвореними та збереженими, що дозволяє укласти, що бази даних, побудовані на основі схеми УБВ, та підтримувані такими механізмами, з точки зору цілісності є захищеними.

4. Як показує практика використання баз даних, побудованих на основі схеми БД з універсальним базисом відношень, для інформаційних систем різних

предметних областей, у проектах яких вимагалося організувати надійне, безпечне зберігання, адаптацію до змін, що відбуваються у ПрО та законодавстві, своєчасну обробку даних, що вони мають досить високий рівень контрольованості доступу до даних, забезпечують необхідну конфіденційність даних, об'єктів схеми БД, їх цілісність, і є досить надійними.

5. Деякі з реалізованих у БД з УБВ методів та засобів забезпечення безпеки, зокрема, метод приховування даних та об'єктів, метод моніторингу цілісності модулів, що зберігаються, можуть знайти широке застосування в базах даних з різними моделями даних.

Результати, представлені у розділі, опубліковані у роботах [13, 14, 123, 213, 234-236].

6. ОЦІНКА БЕЗПЕКИ БАЗИ ДАНИХ З УНІВЕРСАЛЬНИМ БАЗИСОМ ВІДНОШЕНЬ

6.1. Основні прийняті припущення

Перш ніж приступити до оцінювання безпеки баз даних, побудованих на основі схеми з універсальним базисом відношень, та порівняння її із захищеністю традиційних реляційних БД (що не володіють функціональністю та можливостями БД з УББВ) введемо деякі припущення, уточнення та позначення.

1. Вважається, що ймовірності: P_t – виникнення відповідних загроз $(t_1,...,t_{11})$ та P_γ – використання відповідних вразливостей $(\gamma_1,...,\gamma_{18})$ щодо конкретних об'єктів ($o_j \in O$, $j = \overline{1,7}$; див. п. 2.3) для порівнюваних БД однакові.

2. Оцінка залишкового ризику для порівнюваних БД виконується для випадку «Низької» величини збитків / шкоди (рівень збитків – 2; Таблиці 2.1, 2.2) при відносному значенні можливих фінансових втрат складових 0.2 ($L^{\mathrm{UBR}_{\mathrm{quant}}} = L^{\mathrm{RDB}_{\mathrm{quant}}} = 0.2$, де $L^{\mathrm{UBR}_{\mathrm{quant}}}$, $L^{\mathrm{RDB}_{\mathrm{quant}}}$ – чисельні значення (відносні) величини збитків / шкоди L для БД з УББВ та традиційної БД відповідно).

3. В якості заходів забезпечення безпеки / засобів контролю ($w_k \in W$) використовуються деякі узагальнені рішення, пов'язані з деяким процесом, політикою, пристроєм, практикою та іншими діями, спрямованими на зміну ризику, а саме:

– w_1 – засоби, що дозволяють ідентифікувати та усувати неправильно призначені привілеї (такі як: засоби аудиту, утиліти, скрипти, які використовуються адміністратором БД для: агрегування прав користувачів у єдиний репозиторій; збору відомостей про користувачів, їх ролі та поведінку, а також конфіденційності даних; виявлення користувачів, у яких дуже багато привілеїв, та користувачів, які не використовують свої права; перегляду та дозволу/відхилення індивідуальних прав/привілеїв користувачів; відстеження

всіх дій щодо доступу до БД; оповіщення та блокування в реальному часі; виявлення незвичайної активності доступу тощо);

– w_2 – засоби, що надаються СКБД та спеціальні розроблені засоби у схемі БД з УБВ (засоби, що забезпечують ведення спеціального журналу змінених даних, формування даних *спеціальної таблиці користувачів* та деякі інші; див. Розділ 5), що дозволяють ідентифікувати та усунути неправильно призначені привілеї;

– w_3 – засоби, що надаються СКБД та спеціальні розроблені засоби у схемі БД з УБВ (засоби, що забезпечують формування даних *спеціальної таблиці розподілу прав доступу до даних інших користувачів* та деякі інші), що дозволяють ідентифікувати та усунути неправильно призначені привілеї;

– w_4 – засоби, що надаються СКБД та спеціальні розроблені засоби у схемі БД з УБВ (засоби, що забезпечують формування даних *спеціальної таблиці обмеження прав доступу до конкретного елемента даних* та деякі інші), що дозволяють ідентифікувати та усунути неправильно призначені привілеї;

– w_5 – засоби, які дозволяють ідентифікувати та усувати зайві привілеї, виявляти вразливості, відсутні оновлення від постачальників, неактивні облікові записи, змінювати паролі за замовчуванням, належним чином налаштовувати систему аудиту подій, у тому числі для відстеження незвичайної активності доступу користувачів тощо. Своєчасне встановлення оновлень або використання віртуальних патчів (англ. patch) для захисту БД;

– w_6 – засоби, що дозволяють виявляти незвичайну активність доступу користувачів та ускладнюють витік конфіденційних даних з таблиць БД (у тому числі використання засобів для маскування даних, що надаються СКБД, та запропонованих у цій роботі; використання засобів обмеження прав доступу до конкретного елемента даних (див. п. 5.2), реалізованих у БД з УБВ);

– w_7 – засоби, що дозволяють виявляти незвичайну активність доступу користувачів і ускладнюють розкриття коду конфіденційних модулів, що постійно

зберігаються (у тому числі використання засобів для маскування, що надаються СКБД, та запропонованих у даній роботі);

– w_8 – засоби, що дозволяють ідентифікувати та усувати неправильно призначені привілеї, виявляти вразливості, неправильні налаштування, реалізації алгоритму, протоколу автентифікації, невідповідний термін дії сеансу. Своєчасне встановлення критичних оновлень або використання віртуальних патчів для захисту БД від спроб використання вразливостей доти, доки повноцінний та постійний патч не буде розгорнутий;

– w_9 – засоби, що дозволяють контролювати споживання ресурсів (наприклад, через механізм профілів – іменований набір обмежень ресурсів, який може бути використаний користувачем);

– w_{10} – засоби, що дозволяють контролювати цілісність коду тригерів і модулів, що постійно зберігаються (у тому числі засоби, що ґрунтуються на потенціалі сучасної блокчейнової моделі і реалізовані в БД з УБВ);

– w_{11} – використання параметризованих запитів, збережених програм, найменших привілеїв; екранування всіх введених даних користувача; приведення типів даних до типу, який передбачався за логікою роботи програми, перевірка даних, що вводяться користувачем, на відповідність дозволеним послідовностям символів;

– w_{12} – супровід списку «заборонених» функцій, процедур, використання яких слід уникати;

– w_{13} – антивірусне програмне забезпечення;

– w_{14} – засоби, що забезпечують підтримку цілісності даних (як вбудовані в СКБД, так і спеціально розроблені у схемі БД з УБВ), а також реалізують моделі безпеки на основі дискреційної, рольової політики;

– w_{15} – засоби, що реалізують моделі безпеки на основі: дискреційної, мандатної, рольової, атрибутної політики, зокрема специфічні для БД з УБВ (див. Розділ 5);

– w_{16} – спеціальні задокументовані функції діагностування, здатні виявляти причини дефектів, спричинені: неправильністю формування первинних ключів; внесенням некоректних даних; неприпустимим занесенням, видаленням, зміною даних; несанкціонованим зверненням до даних; несанкціонованою зміною схеми БД з УБВ та її об'єктів (у тому числі, використовуючи можливості технології блокчейн), включаючи динамічні аналізатори покриття, реалізовані в інтерпретаторі спеціальної мови моделі даних; спеціальні тригери, з допомогою яких можна здійснювати перехоплення, протоколювання операцій, що виконуються в базі (у тому числі зміни базових таблиць схеми), видавати відповідні інформаційні повідомлення (див. Розділ 5); засоби аудиту СКБД;

– w_{17} – вбудовані в СКБД засоби аудиту, у тому числі спеціальні розроблені засоби у схемі БД з УБВ (засоби, що забезпечують ведення спеціального журналу змінених даних);

– w_{18} – маскування даних таблиць, з урахуванням запропонованих у цій роботі методів;

– w_{19} – маскування збережених об'єктів за допомогою засобів, що надаються СКБД, а також на основі запропонованих у даній роботі методів приховування PSM;

– w_{20} – використання прозорого шифрування (TDE) та застосування криптографічно стійких примітивів, вбудованих у СКБД, а також використання національних стандартів шифрування;

– w_{21} – своєчасне встановлення критичних оновлень, моніторинг криптостійкості використовуваних реалізацій алгоритмів шифрування та випадковості чисел, що формуються генераторами псевдовипадкових чисел, що задовольняють заданим вимогам;

– w_{22} – вбудовані в СКБД засоби адміністратора БД, а також спеціально розроблені скрипти, що спрощують роботу АБД;

– w_{23} – детальна документація на СКБД, БД з описом всіх відповідних її елементів, їх використання, включаючи всі основні компоненти схеми БД з УБВ;

– w_{24} – аудит, блокування відповіді за неправильної кількості запитів.

6.2. Порівняльний аналіз захищеності баз даних, побудованих за традиційною технологією та на основі універсального базису відношень

Відповідно до розробленого у розділі 2 методу і прийнятими вище припущеннями, оцінимо потенційну величину захищеності БД з універсальним базисом відношень і порівняємо її із захищеністю традиційних реляційних БД. Для чого, на підставі певного переліку основних об'єктів, загроз, вразливостей (п. 2.3), наявних заходів забезпечення безпеки (п. 6.1), узагальнюючи досвід експлуатації та побудови систем захисту для реляційних баз даних та БД з УББВ, спочатку визначимо значення відповідних компонентів бар'єрів безпеки ($P_l = f(P_{t_i}, P_{\gamma_\psi})$, L_l, R_l), співвідносячи їх із четвіркою відповідних найбільш значимих (з погляду питань, що розглядаються) елементів бар'єру $b_l = (t_i, \gamma_\psi, o_j, w_k)$ у базовій системі захисту. На рисунку 6.1 представлений фрагмент моделі системи захисту бази даних у вигляді орієнтованого графа.

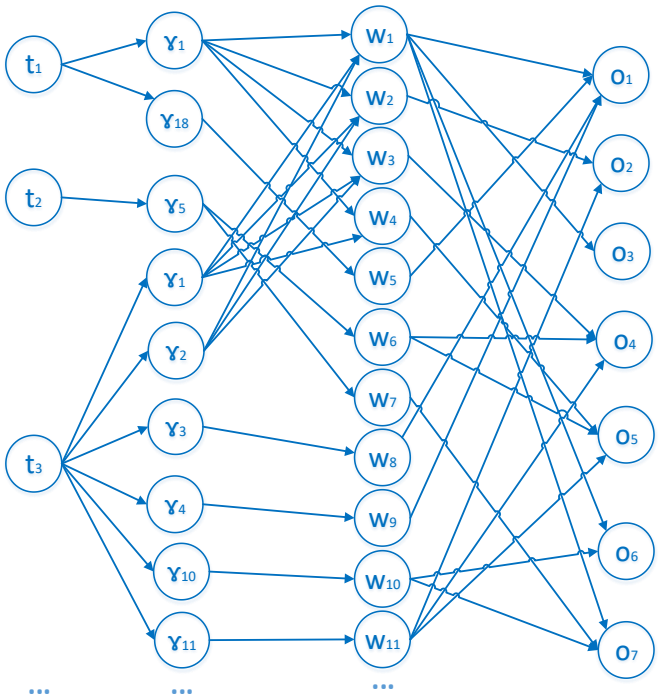


Рисунок 6.1. Фрагмент моделі системи захисту БД у вигляді графа

У таблиці 6.1 наведено фрагмент результатів оцінки основних компонентів бар'єрів безпеки та стійкості кожного з них (у Додатку Е наведена повна таблиця цих результатів оцінки).

Таблиця 6.1 – Фрагмент результатів оцінки основних компонентів бар'єрів безпеки

№ бар'єру	Загроза (t)	$\frac{P_t^{\text{verbal}}}{P_t^{\text{quant}}}$	Вразливість (γ)	$\frac{P_\gamma^{\text{verbal}}}{P_\gamma^{\text{quant}}}$	Захід безпеки (w)	$\frac{R^{\text{UBR}_{\text{verbal}}}}{R^{\text{UBR}_{\text{quant}}}}$	$\frac{R^{\text{RDB}_{\text{verbal}}}}{R^{\text{RDB}_{\text{quant}}}}$	Об'єкт (o)	Rr^{UBR}	Rr^{RDB}
1.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
2.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
3.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
4.	t_1	«Н» / 0.1	γ_1	«С» / 0.5	w_4	«В» / 0.8	«Н» / 0	o_5	0.002	0.01
5.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_3	0.008	0.008
6.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
7.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
8.	t_1	«С» / 0.4	γ_{18}	«С» / 0.5	w_5	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
9.	t_2	«С» / 0.4	γ_5	«В»/ 0.85	w_6	«В» / 0.8	«С» / 0.6	o_4	0.0136	0.0272
10.	t_2	«С» / 0.4	γ_5	«В»/ 0.85	w_6	«В» / 0.8	«С» / 0.6	o_5	0.0136	0.0272
11.	t_2	«С» / 0.4	γ_5	«В»/ 0.85	w_7	«В» / 0.8	«С» / 0.6	o_7	0.0136	0.0272
12.	t_3	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
13.	t_3	«С» / 0.4	γ_1	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
14.	t_3	«С» / 0.4	γ_1	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
15.	t_3	«Н» / 0.1	γ_1	«С» / 0.5	w_4	«В» / 0.8	«Н» / 0	o_5	0.002	0.01
16.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
17.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
18.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
19.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_3	0.008	0.008
20.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
21.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
22.	t_3	«С» / 0.4	γ_3	«С» / 0.5	w_8	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
23.	t_3	«С» / 0.4	γ_4	«С» / 0.5	w_9	«С» / 0.4	«С» / 0.4	o_1	0.024	0.024
24.	t_3	«С» / 0.4	γ_{10}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_6	0.004	0.024
25.	t_3	«С» / 0.4	γ_{10}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_7	0.004	0.024
26.	t_3	«С» / 0.4	γ_{11}	«С» / 0.5	w_{11}	«В» / 0.8	«В» / 0.8	o_2	0.008	0.008

Де P_t^{verbal} – вербальне значення лінгвістичної змінної – ймовірність виникнення загрози P_t; P_t^{quant} – чисельне значення ймовірності P_t; P_γ^{verbal} – вербальне значення лінгвістичної змінної – ймовірність використання вразливості P_γ; P_γ^{quant} – чисельне значення ймовірності P_γ; $R^{\text{UBR}_{\text{verbal}}}$ – вербальне значення лінгвістичної змінної – ступінь опірності засобу захисту (заходу безпеки) R ($R = 1 - P^{ov}$) для БД з УБВ; $R^{\text{UBR}_{\text{quant}}}$ – чисельне значення ступеня опірності засобу захисту R для БД з УБВ; $R^{\text{RDB}_{\text{verbal}}}$ – вербальне значення лінгвістичної змінної – ступінь опірності засобу захисту R для традиційної БД; $R^{\text{RDB}_{\text{quant}}}$ – чисельне значення ступеня опірності засобу захисту R для традиційної БД; Rr^{UBR} –

чисельне значення величини залишкового ризику для БД з УБВ; Rr^{RDB} – чисельне значення величини залишкового ризику для традиційної БД.

Відповідно до отриманих результатів оцінок основних компонент бар'єрів безпеки та величин залишкового ризику Rr_l (наведених у Таблиці Е.1 Додатка Е та на рисунку 6.2) для кожного з них, при заданих припущеннях (п. 6.1), відповідно до виразу (2.3), були розраховані та представлені на рисунку 6.3 у вигляді діаграми значення величин захищеності традиційних баз даних ($S^{\mathrm{RDB}} = \sum_{\forall b_l \in B} \frac{1}{P_l^{\mathrm{RDB}} L_l^{\mathrm{RDB}} (1 - R_l^{\mathrm{RDB}})}$) і БД з УБВ ($S^{\mathrm{UBR}} = \sum_{\forall b_l \in B} \frac{1}{P_l^{\mathrm{UBR}} L_l^{\mathrm{UBR}} (1 - R_l^{\mathrm{UBR}})}$). При цьому слід зазначити, що в БД з УБВ були реалізовані відповідні спеціальні методи та засоби забезпечення безпеки, а традиційні БД не мали відповідних реалізацій.

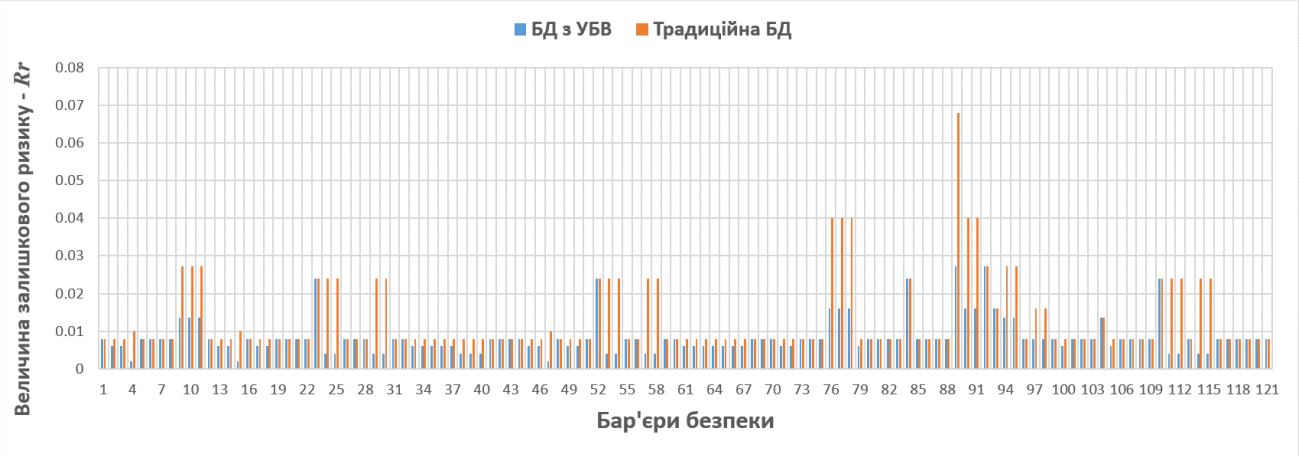


Рисунок 6.2. Значення оцінок величини залишкового ризику кожного з бар'єрів безпеки для порівнюваних БД

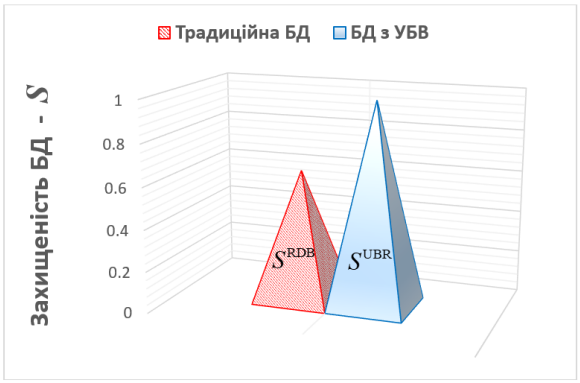


Рисунок 6.3. Результат порівняльного аналізу величин захищеності традиційних баз даних і БД з УБВ

На підставі отриманих результатів було зроблено загальний висновок про більшу ефективність пропонованих у рамках схеми БД з УБВ заходів безпеки у

порівнянні з наявними рішеннями, реалізованими в рамках традиційних реляційних БД. З урахуванням отриманої кількісної оцінки, використання запропонованих заходів дозволить підвищити ефективність / результативність захисту (захищеність як зворотної величини сумарного залишкового ризику – вираз (2.3)) баз даних, побудованих на основі схеми з універсальним базисом відношень, більш ніж у 1.5 рази ($\frac{S^{\text{UBR}}}{S^{\text{RDB}}} \approx 1.5$).

Аналізуючи різні контрзаходи, спрямовані на забезпечення безпеки, слід зазначити, що багато проблем із захистом даних, що зберігаються в базі даних, часто виникають не через відсутність досліджень, наявність теоретично розроблених моделей, методів, а через недостатню безпеку у відповідній конкретній реалізації бази даних або застосунків, що працюють з нею. У цьому сенсі БД з УБВ мають перевагу, тому що вони не проектуються щоразу з нуля і не піддаються істотній модифікації при реінжинірингу, у тому числі в частині, що стосується їх безпеки. Інваріантна до різних предметних областей, заздалегідь розроблена, включаючи спеціальні заходи (у вигляді відповідних методів, реалізованих об'єктів), що забезпечують безпеку, і протестована схема БД одноразово інсталюється на платформу деякої СКБД. При розширенні набору об'єктів, подій, характеристик об'єктів, подій, параметрів об'єктів предметних областей, що моделюється, в БД не створюються нові базові відношення, атрибути, ключі та інші об'єкти схеми, в тому числі що забезпечують її безпеку, а просто додається новий запис в одне з існуючих базових відношень схеми. Це дає можливість при реінжинірингу баз даних, побудованих на основі такої схеми, спростити процес їх адаптації до динамічних змін предметних областей.

6.3. Висновки за розділом

1. Узагальнюючи досвід експлуатації та побудови систем захисту для реляційних БД та баз даних, побудованих на основі схеми з універсальним базисом відношень (використовуючи розроблені заходи забезпечення безпеки, що спираються на загальні формальні моделі управління доступом, забезпечення цілісності даних, криптографію, стеганографію, положення теорії реляційних БД,

технології RLS, сучасної блокчейнової моделі, можливості SQL та спеціальної мови моделі даних), виходячи з аналізу виявлених актуальних загроз, слабких місць (вразливостей) в об'єктах захисту сучасних традиційних БД, використовуючи розроблений у роботі метод, з метою визначення ступеня забезпечення безпеки було проведено оцінку та порівняльний аналіз захищеності баз даних, побудованих за вказаними вище технологіями.

2. Проведений порівняльний аналіз дозволив зробити загальний висновок про більшу ефективність запропонованих та реалізованих у рамках схеми БД з УБВ заходів щодо забезпечення безпеки порівняно з наявними, реалізованими в рамках традиційних реляційних баз даних. Наведені при заданих припущеннях кількісні розрахунки свідчать, що використання запропонованих рішень дозволить підвищити ефективність / результативність захисту баз даних, побудованих з урахуванням схеми з універсальним базисом відношень, більш ніж 1.5 разу щодо традиційних реляційних БД.

3. Бази даних, побудовані на основі інваріантної до різних предметних областей схеми з універсальним базисом відношень, щодо традиційно проектованих РБД менш залежні від проблеми захисту даних, пов'язаної з реалізацією бази даних і застосунків, що працюють з нею, оскільки БД не проектуються щоразу з нуля, а на платформу деякої СКБД інсталюється вже протестована, налагоджена, перевірена часом готова схема. До того ж бази даних з УБВ, у тому числі об'єкти схеми, що забезпечують безпеку БД, не піддаються суттєвій модифікації при реінжинірингу як це властиво традиційним РБД.

4. Наведені в роботі зрозумілі, несуперечливі результати оцінки безпеки баз даних, спроектованих за різними технологіями, з різними заходами захисту, свідчать про об'єктивність розробленого методу оцінювання основних компонентів бар'єрів безпеки та захищеності бази даних у цілому. Все це є дуже важливим, і насамперед, з погляду можливості та доцільності практичного застосування даного методу надалі для оцінювання та порівняння безпеки різних БД.

Основні питання та результати, що стосуються представлених у розділі, опубліковані в роботах [12-15, 59, 123, 151, 213, 234-236].

ВИСНОВКИ

У результаті виконання дисертаційних досліджень вирішено важливе науково-прикладне завдання, яке полягає в розробці моделей, методів і засобів, що дозволяють підвищити захищеність баз даних, побудованих на основі схеми з універсальним базисом відношень.

Проведені дослідження дозволяють зробити такі висновки.

1. Аналіз ключових проблем, сучасного стану та розвитку технологій баз, сховищ даних показав, що існуючі підходи до забезпечення безпеки даних, що зберігаються і оброблюються в них, не повною мірою відповідають відповідним вимогам різних міжнародних, вітчизняних стандартів та інших законодавчих актів у галузі інформаційної безпеки. Без необхідного захисту баз та сховищ даних, нові інформаційні технології внаслідок помилкових дій з боку законних користувачів та власників, а більшою мірою протиправних дій з боку зловмисників, здатні порушити не лише приватне життя людей, а й діяльність великих організацій. Тому в ситуації, що склалася, беручи до уваги сучасний стан розвитку технологій баз, сховищ даних, науково-практичні досягнення в галузі ІБ, кваліфікацію зловмисників, доцільним є перегляд підходу до вирішення проблеми управління даними та забезпечення їх безпеки, результатом якого були б певні моделі, методи, прийоми, засоби, актуальні як у теоретичному, так і в прикладному аспектах. Усе це визначило необхідний склад, логічну послідовність досліджень, що проводилися, і дозволило отримати нові наукові та практичні результати.

2. Удосконалено модель системи безпеки з повним перекриттям Клементса–Гофмана, яка відрізняється від відомої розширеним та конкретизованим для баз даних складом компонент, що дозволяє більш адекватно оцінювати ймовірність небажаного інциденту (реалізації загрози) та захищеність бази даних у цілому. Розширення здійснено за рахунок доповнення моделі множиною вразливостей об'єктів як окремо об'єктивно існуючої категорії. Це означає, що ймовірність небажаного інциденту розглядається як двофакторна складова, в якій один із факторів відображає мотиваційну складову виникнення загрози, а другий

враховує наявні вразливості. В якості інтегрального показника безпеки бази даних було визначено величину зворотну сумарному залишковому ризику, який по суті є мірою незахищеності активу. Це дозволило кількісно оцінювати безпеку баз даних.

3. Удосконалено метод маскування MOBAT, який відрізняється від відомого, можливістю обфускації даних, шляхом математичних перетворень на основі обчислення операцій за модулем, що застосовуються до елементів даних не тільки числового, а й широко поширеного в базах даних рядкового типу, що дозволяє суттєво розширити охоплення різноманітних маскованих з метою утруднення реалізації зловмисником загрози умовиводу даних БД. Даний метод доцільно використовувати при невеликих розмірностях рядків, що маскуються, і за умови можливості зміни типу і розмірності даних.

4. Розроблено метод маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних, що відрізняється від відомих оригінальним підходом до процесу перемішування з можливістю випадкової заміни елементів даних різного типу всередині заданого поля рядка та використання технології динамічного маскування, що дозволяє при менших обчислювальних витратах на перетворення і без зміни формату вихідних даних забезпечити ефективне приховування даних, яке ускладнює реалізацію загрози логічного висновку. Розроблений метод, орієнтований на ускладнення реалізації загрози логічного висновку, дозволяє зменшити час на відповідні операції перетворення на (10-17)% щодо методу класичного шифрування, при цьому його застосування не тягне за собою зміну формату та збільшення розмірності даних, що зберігаються. Даний метод може бути також використаний у невиробничих базах даних, розширюючи можливості так званого статичного маскування даних.

5. Розроблено метод приховування коду збережених у базі даних програм, який на відміну від відомих, дозволяє за рахунок випадкової перестановки (що спирається на сучасний варіант алгоритму тасування Фішера-Йейтса) символів коду з можливою заміною кожного з них на інший випадково вибраний із стандарту Unicode забезпечити більше ефективний (якій вимагає значно більших

обчислювальних витрат) захист коду від його розкриття зловмисником, при цьому гарантуючи цілісність коду.

6. Вперше запропоновано метод моніторингу збережених програм, що ґрунтується на можливостях технології блокчейн, який на відміну від відомих дозволяє за рахунок використання створеної зумовленої структури, правил формування первинного та наступних блоків у блокчейновому ланцюжку, організації зберігання цієї структури в рамках реляційної моделі даних, способів обчислення кореня геш-дерева, суворо контролювати набір програм БД, їх цілісність, справжність при менших обсягах збережених для цього даних і необхідних ресурсів процесора. Дослідження показали, що для того, щоб переконатися в цілісності коду збережених програм, використовуючи відомий метод контрольних сум, потрібно було б виконати процедури гешування та цифрового підпису із збереженням відповідних даних для кожного конкретного PSM у конкретній схемі БД. Однак навіть у цьому випадку, не забезпечувався б контроль всього набору PSM в цілому, і віддаленому легітимному користувачеві БД було б важко переконатися в цілісності збережених програм, що використовуються в його застосунках.

7. Розроблені в процесі створення схеми БД з УБВ спеціальні заходи у вигляді відповідних методів, реалізованих об'єктів (тригерів, процедур, пакетів, таблиць, функцій) та правил їх використання підвищують безпеку таких БД, забезпечуючи високий рівень контрольованості доступу до даних (аж до конкретного елемента), необхідну конфіденційність, цілісність даних та об'єктів схеми БД, на відміну від традиційних РБД, що не володіють подібними заходами та функціональністю. Слід зазначити, що запропоновані та реалізовані в роботі рішення можуть бути використані при розробці нових та вдосконаленні існуючих інструментальних засобів захисту у складі сучасних СКБД. При цьому деякі з реалізованих у БД з УБВ методів та засобів забезпечення безпеки, зокрема, метод приховування даних та об'єктів, метод моніторингу цілісності модулів, що зберігаються, можуть знайти широке застосування в базах даних з різними моделями даних.

8. Удосконалено метод оцінювання основних компонент бар'єрів безпеки та захищеності бази даних в цілому, який на відміну від відомих, за рахунок комплексування вдосконаленої моделі Клементса–Гофмана, введеного інтегрального показника безпеки, положень теорії нечітких множин та ризику, дозволяє адаптуватися до нових умов функціонування та прозоро, комплексно та кількісно оцінювати безпеку баз даних з різними моделями даних. Відповідно до даного методу було виконано порівняльний аналіз захищеності реляційних баз даних, спроектованих за різними технологіями, результати якого свідчать про більшу ефективність запропонованих та реалізованих у рамках схеми БД з УБВ заходів щодо забезпечення безпеки порівняно з наявними, реалізованими в рамках традиційних реляційних баз даних. А саме, використання запропонованих рішень, реалізованих у БД з УБВ, дозволяє підвищити ефективність / результативність захисту баз даних, побудованих на основі схеми з універсальним базисом відношень, більш ніж у 1.5 рази щодо традиційних реляційних БД.

9. Обґрунтованість і достовірність наукових результатів, висновків і рекомендацій, сформульованих у дисертаційній роботі, забезпечується: коректним використанням відомих положень теорії множин, відношень, графів, математичної логіки, математичної статистики, формальних моделей безпеки, комплексним урахуванням набору взаємопов'язаних об'єктів БД, загроз, вразливостей, заходів забезпечення безпеки, як відповідних елементів бар'єрів безпеки у базовій системі захисту, несуперечливістю відомим результатам, науковою апробацією результатів дисертаційних досліджень на науково-технічних і науково-практичних конференціях різного рівня.

10. Результати дисертаційного дослідження можуть бути використані організаціями та компаніями, що займаються проектуванням та розробкою систем та засобів захисту для баз даних, оцінкою їх захищеності.

11. Сукупність отриманих у дисертаційній роботі наукових результатів, позитивна оцінка їх обґрунтованості, достовірності, наукової та практичної значущості дозволяють вважати сформульовані завдання – вирішеними, а поставлену мету – підвищення ефективності захисту баз даних, побудованих на основі схеми з універсальним базисом відношень, шляхом розробки та застосування моделей, методів та засобів забезпечення їх безпеки – досягнутою.

ПЕРЕЛІК ПОСИЛАНЬ

1. Abadi D., Agrawal R., Ailamaki A., Balazinska M., Bernstein P. A., Carey M. J., et al. The Beckman Report on Database Research. *ACM SIGMOD Record*. 2014. № 43(3). P. 61–70.

2. Abadi D., Ailamaki A., Andersen D., P. Bailis P., et al. The Seattle Report on Database Research. *ACM SIGMOD Record*. 2019. № 48. P. 44–53.

3. Porter H. Nine sacked for breaching core ID card database. The Guardian 10 Aug 2009. URL: https://www.theguardian.com/commentisfree/henryporter/2009/aug/10/id-card-database-breach (дата звернення: 17.12.2023).

4. Adrian M., Greenwald R., Cook H., Gu X. Critical Capabilities for Cloud Database Management Systems for Operational Use Cases. ID G00763650. URL: https://www.gartner.com/doc/reprints?id=1-2C00XM3R&ct=221214&st=sb (дата звернення: 20.12.2023).

5. Chaos Manifesto 2013: Think Big, Act Small online version. The Standish Group. URL: https://www.immagic.com/eLibrary/ARCHIVES/GENERAL/GENREF/S130301C.pdf (дата звернення: 17.12.2023).

6. Standish Group 2015 Chaos Report – Q&A with Jennifer Lynch. URL: https://www.infoq.com/articles/standish-chaos-2015 (дата звернення: 17.12.2023).

7. Connolly T. M., Begg C. E. Database systems: a practical approach to design, implementation, and management. Sixth edition. Harlow, Essex, England: Pearson Education Limited, 2015. 1329 p.

8. DB-Engines Ranking. URL: https://db-engines.com/en/ranking (дата звернення: 17.12.2023).

9. TOPDB Top Database index. URL: https://pypl.github.io/DB.html (дата звернення: 17.12.2023).

10. Єсін В. І. Моделі і методи забезпечення адаптованості баз даних до змін в предметній області: дис. ... д-ра техн. наук: 05.13.06. Харків, 2016. 412 с.

11. Есин В. И. Инвариантная к предметным областям схема базы данных и ее отличительные особенности. *Радіотехніка*. 2018. № 193. С. 133–142.

12. Есин В. И., Вилигура В. В. Метод разработки баз данных, легко адаптируемых к изменениям в предметной области. *Радіотехніка*. 2018. № 195. С. 184–192.

13. Yesin V. I., Vilihura V. V. Method for development of databases easily adaptable to variations in the subject domain. *Telecommunications and Radio Engineering*. 2019. № 78(7). P. 595–605.

14. Yesin V. I., Karpinski M., Yesina M. V., Vilihura V. V. Formalized representation for the data model with the universal basis of relations. *International Journal of Computing*. 2019. № 18(4). P. 453–460.

15. Yesin V. I., Karpinski M., Yesina M. V., Vilihura V. V., Veselska O., Wieclaw L. Approach to Managing Data From Diverse Sources. Proceedings of the 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 18-21 September, 2019. Metz. France. № 1. P. 1–6.

16. Gartner Inc. Forecast: Information Security and Risk Management, Worldwide, 2019-2025, 2Q21 Update. URL: https://www.gartner.com/en/documents/4002908/forecast-information-security-and-risk-management-worldw (дата звернення: 17.12.2023).

17. General Data Protection Regulation GDPR. URL: https://gdpr-info.eu/ (дата звернення: 17.12.2023).

18. Health insurance portability and accountability act of 1996. URL: https://www.govinfo.gov/content/pkg/CRPT-104hrpt736/pdf/CRPT-104hrpt736.pdf (дата звернення: 17.12.2023).

19. Data Protection in a Multi-Cloud World. Global Data Protection Index: Global Results for Cloud Infrastructure. URL: https://leadcomm.com.br/wp-content/uploads/2020/03/global-data-protection-index-2020-snapshot.pdf (дата звернення: 17.12.2023).

20. Dell Technologies Research: Cyber Attacks and Disruptive Events Are on the Rise, Affecting 82% of Organizations Surveyed. URL: https://www.prnewswire.com/news-releases/dell-technologies-research-cyber-attacks-and-disruptive-events-are-on-the-rise-affecting-82-of-organizations-surveyed-301021374.html (дата звернення: 17.12.2023).

21. Global Data Protection Index 2022 Key Findings. October 2022. URL: https://www.delltechnologies.com/asset/en-nz/products/data-protection/industry-market/global-data-protection-index-key-findings.pdf. (дата звернення: 17.12.2023).

22. Cisco 2018 Annual Cybersecurity Report URL: https://www.cisco.com/c/dam/m/digital/elq-cmcglobal/witb/acr2018/acr2018final.pdf?dtid=odicdc000016&ccid=cc000160&oid=anrsc005679&ecid=8196&elqTrackId=686210143d34494fa27ff73da9690a5b&elqaid=9452&elqat=2 (дата звернення: 17.12.2023).

23. Fortinet. Q4 2019. Quarterly Threat Landscape Report. URL: https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/ko_kr/threat-report-q4-2019.pdf (дата звернення: 17.12.2023).

24. Panda Security. PandaLabs Annual Report 2018. URL: https://partnernews.pandasecurity.com/uk/src/uploads/2018/12/PandaLabs-2018_Annual_Report-uk.pdf (дата звернення: 17.12.2023).

25. Fortinet. Q4 2018. Quarterly Threat Landscape Report. URL: https://www.fortinet.com/content/dam/fortinet/assets/threat-reports/threat-report-q4-2018.pdf (дата звернення: 17.12.2023).

26. Есин В. И., Вилигура В. В. Метод преобразования унаследованных баз данных в базу данных с универсальным базисом отношений. *Прикладна радіоелектроніка*. 2018. № 17(3, 4). С. 176–181.

27. Date C. J. An Introduction to Database Systems, 8th Edition. Pearson. Addison-Wesley, 2004. 983 p.

28. Sadalage P. J., Fowler M. NoSQL distilled: a brief guide to the emerging world of polyglot persistence. Pearson Education, 2013. 192 p.

29. Meier A., Kaufmann M. SQL & NoSQL databases. Databases Models, Languages, Consistency Options and Architectures for Big Data Management. Springer Fachmedien Wiesbaden, 2019. 228 p.

30. Harrison G. Next Generation Databases: NoSQL, NewSQL, and Big Data. Apress, 2015. 235 p.

31. Pavlo A., Aslett M. What's really new with NewSQL? *ACM Sigmod Record*. 2016. № 45(2). P. 45–55.

32. Garcia-Molina H., Ullman J. D., Widom J. Database Systems. The Complete Book, 2th ed. Pearson Prentice Hall, 2009. 1203 p.

33. Єсін В. І., Вілігура В. В. Основні категорії NewSQL баз даних та їх особливості. *Радіотехніка: Всеукр. міжвід. наук.-техн. зб.* № 211. 2022. С. 37–66.

34. Закон України «Про захист персональних даних» від 01.06.2010 №2297-VI. *Відомості Верховної Ради України (ВВР)*. 2010. № 34, Ст. 481. Із змінами, внесеними згідно із Законами № 4452-VI від 23.02.2012, ВВР. 2012. № 50. Ст. 564. № 5491-VI від 20.11.2012 ВВР. 2013. № 51. Ст. 715. № 245-VII від 16.05.2013 ВВР. 2014. № 12. Ст.178. № 383-VII від 03.07.2013 ВВР. 2014. № 14. Ст.252. № 1170-VII від 27.03.2014 ВВР. 2014. № 22. Ст.816. № 1262-VII від 13.05.2014 ВВР. 2014. № 27. Ст.914. № 316-VIII від 09.04.2015 ВВР. 2015. № 26. Ст.218. № 675-VIII від 03.09.2015 ВВР. 2015. № 45. Ст.410. № 1774-VIII від 06.12.2016 ВВР. 2017. № 2. Ст.25. № 2168-VIII від 19.10.2017 ВВР. 2018. № 5. Ст.31. № 324-IX від 03.12.2019. ВВР. 2020. № 11. Ст.63. № 524-IX від 04.03.2020. ВВР, 2020. № 38. Ст.279. № 1357-IX від 30.03.2021. ВВР, 2021. № 29. Ст.234. № 1667-IX від 15.07.2021. № 1723-IX від 08.09.2021.

35. Payment Card Industry (PCI) Data Security Standard. Requirements and Security Assessment Procedures Version 3.2.1. 2018. URL: https://www.cuny.edu/wp-content/uploads/sites/4/page-assets/about/administration/offices/budget-and-finance/services/payment-card-industry-compliance/PCI_DSS_v3-2-1.pdf (дата звернення: 17.12.2023).

36. Payment Card Industry (PCI) Data Security Standard. Requirements and Testing Procedures Version 4.0. 2022. URL:

https://www.pcisecuritystandards.org/documents/PCI-DSS-v4_0.pdf (дата звернення: 17.12.2023).

37. Franklin M., Halevy A., Maier D. From databases to dataspaces: A new abstraction for information management. *ACM Sigmod Record*. 2005. № 34(4). P. 27–33.

38. Sandhu R. S., Jajodia S. Data and database security and controls. Handbook of information security management, Auerbach Publishers, 1993. P. 481–499.

39. Bertino E., Sandhu R. Database security-concepts, approaches, and challenges. *IEEE Transactions on Dependable and secure computing*. 2005. № 2(1). P. 2–19.

40. Lesov P. Database security: a historical perspectiv. URL: http://arxiv.org/ftp/arxiv/papers/1004/1004.4022.pdf (дата звернення: 17.12.2023).

41. Gertz M., Gandhi M. Security re-engineering for databases: concepts and techniques. Handbook of Database Security. Springer, Boston, MA, 2008. P. 267–296.

42. Pfleeger C. P., Pfleeger S. L., Margulies J. Security in Computing. Fifth Edition. Prentice Hall, 2015. 944 p.

43. Burtescu E. Database security – attacks and control methods. *Journal of applied quantitative methods*. 2009. № 4(4). P. 449–454.

44. Kulkarni S., Urolagin S. Review of Attacks on Databases and Database Security Techniques. *International Journal of Engineering Technology and Database Security Techniques Research*. 2012. № 2(11). P. 2250–2459.

45. Mayer-Schonberger V., Cukier K. Big Data: A Revolution That Will Transform How We Live, Work and Think. Eamon Dolan. Houghton Mifflin Harcourt, 2013. 242 p.

46. Завгородний В. И. Комплексная защита информации в компьютерных системах. М.: Логос, 2001. 264 с.

47. Yesin V. I., Vilihura V. V. Some approach to data masking as means to counter the inference threat. *Radiotekhnika : All-Ukr. Sci. Interdep. Mag.*, 2019. № 198. P. 113–130.

48. Near J., Darais D., Boeckl K. Differential Privacy for Privacy-Preserving Data Analysis: An Introduction to our Blog Series. URL:

https://www.nist.gov/blogs/cybersecurity-insights/differential-privacy-privacy-preserving-data-analysis-introduction-our (дата звернення: 17.12.2023).

49. Garfinkel S., Abowd J. M., Martindale C. Understanding database reconstruction attacks on public data. *Communications of the ACM*. 2019. № 62(3). P. 46–53.

50. Data Masking: What You Need to Know. A Net 2000 Ltd. White Paper. URL: https://studylib.net/doc/18574198/data-masking--what-you-need-to-know (дата звернення: 17.12.2023).

51. Oracle. Data Masking and Subsetting Guide. URL: https://docs.oracle.com/en/database/oracle/oracle-database/12.2/dmksb/oracle-data-masking-and-subsetting-users-guide.pdf (дата звернення: 17.12.2023).

52. Santos R. J., Bernardino J., Vieira M. A data masking technique for data warehouses. Proceedings of the 15th Symposium on International Database Engineering & Applications, IDEAS11, Lisbon, Portugal, 21–23 September 2011, 2011. P. 61–69.

53. Archana R. A., Ravindra S. Hegadi, Manjunath T. N. A Big Data Security using Data Masking Methods. *Indones. J. Electr. Eng. Comput. Sci*. 2017. № 7. P. 449–456.

54. Vishnu B., Manjunath T. N., Hamsa C. An Effective Data Warehouse Security Framework. Proceedings of the IJCA National Conference on Recent Advances in Information Technology, Solapur, India, 15–16 February 2014, 2014. P. 33–37.

55. Larsonk K. S., Boukari S. An Improved Data Masking Security Solution Using Modulus Based Technique (MOBAT) for Data Warehouse System. *Int. J. Sci. Eng. Appl*. 2020. № 9. P. 68–78.

56. Marino S., Zhou N., Zhao Y., Wang L., Wu Q., Dinov I. D. HDDA: DataSifter: Statistical obfuscation of electronic health records and other sensitive datasets. *J. Stat. Comput. Simul*. 2019. № 89. P. 249–271.

57. Bellare M., Hoang V. T., Tessaro S. Message-Recovery Attacks on Feistel-Based Format Preserving Encryption CCS'16. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016, 2016. P. 444–455.

58. Dworkin M. Recommendation for Block Cipher Modes of Operation Methods for Format-Preserving Encryption. Draft NIST Special Publication 800-38G Revision 1. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38G.pdf (дата звернення: 17.12.2023).

59. Yesin V., Karpinski M., Yesina M., Vilihura V., Warwas K. Hiding the Source Code of Stored Database Programs. *Information*. 2020. № 11(12). 576.

60. Chapple M., Stewart J. M., Gibson D. CISSP®Certified Information Systems Security Professional Official Study Guide, 8th ed. Sybex, John Wiley & Sons Inc.: Indianapolis, IN, USA, 2018. 1050 p.

61. Chapple M., Stewart J. M., Gibson D. CISSP: certified information systems security professional official study guide. 9th Edition. Sybex, John Wiley & Sons, Inc.: Indianapolis, Indiana, 2021. 1248 p.

62. Jueneman R. R. Integrity controls for military and commercial applications. Proceedings of the 1988 Fourth Aerospace Computer Security Applications, 12-16 Sept. 1988. IEEE. Orlando, FL, USA, 1988. P. 298–322.

63. Гайдамакин Н. А. Теоретические основы компьютерной безопасности. Екатеринбург: Изд-во Уральского университета, 2008. 212 с.

64. Microsoft. Create procedure (Transact-SQL). URL: https://docs.microsoft.com/en-us/sql/t-sql/statements/create-procedure-transact-sql?view=sql-server-ver15 (дата звернення: 17.12.2023).

65. McLaughlin M. Oracle Database 12c PL/SQL Programming. McGraw-Hill Education: New York, NY, USA, 2014. 1154 p.

66. Feuerstein, S., Pribyl B. Oracle PL/SQL Programming. 6th ed. O'Reilly Media: Sebastopol, CA, USA, 2014. 1392 p.

67. Scheffer A. Unwrapping 10G Wrapped PL/SQL. URL: https://technology.amis.nl/2009/02/03/unwrapping-10g-wrapped-plsql/ (дата звернення: 17.12.2023).

68. Lambrechts M. Unwrapping Wrapped PLSQL in 10g, 11g and 12c. URL: http://marcel.vandewaters.nl/oracle/security/unwrapping-wrapped-plsql-in-10g-and-11g (дата звернення: 17.12.2023).

69. White P. The Internals of WITH ENCRYPTION. URL: https://sqlperformance.com/2016/05/sql-performance/the-internals-of-with-encryption (дата звернення: 17.12.2023).

70. Savola R. A security metrics taxonomization model for software-intensive systems. *Journal of Information Processing Systems*. № 5(4). 2009. P. 197–206.

71. Jansen W., Gallagher P. D. NISTIR 7564. Directions in Security Metrics Research. URL: https://nvlpubs.nist.gov/nistpubs/legacy/ir/nistir7564.pdf (дата звернення: 17.12.2023).

72. Information Technology Security Evaluation Criteria (ITSEC): Provisional evaluation criteria. Document COM(90) 314, Version 1.2. Commission of the European Communities. URL: https://op.europa.eu/en/publication-detail/-/publication/b2ef2a4f-f33c-4561-82d2-d8de49bf9602 (дата звернення: 17.12.2023).

73. ISO/IEC 21827:2008 Information technology. Security techniques. Systems Security Engineering. Capability Maturity Model® (SSE-CMM®). URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:21827:ed-2:v1:en (дата звернення: 17.12.2023).

74. Savola R. M. Towards Measurement of Security Effectiveness Enabling Factors in Software Intensive Systems. *Lecture Notes on Software Engineering*. 2014. № 2(1). P. 104–109.

75. ISO/IEC 15408-1:2022. Information security, cybersecurity and privacy protection Evaluation criteria for IT security. Part 1: Introduction and general model. URL: https://www.iso.org/standard/72891.html (дата звернення: 17.12.2023).

76. ISO/IEC 27001:2022. Information security, cybersecurity and privacy protection. Information security management systems. Requirements. URL: https://www.iso.org/standard/27001 (дата звернення: 17.12.2023).

77. ISO/IEC 27004:2016. Information technology. Security techniques. Information security management. Monitoring, measurement, analysis and evaluation. URL: https://www.iso.org/standard/64120.html (дата звернення: 17.12.2023).

78. Latham D. C. Department of defense standard. Department of defense. Trusted computer system. Evaluation criteria. URL: http://csrc.nist.gov/publications/history/dod85.pdf (дата звернення: 17.12.2023).

79. Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model. Version 3.1 Revision 5 CCMB-2017-04-001. URL: https://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R5.pdf (дата звернення: 17.12.2023).

80. Анищенко В. В., Криштофик А. М. Методы оценки эффективности защиты активов в объектах информационных технологий. *Информатика*. 2004. № 3(03). С. 95–105.

81. Маслова Н. А. Методы оценки эффективности систем защиты информационных систем. *Штучний інтелект*. 2008. № 4. С. 253–264.

82. Домарєв Д. В., Домарєв В. В., Прокопенко С. Д. Методика оцінювання захищеності інформаційних систем за допомогою СУІБ «Матриця». *Захист інформації*. 2013. № 15(1). С. 80–86.

83. Домарев В. В. Безопасность информационных технологий. Системный подход. К.: ООО ТИД «ДС», 2004. 992 с.

84. Hoffmann R., Kiedrowicz M., Stanik J. Evaluation of information safety as an element of improving the organization’s safety management. Proceedings of the 20th International Conference on Circuits, Systems, Communications and Computers (CSCC 2016). MATEC Web of Conferences, Corfu Island, Greece, July 14-17, 2016. 76, 04011.

85. Hoffman L. J., Clements D. Fuzzy computer security metrics: A preliminary report. Electronics research laboratory. College of Engineering University of California. 1977. URL: https://www2.eecs.berkeley.edu/Pubs/TechRpts/1977/ERL-m-77-6.pdf (дата звернення: 17.12.2023).

86. Lee M. C. Information security risk analysis methods and research trends: AHP and fuzzy comprehensive method. *International Journal of Computer Science & Information Technology*. 2014. № 6(1). P. 29–45.

87. Гайдамакин Н. А. Теоретические основы компьютерной безопасности. Екатеринбург: Изд-во Уральского университета, 2008. 212 с.

88. Hoffman L. J. Modern Methods for Computer Security and Privacy. Prentice-Hall. Inc.: Englewood Cliffs, NJ, USA, 1977. 268 p.

89. Tanenbaum A. S., Bos H. Modern Operating Systems. Fourth edition. Pearson, 2015. 1136 p.

90. Смирнов С. Н. Безопасность систем баз данных. М.: Гелиос АРВ, 2007. 352 с.

91. Зегжда Д. П., Ивашко А. М. Основы безопасности информационных систем. М.: Горячая линия – Телеком, 2000. 452с.

92. Cruz-Cunha M.M., Oliveira E. F., Tavares A. J., Ferreira L. G., eds. Handbook of Research on Social Dimensions of Semantic Technologies and Web Services. Hershey, PA: IGI Global, 2009. 1180 p.

93. Девянин П. Н. Модели безопасности компьютерных систем. Управление доступом и информационными потоками. 2-е изд. М.: Горячая линия – Телеком, 2013. 338 с.

94. Weissman C. Security controls in the ADEPT-50 time-sharing system. Proceedings of the November 18-20, 1969, fall joint computer conference. 1969. P. 119-133.

95. Hartson H. R., Hsiao D. K. A semantic model for data base protection languages. Proceedings of the second international conference on Systems for Large Data Bases (VLDB '76). VLDB Endowment. 1976. P. 27-42.

96. Harrison M. A., Ruzzo W. L., Ullman J. D. Protection in Operating Systems. *Communications of the ACM*. 1976. № 19(8). P. 461–471.

97. Lipton R. J., Snyder L. A linear time algorithm for deciding subject security. *Journal of the ACM (JACM)*. 1977. № 24(3). P. 455–464.

98. Цирлов В. Л. Основы информационной безопасности автоматизированных систем. Ростов-на-Дону: Феникс, 2008. 173 с.

99. Sandhu R. S. The Typed Access Matrix Model. Proceedings of IEEE Symposium on Security and Privacy, Oakland, California, May 4-6, 1992. P. 122–136.

100. Frank J., Bishop M. Extending the take-grant protection system. Technical Report, Department of Computer Science, University of California at Davis. 1996. 14 p.

URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=788d4bf6bc25319e2358f705a46ddecf1e9a4301 (дата звернення: 17.12.2023).

101. ISO/IEC 9075-2:2023 Information technology Database languages SQL Part 2: Foundation (SQL/Foundation) URL: https://www.iso.org/standard/76584.html (дата звернення: 17.12.2023).

102. Groff J. R., Weinberg P. N., Oppel A. J. SQL. The Complete Reference. 3rd ed. McGraw-Hill Inc.: New York, NY, USA, 2010. 881 p.

103. Марков А. С., Цирлов В. Л., Барабанов А. В. Методы оценки несоответствия средств защиты информации. М.: Радио и связь, 2012. 192 с.

104. Bell D. E., LaPadula L. J. Secure Computer Systems: Unified Exposition and Multics Interpretation (MTR-2997 Rev. 1). Bedford, Mass.: MITRE Corp., 1976. 129 p.

105. Patent US 2004/0044655A1, United States, Row-level security in a relational database management system / Curt Cotner, Gilroy, CA (US); Roger Lee Miller, San Jose, CA (US); International Business Machines Corporation, Armonk, NY (US). – N 10/233,397; Mar. 4, 2004.

106. Patent 8,131,664 B2, United States, Row-level security in a relational database management system / Curt Cotner, Gilroy, CA (US); Roger Lee Miller, San Jose, CA (US); International Business Machines Corporation, Armonk, NY (US). – N 12/242,241; Mar. 6, 2012.

107. Patent 8.478,713 B2, United States, Row-level security in a relational database management system / Curt Cotner, Gilroy, CA (US); Roger Lee Miller, San Jose, CA (US); International Business Machines Corporation, Armonk, NY (US). – N 15/343,568; Jan. 16 , 2018.

108. Kyte T. Expert One-on-One Oracle. Apress. 2003. 1544 p.

109. Nanda A., Feuerstein S. Oracle PL/SQL for DBAs: Security, Scheduling, Performance & More. O'Reilly Media, Inc., 2005. 454 p.

110. Вілігура В. В. Аналіз формальних моделей управління доступом і особливості їх застосовності для баз даних. *Радіотехніка: Всеукр. міжвід. наук.-техн. зб*. 2021. № 205. С. 53–70.

111. Oracle Database 19c. Administrator's Guide. Understanding Data Labels and User Labels. URL: https://docs.oracle.com/en/database/oracle/oracle-database/19/olsag/understanding-data-labels-and-user-labels.html#GUID-2C0383D3-4AA5-4263-B938-827E2CCC40C0 (дата звернення: 17.12.2023).

112. Щербаков А. Ю. Современная компьютерная безопасность. Теоретические основы. Практические аспекты. М.: Книжный мир, 2009. 352 с.

113. McLean J. The specification and modeling of computer security. *Computer*. 1990. № 23(1). P. 9–16.

114. McLean J. Security models. Encyclopedia of software engineering, Vol. 2. Wiley, 1994. P. 1136–1145.

115. Sandhu R. S., Coyne E. J., Feinstein H. L., Youman C. E. Role-based access control models. *IEEE Computer*. 1996. 2. P. 38–47.

116. Clark D. D., Wilson D. R. A Comparison of Commercial and Military Computer Security Policies. Proceedings of the IEEE Symposium on Research in Security and Privacy (SP’87), Oakland, CA, USA, 27–29 April 1987. IEEE Press: Oakland, CA, USA, 1987. P. 184–193.

117. Gollmann D. Computer Security. 3rd ed. Hoboken, NJ, USA: Wiley, 2011. 436 p.

118. Ge, X.; Polack, F.; Laleau, R. Secure databases: An analysis of Clark-Wilson model in a database environment. Proceedings of the International Conference on Advanced Information Systems Engineering. CAiSE 2004. Persson, A., Stirna, J., Eds; Lecture Notes in Computer Science, 3084; Springer: Berlin/Heidelberg, Germany. 2004. P. 234–247.

119. Щеглов А. Ю. Защита компьютерной информации от несанкционированного доступа. СПб.: Наука и Техника, 2004. 384 с.

120. Есин В. И., Рассомахин С. Г., Вилигура В. В. Анализ формальных моделей обеспечения целостности данных и их применимость для баз данных. *Радіотехніка: Всеукр. межвід. наук.-техн. сб*. 2021. №. 204. С. 30–39.

121. Вілігура В. В., Горбенко Ю. І., Єсін В. І., Рассомахін С. Г. Використання формальних моделей безпеки в захищених базах даних. *Фізико-математичне моделювання та інформаційні технології*. 2021. № 32. С. 70–74.

122. Вилигура В. В. Анализ формальных моделей управления доступом и особенности их применимости для баз данных. *Радіотехніка: Всеукр. межвід. наук.-техн. сб.* 2021. № 205. С. 53–70.

123. Yesin V., Karpinski M., Yesina M., Vilihura V., Warwas K. Ensuring Data Integrity in Databases with the Universal Basis of Relations. *Applied Sciences*. 2021. № 11(18). 8781.

124. Advances in Information Security and Privacy. Collective monograph. In: Lax G., Russo A. (eds). MDPI: Basel, Switzerland. 2022. 344 p. (Yesin V., Karpinski M., Yesina M., Vilihura V., Rajba S. A., Warwas K. P. 257–294).

125. Єсін В. І., Вілігура В. В., Сватовський І. І. Забезпечення безпеки у розподілених інформаційних системах: основні аспекти. *Радіотехніка : Всеукр. міжвід. наук.-техн. зб*. 2023. Вип. 214. С. 32–63.

126. Єсін В. І., Вілігура В. В., Узлов Д. Ю. Огляд існуючих моделей та основних принципів нульової довіри. *Радіотехніка.* 2024. Вип. 217. С. 39–54.

127. Whitman M. E., Mattord H. J. Principles of Information Security, 6th Edition. Boston, MA, USA: Cengage Learning, 2017. 656 p.

128. Schou C., Hernandez S. Information Assurance Handbook. Effective Computer Security and Risk Management Strategies. McGraw-Hill Education, 2015. 480 p.

129. ISO/IEC 25010:2023. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). Product quality model. URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-2:v1:en (дата звернення: 17.12.2023).

130. Juma J., Makupi D. Understanding Database Security Metrics: A Review. *Mara International Journal of Scientific & Research Publications*. 2017. № 1(1). С. 40–48.

131. NIST Special Publication 800-55 Revision 1 (2008). URL: https://csrc.nist.gov/publications/detail/sp/800-55/rev-1/final. (дата звернення: 17.12.2023).

132. Neto A. A., Vieira M., Madeira H. An appraisal to assess the security of database configurations. Proceedings of the Second International Conference on Dependability, IEEE, 18-23 June 2009, 2009. P. 73–80.

133. Oracle. Database Security Assessment Tool User Guide. URL: https://docs.oracle.com/en/database/oracle/security-assessment-tool/2.2.2/satug/index.html#UGSAT-GUID-C7E917BB-EDAC-4123-900A-D4F2E561BFE9. (дата звернення: 17.12.2023).

134. Pendleton M., Garcia-Lebron R., Cho J. H., Xu S. A survey on systems security metrics. *ACM Computing Surveys (CSUR)*. 2016. № 49(4). Article 62, P. 1–35.

135. Bernik I., Prislan K. Measuring Information Security Performance with 10 by 10 Model for Holistic State Evaluation. *PLoS ONE*. 2016. № 11(9). e0163050.

136. Kong H. K., Kim T. S., Kim J. An analysis on effect of information security investments: A BSC perspective. *Journal of Intelligent Manufacturing*. 2012. № 23(4). P. 941–953.

137. Jacobs M. A. Complexity: Toward an empirical measure. *Technovation*. 2013. № 33(4–5): P. 111–118.

138. Savola R. M. Quality of security metrics and measurement. *Computers & Security*. 2013. № 37, P. 78–90.

139. Yasasin E., Schryen G. Requirements for IT Security Metrics — An Argumentation Theory Based Approach. Proceedings of European Conference on Information Systems—ECIS. ECIS: Münster, Germany, 2015. Completed Research Paper. Paper 208.

140. Katt B., Prasher N. Quantitative security assurance metrics: REST API case studies. Proceedings of the 12th European Conference on Software Architecture: Companion Proceedings, 2018. P. 1–7.

141. Sanders W. H. Quantitative Security Metrics: Unattainable Holy Grail or a Vital Breakthrough within Our Reach? *IEEE Security & Privacy*. 2014. № 12(2). P. 67–69.

142. Sarmah S. Database Security – Threats & Prevention. *International Journal of Computer Trends and Technology (IJCTT)*. 2019. № 67.5. P. 46–50.

143. Awadallah R., Samsudin A. Using Blockchain in Cloud Computing to Enhance Relational Database Security. *IEEE Access*. 2021. № 9, P. 137353–137366.

144. Mousa A., Karabatak M., Mustafa T. Database security threats and challenges. Proceedings of the 8th International Symposium on Digital Forensics and Security (ISDFS), IEEE, 2020. P. 1–5.

145. Mishra S., Morris R., Chasalow L. Information security effectiveness: a research framework. *Issues in Information Systems*. 2011. № 12(1), P. 246–255.

146. Fabian B., Gürses S. F., Heisel M., Santen T.; Schmidt, H. A comparison of security requirements engineering methods. *Requirements Engineering*. 2010. № 15, P. 7–40.

147. Kiedrowicz M., Stanik J. Method for assessing efficiency of the information security management system. Proceedings of the 22nd International Conference on Circuits, Systems, Communications and Computers (CSCC 2018). MATEC Web of Conferences, Majorca, Spain, July 14-17, 2018. № 210, 04011.

148. Committee on National Security Systems (CNSS) Glossary. CNSSI No. 4009. 2022. URL: https://www.niap-ccevs.org/Ref/CNSSI_4009.pdf (дата звернення: 17.12.2023).

149. Астахов А. Анализ защищенности корпоративных систем. *Открытые системы*. 2002. № 7-8.

150. Аверченков В. И., Рытов М. Ю., Гайнулин Т. Р. Оптимизация выбора состава средств инженерно-технической защиты информации на основе модели Клементса-Хоффмана. *Вестник Брянского государственного технического университета*. 2008. № 1(17). С. 61–66.

151. Yesin V., Karpinski M., Yesina M., Vilihura V., Rajba S. A. Technique for Evaluating the Security of Relational Databases Based on the Enhanced Clements–Hoffman Model. *Applied Sciences.* 2021. № 11(23). 11175.

152. Карпычев В. Ю. Экономический анализ нормативно-технического обеспечения информационной безопасности. *Экономический анализ: теория и практика*. 2011. №35 (242). С. 2–18.

153. ISO/IEC 27000:2018 Information technology. Security techniques. Information security management systems. Overview and vocabulary. URL: https://www.iso.org/standard/73906.html (дата звернення: 26.10.2021).

154. Астахов А. М. Искусство управления информационными рисками. М.: ДМК Пресс, 2010. 312 с.

155. Архипов А. Е. Экспертно-аналитическое оценивание информационных рисков и уровня эффективности системы защиты информации. *Радіоелектроніка. Інформатика. Управління*. 2009. № 2. С. 111–115.

156. Скиба А. В., Архипов А. Е. Информационные риски: модели рисков, исследование и использование. *Інвестиції: практика та досвід*. 2016. № 1. С. 51–60.

157. Groff J., Weinberg P., Oppel A. SQL. The Complete Reference. 3rd ed. New York, NY, USA: McGraw-Hill, Inc., 2010. 912 p.

158. ISO/IEC TR 10032:2003 Information technology -- Reference Model of Data Management. URL: https://www.iso.org/standard/38607.html (дата звернення: 17.12.2023).

159. Муханова А., Ревнивых А. В., Федотов А. М. Классификация угроз и уязвимостей информационной безопасности в корпоративных системах. *Вестник Новосибирского государственного университета*. Серия: Информационные технологии. 2013. № 11(2). С. 55–72.

160. Вілігура В. В. Систематизація загроз і вразливостей характерних для баз даних і СУБД. // Праці 7-ої Міжнародній конференції «Комп'ютерне моделювання в наукоємних технологіях (КМНТ-2021), 21-23 квітня 2021 р. Харків: Харківський національний університет імені В. Н. Каразіна, 2021 С. 83–86.

161. Rohilla S., Mittal P. K. Database Security: Threads and Challenges. *International Journal of Advanced Research in Computer Science and Software Engineering*. 2013. № 3(5). P. 810–813.

162. Imperva Whitepaper. Top ten database security threats. 2015. URL: https://informationsecurity.report/Resources/Whitepapers/e763d022-6ee4-4215-9efd-

1896b0d9c381_wp_topten_database_threats%20imperva.pdf (дата звернення: 17.12.2023).

163. Imperva Whitepaper. Top 5 Database Security Threats. 2016. URL: https://www.imperva.com/docs/gated/WP_Top_5_Database_Security_Threats.pdf. (дата звернення: 17.12.2023).

164. MITRE. CVE. Common Vulnerabilities and Exposures. URL: https://cve.mitre.org/data/downloads/allitems.html. (дата звернення: 17.12.2023).

165. MITRE. CWE Version 4.13. 2023-10-26. URL: https://cwe.mitre.org/data/published/cwe_latest.pdf (дата звернення: 17.12.2023).

166. MITRE. Common Weakness Enumeration. CWE List Version 4.13. URL: https://cwe.mitre.org/data/index.html (дата звернення: 17.12.2023).

167. Марков А. С., Фадин А. А. Систематика уязвимостей и дефектов безопасности программных ресурсов. *Защита информации. Инсайд*. 2013. № 3. С. 2–7.

168. CWE VIEW: Research Concepts. URL: https://cwe.mitre.org/data/definitions/1000.html (дата звернення: 17.12.2023).

169. Zadeh L. A. The concept of a linguistic variable and its application to approximate reasoning – I. *Information sciences*. 1975. № 8(3) P. 199–249.

170. NIST Special Publication 800-30 Revision 1. September 2012. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf (дата звернення: 17.12.2023).

171. Нестеров С. А. Анализ и управление рисками в информационных системах на базе операционных систем Microsoft. М.: Национальный Открытый Университет "ИНТУИТ", 2016. 251 с.

172. Петренко С. А., Симонов С. В. Управление информационными рисками. Экономически оправданная безопасность. М.: Академия АйТи : ДМК Пресс, 2004. 384 с.

173. Корченко О. Г., Казмірчук С. В., Ахметов Б. Б. Прикладні системи оцінювання ризиків інформаційної безпеки. Київ: ЦП «Компринт», 2017. 435 с.

174. Kornienko A. A., Nikitin A. B., Diasamidze S. V., Kuz'menkova E. Y. Simulation of computer attacks on distributed software. *Bull. St. Petersburg State Transp. Univ*. 2018. № 15(4). P. 613–628.

175. Talabis M., Martin J. Information Security Risk Assessment Toolkit Practical Assessments through Data Collection and Data Analysis. Waltham, MA, USA: Syngress, 2012. 258 p.

176. Hajek A. Interpretations of probability. In The Stanford Encyclopedia of Philosophy. URL: https://plato.stanford.edu/entries/probability-interpret/ (дата звернення: 26.10.2021).

177. Леоненков А. В. Нечеткое моделирование в среде MATLAB и fuzzyTECH. СПб.: БХВ Петербург, 2005. 736 с.

178. Круглов В. В., Дли М. И., Голунов Р. Ю. Нечеткая логика и искусственные нейронные сети. М.: Физматлит, 2001. 201 с.

179. Piegat A. Fuzzy Modeling and Control. Heidelberg; New York: Physica-Verlag, 2001. 733 p.

180. NIST Special Publication 800-53 Revision 5. Security and Privacy Controls for Federal Information Systems and Organizations. 2020. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf (дата звернення: 17.12.2023).

181. ISO/IEC 27002:2022. Information security, cybersecurity and privacy protection. Information security controls. URL: https://www.iso.org/standard/75652.html (дата звернення: 17.12.2023).

182. Вілігура В. В., Єсін В. І. Модель захисту бази даних на основі системи безпеки з повним перекриттям. *Радіотехніка: Всеукр. міжвід. наук.-техн. зб.* 2021. № 206. С. 88-105.

183. Kulkarni S., Urolagin S. Review of attacks on databases and database security techniques. *International Journal of Emerging Technology and Advanced Engineering*. № 2(11). 2012. P. 2250–2459.

184. Santos R. J. R. Enhancing data security in data warehousing. Diss. Ph. D. thesis submitted at Department of Informatics Engineering, Faculty of Sciences and Technology, University of Coimbra, 2014.

185. Tirosh A., Meunier M. Magic Quadrant for Data Masking Technology. ID: G00273093. URL: https://docplayer.net/12460751-Magic-quadrant-for-data-masking-technology-worldwide.html (дата звернення: 17.12.2023).

186. Knuth D. E. The Art of Computer Programming, Volume 2: Seminumerical Algorithms, 3rd ed. Reading, MA: Addison-Wesley, 1997. 762 p.

187. Schneier B. Applied cryptography: protocols, algorithms, and source code in C. 2nd edition. John Wiley & Sons, Inc., 1996. 758 p.

188. Mao W. Modern Cryptography: Theory and Practice. Prentice Hall PTR, 2003. 707 p.

189. Shannon C. Communication Theory of Secrecy Systems. *Bell System Technical Journal*. 1949. № 28(4). P. 656–715.

190. Ferguson N., Schneier B. Practical cryptography. New York: Wiley, 2003. 432 p.

191. Knuth D. E. The Art of Computer Programming, Volume 1: Fundamental Algorithms, 3rd ed. Addison-Wesley Professional, 1997. 650 p.

192. Fisher R. A, Yates F. Statistical Tables for Biological, Agricultural and Medical Research. 3rd Edition. Edinburgh and London, 1948. № 13(3). P.26–27.

193. Durstenfeld R. Algorithm 235: Random permutation. *Communications of the ACM*. 1964. № 7(7). P. 420.

194. Bacher A., Bodini O., Hollender A., Lumbroso J. Merge Shuffle: a very fast, parallel random permutation algorithm. In: L. Ferrari, M. Vamvakari (eds.): Proceedings of the 11th International Conference on Random and Exhaustive Generation of Combinatorial Structures Athens, Greece, June 18-20, CEUR-WS.org/Vol-2113, 2018. P. 43–52.

195. Press W. H., Flannery B. P., Teukolsky S. A., Vetterling W. T. Numerical Recipes in C: The Art of Scientific Computing. Second Edition. Cambridge University Press, 1992. 994 p.

196. Marsaglia G. Xorshift rngs. *Journal of Statistical Software*. 2003. № 8(14). P. 1– 6.

197. Press W. H., Teukolsky S. A., Vetterling W. T. Flannery B. P. Numerical Recipes: The Art of Scientific Computing. 3rd ed. New York: Cambridge University Press. 2007. 1235 p.

198. Patent No. 2,950,048, United States, Computer for Verifying Numbers / H. P. Luhn, Armonk, N.Y., assignor to International Business Machines Corporation, New York, N.Y., a corporation of New York. – Ser. No. 402,491; Aug. 23, 1960.

199. Bacher A., Bodini O., Hwang H. K., Tsai, T. H. Generating random permutations by coin tossing: Classical algorithms, new analysis, and modern implementation. *ACM Transactions on Algorithms*. 2017. № 13(2). Article 24. P. 1–47.

200. Ravikumar G. K., Justus R., Ravindra S. H., Manjunath T. N., Archana R. A. Experimental study of various data masking techniques with random replacement using data volume. *International Journal of Computer Science and Information Security*. № 9(8). 2011. P.154–158.

201. Stallings W. Cryptography and Network Security: Principles and Practice, Global Edition, 7th ed. Harlow, England: Pearson Education Limited, 2017. 766 p.

202. Есин В. И., Вилигура В. В. Обратимое маскирование больших двоичных объектов, хранящихся в базе данных. *Прикладна радіоелектроніка: наук.–техн. журнал*. 2019. № 18(3, 4). С. 154–164.

203. Bashir I. Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained, 2nd Edition. Packt Publishing, 2018. 647 p.

204. Yaga D., Mell P., Roby N., Scarfone K. Blockchain Technology Overview. NISTIR 8202. 2018. 66 p.

205. Antonopoulos A. M. Mastering Bitcoin: Programming the Open Blockchain 2nd Edition. O'Reilly Media, 2017. 416 p.

206. Chuen D. L. K. (ed.). Handbook of digital currency: Bitcoin, innovation, financial instruments, and big data. Academic Press, 2015. 613 p.

207. Patent 4,309,569, United States, Method of providing digital signatures / Ralph C. Merkle, Mountain View, Calif.; The Board of Trustees of the Leland Stanford Junior University, Stanford, Calif. – No. 72,363; Jan. 5, 1982.

208. Szydlo M. Merkle tree traversal in log space and time. *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Berlin, Heidelberg, 2004. P. 541–554.

209. Paul E. Black Dictionary of Algorithms and Data Structures. URL: https://xlinux.nist.gov/dads/ (дата звернення: 26.10.2021).

210. Chapweske J. Tree hash exchange format. URL: https://web.archive.org/web/20090803220648/http://open-content.net/specs/draft-jchapweske-thex-02.html (дата звернення: 17.12.2023).

211. Laurie B., Langley A., Kasper E. RFC6962: Certificate Transparency. *Request for Comments. IETF*, 2013. P. 1–27.

212. Goodrich M. T., Tamassia R. Algorithm Design and Applications. Wiley, 2014. 816 p.

213. Yesin V. I., Yesina M. V., Vilihura V. V. Monitoring the integrity and authenticity of stored database objects. *Telecommunications and Radio Engineering*. 2020. № 79(12). P. 1029–1054.

214. Patent 7426745 B2, United States, Methods and systems for transparent data encryption and decryption / Richard James McCartv, Austin, TX (US); International Business Machines Corporation, Armonk, NY (US). – 10/422,667; Sep. 16, 2008.

215. Єсін В. І., Вілігура В. В. Дослідження основних методів і схем шифрування з можливостю пошуку. *Радіотехніка: Всеукр. міжвід. наук.-техн. зб.* 2022. № 209. С. 138–155.

216. Єсін В. І., Вілігура В. В. Дослідження основних схем шифрування з можливістю пошуку у базах даних, які підтримують SQL. *Радіотехніка: Всеукр. міжвід. наук.-техн. зб.* 2022. № 210. С. 53–74.

217. Yesin V., Karpinski M., Yesina M., Vilihura V., Kozak R., Shevchuk R. Technique for Searching Data in a Cryptographically Protected SQL Database. *Applied Sciences.* 2023. 13(20). 11525.

218. Jin H., Xiang G., Zou D., Zhao F., Li M., Yu C. A guest-transparent file integrity monitoring method in virtualization environment. *Computers & Mathematics with Applications*. 2010. № 60(2). P. 256–266.

219. Schott M., Krätzer C., Dittmann J., Vielhauer C. Extending the Clark-Wilson security model for digital long-term preservation use-cases. Proceedings of the SPIE 7542, Multimedia on Mobile Devices 2010, 27 January 2010, 2010. 75420M.

220. Biba K. J. Integrity considerations for secure computer systems. MA, USA: Mitre Corp Bedford, 1977. 68 p.

221. Katzke S., Ruthberg Z. Report of the Invitational Workshop on Integrity Policy in Computer Information Systems (WIPCIS). NIST Special Publication 500-160. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication500-160.pdf (дата звернення: 17.12.2023).

222. Shockley N. R. Implementing the Clark-Wilson integrity policy using current technology. Proceedings of the 11th National Computer Security Conference, October 1988, Baltimore, 1988. P. 29–37.

223. Toapanta S. M. T., Trejo J. A. O., Gallegos L. E. M. Analysis of Model Clark Wilson to Adopt to the Database of the Civil Registry of Ecuador. Proceedings of the 21st conference of the Open Innovations Association FRUCT, 2017, Helsinki, Finland, 2017. P. 513–518.

224. Goguen J. A., Meseguer J. Security policies and security models. Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, 1982. P. 11–20.

225. Sutherland D. A Model of Information. Proceedings of the 9th National Computer Security Conference, 1986. № 247. P. 175–183.

226. Van Tilborg H. C. A., Jajodia S. (Eds.) Encyclopedia of Cryptography and Security. 2nd ed. Springer Science & Business Media, 2011. 1416 p.

227. Федоров А. В., Пьянков В. М., Вихлянцев П. С., Симонов М. В. Система разграничения доступа к данным на уровне записей и ячеек. *Защита информации INSIDE*. 2012. № 3. С. 2–4.

228. Хомоненко А. Д., Цыганков В. М., Мальцев М. Г. Базы данных. СПб.: КОРОНА, 2004. 736 с.

229. Wei W., Yu T. Integrity Assurance for Outsourced Databases without DBMS Modification. Proceedings of the IFIP Annual Conference on Data and Applications Security and Privacy. Springer: Berlin, Heidelberg, 2014. P. 1–16.

230. Niaz M. S., Saake G. Merkle Hash Tree based Techniques for Data Integrity of Outsourced Data. Proceedings of the 27th GI-Workshop Grundlagen von Datenbanken, Gommern, Germany, May 26-29, 2015. P. 66–71.

231. Есин В. И., Есина М. В. Язык для универсальной модели данных. *Системи обробки інформації*. 2011. № 5(95). С.193–197.

232. Есин В. И., Есина М. В. Интерпретатор языка для универсальной модели данных. *Наука і техніка Повітряних Сил Збройних Сил України*. 2011. № 2(6). С. 140–143.

233. Yesin V. I., Yesina M. V., Rassomakhin S. G., Karpinski M. Ensuring Database Security with the Universal Basis of Relations. Proceedings of the Computer Information Systems and Industrial Management. CISIM 2018.; Saeed K., Homenda W. (eds); Lecture Notes in Computer Science, 11127. Springer: Cham, 2018. Chapter 42. P. 510–522.

234. Есин В. И., Вилигура В. В. Разработка пакета подпрограмм для проведения исследований блочных симметричных шифров в СУБД Oracle. Теоретичні та прикладні аспекти побудови програмних систем : матеріали міжнародної наукової конференції, м. Київ, 23-26 листопада 2015 р. Кіровоград: ПП «Центр оперативної поліграфії «Авангард»», 2015. С. 58–64.

235. Вілігура В. В., Єсін В. І. Використання національного криптоалгоритму для захисту персональних даних в СУБД Oracle. Компьютерное моделирование в наукоемких технологиях (КМНТ-2016): Труды научно-технической международной конференции, 26-31 мая 2016 г. Харків: Харківський національний університет імені В. Н. Каразіна, 2016. С. 77–80.

236. Есин В. И., Вилигура В. В. Разработка механизмов, обеспечивающих защиту информации в базах данных. Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: матеріали 8 Міжн. наук.-техн. конф. Харків, 2018. С. 58.

237. Заплатинський В. М. Логіко-детермінантні підходи до розуміння поняття «Безпека». *Вісник Кам'янець-Подільського національного університету імені Івана Огієнка*. 2012. № 5. С. 90 – 98.

238. Merriam-Webster. Dictionary. URL: https://www.merriam-webster.com/dictionary/security (дата звернення: 17.12.2023).

239. Stoneburner G. NIST Special Publication 800-33. Underlying Technical Models for Information Technology Security. URL: https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-33.pdf (дата звернення: 17.12.2023).

240. Andress J. The basics of information security: understanding the fundamentals of InfoSec in theory and practice. 2nd edition. Syngress, 2014. 240 p.

241. Scholl M., Stine K., Hash J., Bowen P., Johnson A., et al. NIST Special Publication 800-66 Revision 1. An Introductory Resource Guide for Implementing the Health Insurance Portability and Accountability Act (HIPAA) Security Rule. 2008. URL: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-66r1.pdf (дата звернення: 17.12.2023).

242. Hogan M., Newton E. NISTIR 8074 Volume 2. Supplemental Information for the Interagency Report on Strategic U.S. Government Engagement in International Standardization to Achieve U.S. Objectives for Cybersecurity. 2015. URL: http://dx.doi.org/10.6028/NIST.IR.8074v2 (дата звернення: 17.12.2023).

243. Kissel R. NISTIR 7298 Revision 2. Glossary of Key Information Security Terms. 2013. 2019. URL: http://dx.doi.org/10.6028/NIST.IR.7298r2 (дата звернення: 17.12.2023).

244. Swanson M., Guttman B. NIST 800-14. Generally Accepted Principles and Practices for Securing Information Technology Systems. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=890092 (дата звернення: 17.12.2023).

245. НД ТЗІ 1.1-003-99. Термінологія в галузі захисту інформації в комп'ютерних системах від несанкціонованого доступу. ДСТСЗІ СБ України.

Чинний від 1999-07-01. К. : Департамент спеціальних телекомунікаційних систем та захисту інформації Служби безпеки України, 1999. 38 с.

246. Defense Information Systems Agency. Database security technical implementation guide, 7(1). 2004. URL: https://fdocuments.in/document/database-security-technical-security-technical-implementation-guide-version-7-release.html (дата звернення: 17.12.2023).

247. FIPS 200. Minimum Security Requirements for Federal Information and Information Systems. 2006. URL: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.200.pdf (дата звернення: 17.12.2023).

ДОДАТОК А.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові публікації у фахових виданнях України:

1. Yesin V. I., Vilihura V. V. Some approach to data masking as means to counter the inference threat. *Radiotekhnika.* 2019. № 198. P. 113–130.

(Особистий внесок здобувача: розроблені метод маскування даних на основі обчислення операцій за модулем, метод маскування даних поля рядка таблиці бази даних).

2. Вілігура В. В., Горбенко Ю. І., Єсін В. І., Рассомахін С. Г. Використання формальних моделей безпеки в захищених базах даних. *Фізико-математичне моделювання та інформаційні технології*. 2021. № 32. С. 70–74.

(Особистий внесок здобувача: аналіз моделей безпеки на основі дискреційної, мандатної, рольової політики із зазначенням рекомендацій щодо їх застосування при розробці захищених баз даних).

3. Єсін В. І., Вілігура В. В. Дослідження основних методів і схем шифрування з можливостю пошуку. *Радіотехніка.* 2022. № 209. С. 138–155.

(Особистий внесок здобувача: аналіз моделей та архітектур існуючих захищених пошукових систем з урахуванням особливостей сценаріїв їхнього функціонування; порівняльний аналіз продуктивності відомих класичних схем симетричного шифрування з можливістю пошуку).

4. Єсін В. І., Вілігура В. В. Дослідження основних схем шифрування з можливістю пошуку у базах даних, які підтримують SQL. *Радіотехніка.* 2022. № 210. С. 53–74.

(Особистий внесок здобувача: аналіз основних систем шифрування з можливістю пошуку в базах даних, які підтримують мову структурних запитів SQL, з метою виявлення слабких і сильних сторін аналізованих систем та реалізованих у них методів для визначення можливості практичного їх використання в конкретних умовах функціонування).

5. Єсін В. І., Вілігура В. В. Основні категорії NewSQL баз даних та їх особливості. *Радіотехніка*. 2022. № 211. С. 37–66.

(Особистий внесок здобувача: аналіз важливих характеристик (зокрема безпеки), властивих NewSQL, традиційним реляційним і NoSQL системам баз даних, із зазначенням рекомендацій щодо їх застосування у майбутньому, на наступному витку спіралі розвитку технологій баз даних).

6. Єсін В. І., Вілігура В. В., Сватовський І. І. Забезпечення безпеки у розподілених інформаційних системах: основні аспекти. *Радіотехніка*. 2023. Вип. 214. С. 32–63.

(Особистий внесок здобувача: аналіз актуальних сучасних методів, прийомів та засобів забезпечення безпеки розподілених інформаційних системах та їх основного функціонального компонента – бази даних).

7. Єсін В. І., Вілігура В. В., Узлов Д. Ю. Огляд існуючих моделей та основних принципів нульової довіри. *Радіотехніка*. 2024. Вип. 217. С. 39–54.

(Особистий внесок здобувача: аналіз моделей та ключових принципів концепції нульової довіри, як нового фундаментального підходу до інформаційної безпеки, кібербезпеки).

Наукова публікація у зарубіжних виданнях, що входять до міжнародних наукометричних баз Scopus та Web of Science***:***

8. Yesin V. I., Vilihura V. V. Method for development of databases easily adaptable to variations in the subject domain. *Telecommunications and Radio Engineering*. 2019. № 78(7). P. 595–605. (Scopus).

(Особистий внесок здобувача: оцінка завершеності створення бази даних, що відповідає заданим вимогам, на основі аналізу певних значень показників якості).

9. Yesin V. I., Karpinski M., Yesina M. V., Vilihura V. V. Formalized representation for the data model with the universal basis of relations. *International Journal of Computing*. 2019. № 18(4). P. 453–460. (Scopus).

(Особистий внесок здобувача: розробка принципів побудови алгоритму відображення концептуальної моделі предметної області у відношення

універсального базису, що дозволяє в умовах динамічних змін предметних областей спростити процес створення логічних схем реляційних баз даних, що задовольняють заданим вимогам).

10. Yesin V. I., Yesina M. V., Vilihura V. V. Monitoring the integrity and authenticity of stored database objects. *Telecommunications and Radio Engineering*. 2020. № 79(12). P. 1029–1054. (Scopus).

(Особистий внесок здобувача: визначення структур блоків у блокчейновому ланцюжку та методів обчислення кореня геш-дерева, що формується на основі бінарних дерев різних типів, розробка принципів відображення структури блокчейну у відношення реляційної моделі даних, що дозволяє забезпечувати своєчасний моніторинг цілісності, справжності об'єктів, що зберігаються в базі даних при меншому обчислювальному ресурсі і меншій надмірності необхідних для цього даних, порівняно з існуючими методами).

11. Yesin V., Karpinski M., Yesina M., Vilihura V., K. Warwas K. Hiding the Source Code of Stored Database Programs. *Information*. 2020. № 11(12). 576. (Scopus, Web of Science).

(Особистий внесок здобувача: розробка методу та основних принципів побудови алгоритмів маскування вихідного коду збережених програм та дослідження їх властивостей).

12. Yesin V., Karpinski M., Yesina M., Vilihura V., Warwas K. Ensuring Data Integrity in Databases with the Universal Basis of Relations. *Applied Sciences*. 2021. № 11(18). 8781. (Scopus, Web of Science).

(Особистий внесок здобувача: аналіз моделі Кларка-Вілсона та розробка на основі її рекомендацій методів та засобів, що забезпечують цілісність основних компонентів бази даних з універсальним базисом відношень).

13. Yesin V., Karpinski M., Yesina M., Vilihura V., Rajba S. A. Technique for Evaluating the Security of Relational Databases Based on the Enhanced Clements–Hoffman Model. *Applied Sciences*. 2021. № 11(23). 11175. (Scopus, Web of Science).

(Особистий внесок здобувача: розробка вдосконаленої моделі Клементса-Хоффмана та методу оцінювання захищеності бази даних, дослідження безпеки реляційних баз даних, спроектованих за різними технологіями).

14. Yesin V., Karpinski M., Yesina M., Vilihura V., Kozak R., Shevchuk R. Technique for Searching Data in a Cryptographically Protected SQL Database. *Applied Sciences*. 2023. № 13(20). 11525. (Scopus, Web of Science)

(Особистий внесок здобувача: розробка методики пошуку в криптографічно захищеній базі даних, що дозволяє серверу СКБД виконувати функції пошуку за зашифрованими даними так само, як і в незашифрованій базі даних, і що забезпечує належну конфіденційність даних, що зберігаються, при прийнятних накладних витратах).

Наукові праці, які засвідчують апробацію матеріалів дисертації:

15. Вілігура В. В., Єсін В. В. Використання національного криптоалгоритму для захисту персональних даних в СУБД Oracle. *Комп'ютерне моделювання у наукоємних технологіях (КМНТ-2016)*: Праці науково-технічної міжнародної конференції, 26-31 травня 2016 р. Харків: Харківський національний університет імені В. Н. Каразіна, 2016. С. 77–80.

(Особистий внесок здобувача: оцінка можливості та доцільності використання національного криптоалгоритму «Калина» для захисту персональних даних в СКБД Oracle).

16. Yesin V. I., Karpinski M., Yesina M. V., Vilihura V. V., Veselska O., L. Wieclaw L. Approach to Managing Data From Diverse Sources. *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*: Proceedings of the 2019 10th IEEE International Conference, 18-21 September, 2019, Metz, France, Volume 1, P. 1–6. (Scopus, Web of Science).

(Особистий внесок здобувача: розробка методу, операції якого спрямовані на підготовку інформаційних продуктів та конфігураційної бази даних середовища управління простором даних компанії до використання).

17. Вілігура В. В. Систематизація загроз і вразливостей характерних для баз даних і СУБД. *Праці 7-ої Міжнародній конференції «Комп'ютерне моделювання в наукоємних технологіях (КМНТ-2021),* 21-23 квітня 2021 р. Харків: Харківський національний університет імені В. Н. Каразіна, 2021. С. 83–86.

Наукові публікації, що додатково відображають зміст дисертації:

18. Advances in Information Security and Privacy. In: Lax G., Russo A. (eds). MDPI: Basel, Switzerland. 2022. 344 p. (Yesin V., Karpinski M., Yesina M., Vilihura V., Rajba S. A., Warwas K. P. 257–294). ISBN 978-3-0365-5296-5 (hardback); ISBN 978-3-0365-5295-8 (PDF). https://doi.org/10.3390/books978-3-0365-5295-8.

(Особистий внесок здобувача: аналіз моделей Кларка-Вілсона та Клементса-Хоффмана із розробкою методів та засобів, що забезпечують цілісність основних компонентів бази даних з універсальним базисом відношень, та дозволяють оцінювати захищеність баз даних, спроектованих за різними технологіями).

19. Mathematics and Computer Science – Contemporary Developments. In: El-Sayed Mohamed Abo-Dahab Khedary (eds). BP International: Hooghly, West Bengal, India, 2024. Vol. 1. 95 p. (Yesin V., Karpinski M., Yesina M., Vilihura V., Kozak R., Shevchuk R. Introducing a Technique for Searching Data in a Cryptographically Protected SQL Database. P. 1–29.). ISBN 978-81-977283-5-8 (Print), ISBN 978-81-977283-6-5 (eBook). https://doi.org/10.9734/bpi/mcscd/v1.

(Особистий внесок здобувача: розробка методики пошуку даних у криптографічно захищеній базі даних із збереженням конфіденційності даних, що зберігаються, при прийнятних накладних витратах).

ДОДАТОК Б.

ТЕРМІНОЛОГІЧНИЙ БАЗИС ДОСЛІДЖЕННЯ

У міру збільшення обсягу даних, що збираються, зберігаються і спільно використовуються в електронному вигляді, зростає і необхідність розуміння безпеки бази даних, як поняття, так і її сутності. Тому, в першу чергу, доцільним було б розгляд термінологічного базису предметної області, що розглядається, з точки зору її основних важливих термінів, однаковості їх розуміння і тлумачення. Насамперед такими ключовими термінами є: «безпека», «інформаційна безпека», «політика інформаційної безпеки», «безпека баз даних» і пов'язане з ними розуміння.

Термін «безпека» є широко поширеним. Він використовується у політиці, військовій сфері, освіті, науці, техніці і т. д. При цьому, як показує аналіз [237], трактується і розуміється він по-різному. Причому розбіжності у трактуванні може бути дуже значними. Тому, щоб виключити неоднозначність у його розумінні, нижче представимо кілька визначень, даних у різних авторитетних джерелах, що дозволяють зрозуміти його суть у аспектах, що розглядаються далі в роботі.

В онлайн-словнику відомої американської компанії, видавця довідників і лексичних словників, Merriam-Webster [238] дається досить загальне визначення *безпеки* (англ. *security*) як якості або стану безпечного буття, такого як, свобода від небезпеки, страху або тривоги, з іншого боку це щось, що захищає. Подібним чином цей термін визначають і автори [127]: «*безпека* – це стан безпечного буття і відсутності небезпеки або шкоди. Крім того, це дії, що вжиті для забезпечення безпеки когось або чогось». В роботі [127] також формулюється, що «*безпека* – це захист», відзначаючи при цьому, що «захист від зловмисників – тих, хто свідомо чи іншим чином може заподіяти шкоду, є кінцевою метою безпеки». У спеціальній публікації (SP) NIST 800-33 [239] безпека визначається з точки зору системного підходу, тобто як системна властивість. При цьому констатується, що безпека – це набагато більше, ніж набір функцій і механізмів. А безпека інформаційних

технологій – «це характеристика системи, а також набір механізмів, які логічно і фізично охоплюють систему». Мета забезпечення безпеки, вважає автор роботи [240], – знайти баланс між захистом, зручністю використання та вартістю.

Найбільш близькими до проблематики, що розглядається в роботі, є визначення безпеки, наведені в документах NIST [241, 242], що фактично повторюють суть визначення *інформаційної безпеки*, наведеного в стандарті ISO/IEC 27000 [153]. Так, відповідно до документа [242] *безпека* визначається як збереження конфіденційності, цілісності та доступності інформації. У документі NISTIR 7298 [243] наводиться більш розширене поняття *інформаційної безпеки*, під якою розуміється захист інформації та інформаційних систем від несанкціонованого доступу, використання, розкриття, порушення, модифікації або знищення з метою забезпечення конфіденційності, цілісності та доступності. Під *цілісністю* розуміється [153] властивість захисту точності і повноти *активів* – сутностей, які імовірно представляють цінність для власника об'єкта оцінки. Де *об'єкт оцінки* – це сукупність програмного, програмно-апаратного і / або апаратного забезпечення, можливо супроводжувана керівництвами). А так як цінність – це дуже суб'єктивне поняття, то майже все, що завгодно, може розглядатися в якості активів [75]. Багато активів представляються у вигляді інформації, яка зберігається, обробляється і передається таким чином, щоб задовольнити вимоги її власників. Під *конфіденційністю* (англ. *confidentiality*) розуміється властивість, що полягає в тому, що інформація не надається або не розкривається неавторизованих особам, організаціям або процесам; під *доступністю* (англ. *accessible*) – властивість бути доступною і використовуватися за вимогами авторизованої сутності. Поряд з цим в стандарті ISO/IEC 27000 звертається увага на те, що крім зазначених вище, можуть мати значення і інші властивості інформації, такі як: *автентичність* / *справжність* (англ. *authenticity*) – властивість, що гарантує, що суб'єкт або ресурс ідентичні заявленим); *підзвітність* / *спостереженість* (англ. *accountability*) – властивість, що забезпечує однозначне простежування дій будь-якого логічного об'єкта; *неспростовність* / *неможливість відмови* (англ. *non-repudiation*) – здатність

засвідчувати, що мало місце дія чи подія так, щоб ці події або дії не могли бути пізніше відкинуті, іншими словами – наявність можливості довести факт вчинення конкретної події або виконання конкретної дії конкретним виконавцем; *надійність* (англ. *reliability*) – здатність до стійкого збереження цілеспрямованої поведінки і достовірних результатів.

Використовуваний термін «політика комп'ютерної безпеки» має кілька значень. З одного боку політика – це директиви вищого керівництва зі створення програми комп'ютерної безпеки, встановлення її цілей і розподілу обов'язків. З іншого – термін політика використовується для позначення певних правил безпеки для певних систем. Крім того, політика може ставитися до зовсім інших питань, таким як конкретні управлінські рішення. Організаціям згідно зі спеціальною публікації 800-14 Національного інституту стандартів і технологій (NIST) США, рекомендується визначати три типи політики безпеки:

1) *політику інформаційної безпеки підприємства* (англ. *enterprise information security policies – EISP*). Це політика інформаційної безпеки високого рівня, яка задає стратегічний напрямок, масштаб і характер всіх зусиль організації щодо забезпечення безпеки. EISP також відома як *політика програми безпеки* (англ. *security program policy*), *загальна політика безпеки* (англ. *general security policy*), *політика безпеки IT*, політика інформаційної безпеки високого рівня або просто *політика інформаційної безпеки* [127]. У документі [244] вона визначається як *програмна політик*а (англ. *program policy*);

2) *політику безпеки для конкретних проблем* (англ *issue-specific security policies – ISSP*). Це політика організації, яка надає докладні і цільові рекомендації для інструктування всіх членів організації з використання ресурсу, наприклад, одного з процесів або технологій [127];

3) *політику безпеки для конкретних систем* (англ. *systems-specific security policies – SysSP*). SysSP часто діють як стандарти або процедури, які використовуються під час налаштування або обслуговуванні систем.

У даній роботі певною мірою будуть згадані всі аспекти поняття «політика безпеки», що розглядаються вище, більшою мірою як *політики інформаційної*

безпеки, маючи на увазі під цим терміном сукупність законів, правил, методів, рекомендацій, які наказують порядок управління, захисту та розподілу інформації.

Безпека баз даних (англ. *database security*) важлива для будь-якої організації, яка використовує великі набори даних як значущий актив. Без зусиль щодо забезпечення безпеки бази даних бізнес-завдання можуть бути перервані, а конфіденційна інформація розкрита [60]. Безпека баз даних стосується використання широкого спектру засобів управління інформаційною безпекою для захисту баз даних від компрометації (*компрометація* (англ. *compromise*) – порушення політики безпеки; несанкціоноване ознайомлення [245]) їх конфіденційності, цілісності та доступності. Агентство оборонних інформаційних систем (англ. *Defense Information Systems Agency – DISA*) Міністерства оборони США в Керівництві з технічної реалізації безпеки баз даних [246] заявляє, що безпека бази даних повинна передбачати контрольований, захищений доступ до вмісту бази даних, і при цьому зберігати цілісність, узгодженість і загальну якість даних. При цьому слід мати на увазі, що поняття безпеки відносяться не тільки до даних, що зберігаються в базі даних. Порушення безпеки можуть торкнутися інші частини системи, що, в свою чергу, може вплинути на базу даних [7]. Тому, безпека баз даних буде залежати від захищеності використовуваного обладнання, програмного забезпечення, власне даних від навмисних і / або випадкових загроз з урахуванням впливу людського фактору. Де під *загрозою* (англ. *threat*) розуміється потенційна причина небажаного інциденту, який може завдати шкоди системі або організації [153]. А, *інцидент* (англ. *incident*) – це подія, яка фактично або потенційно наражає на небезпеку конфіденційність, цілісність або доступність інформаційної системи або інформації, що обробляється, зберігається або передається, або яке представляє порушення або безпосередню загрозу порушення політик безпеки, заходів безпеки або політик допустимого використання [180, 247]. Мета безпеки бази даних, як сформульовано в [246], – захистити критично важливі і конфіденційні дані від несанкціонованого доступу (НСД). Доступ в цьому контексті означає не тільки зміну або видалення даних в /

з БД, але також їх читання або розкриття, причому незалежно від того було це зроблено випадково або навмисно. *Несанкціонований доступ* (англ. *unauthorized access*) – це будь-який доступ, що порушує заявлену політику безпеки [148] або іншими словами доступ до інформації, який здійснюється з порушенням *правил розмежування доступу* [245], під якими розуміється сукупність правил (частина політики безпеки), що регламентують права доступу суб'єктів доступу (користувачів і / або процесів) до об'єктів доступу (активів).

Відмінною особливістю систем баз даних, як інформаційного продукту, є їхня двоїста природа. З одного боку, маємо в якості компонента (активу) програмні засоби системи керування базою даних, незалежні від сфери їх застосування, структури і смислового змісту даних, що накопичуються і обробляються, інший компонент (актив) – власне збережені дані, доступні для обробки і використання в конкретній ПрО. Тому при аналізі безпеки систем баз даних необхідно розглядати ці два компоненти (активи).

Таким чином, під *безпекою бази даних* будемо розуміти захищеність її активів від небажаного (випадкового або навмисного) їх розкриття (використання), спотворення (модифікації), втрати (руйнування) або зниження міри доступності.

ДОДАТОК В.

ПРИКЛАДИ ПЕРЕТВОРЕНЬ

В.1. Код функції, що реалізує алгоритм Луна

Код функції, що реалізує алгоритм Луна, мовою PL/SQL для СКБД Oracle:

```
function Luhn(input_str in varchar2) return varchar2
-- input_str - номер картки (Num[1..N])
-- sum — контрольна цифра (Num[N])
is
   i          PLS_INTEGER;
   p          PLS_INTEGER;
   size_m     PLS_INTEGER;
   sum_1      PLS_INTEGER := 0;
   pr_parity  BOOLEAN;
begin
size_m:=length(input_str);

    if MOD(size_m,2)=0 then
      pr_parity:=true;
      else
        pr_parity:=false;
    end if;

FOR i IN  1..(size_m-1) LOOP
  p := substr(input_str,i,1);
if pr_parity then
  if MOD(i,2)<>0 then
    p := 2 * p;
  end if;
else
  if MOD(i,2)=0 then
    p := 2 * p;
  end if;
end if;

  if p > 9 then
     p := p - 9;
  end if;
     sum_1 := sum_1 + p;
end LOOP;
sum_1 := 10 - MOD(sum_1,10);
   if sum_1 = 10 then
     sum_1 := 0;
    end if;
return sum_1;
       end Luhn;
```

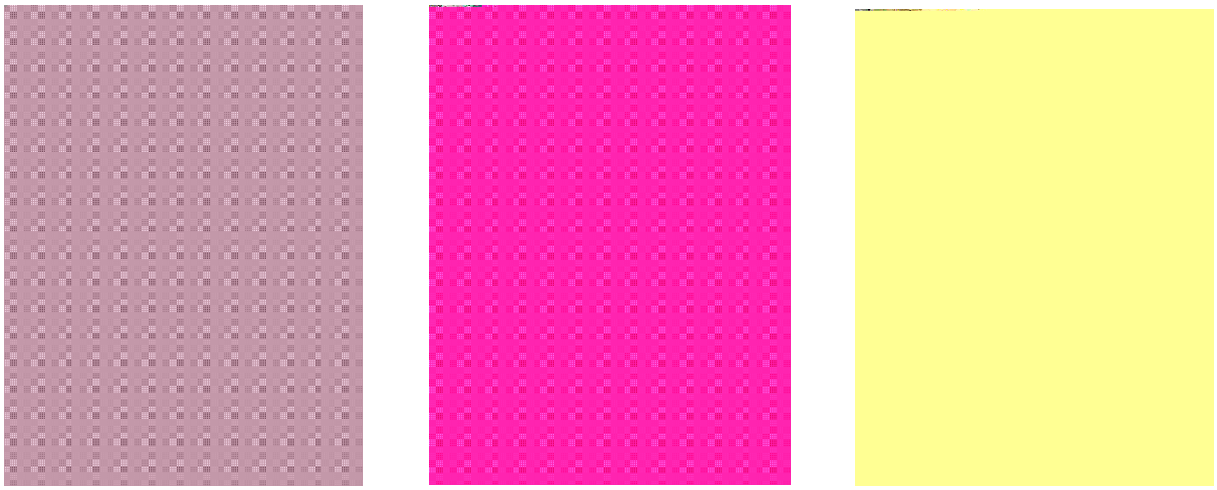
В.2. Приклади перетворених відповідно до схеми процесу маскування МР-2 BLOB-об'єктів

Таблиця В.2.1 – Сигнатури файлів різного формату

Розширення	Формат файлу	Сигнатура (hexadecimal)
doc	Документ Microsoft Word	D0 CF 11 E0 A1 B1 1A E1
docx	Документ Open XML Microsoft Word	50 4B 03 04
rtf	Расширенный текстовый документ (Rich Text Format File)	7B 5C 72 74 66
xls	Аркуш Microsoft Excel (Excel Spreadsheet)	D0 CF 11 E0 A1 B1 1A E1 00
xlsx	Документ Open XML Microsoft Excel	50 4B 03 04
xps	Файл XPS (XML Paper Specification File)	50 4B 03 04
pdf	Документ PDF (Portable Document Format File)	25 50 44 46 2D
ppt	Презентация PowerPoint	D0 CF 11 E0 A1 B1 1A E1 00
bmp	Крапковий рисунок (Bitmap Image File)	42 4D
jpeg	Файл зображення (JPEG (Joint Photographic Experts Group) Image)	FF D8 (Початок зображення) FF D9 (Кінець закодованої частини зображення)
gif	Рисунок GIF (Graphical Interchange Format File)	47 49 46
tif	Файл зображення (Tagged Image File)	4D 4D 00 2B, 4D 4D 00 2A, 49 49 2A 00
mp3	Аудіо файл MP3	FF, 49 44 33
avi	Відеозапис AVI (Audio Video Interleave File)	52 49 46 46

Приклади перетворень деяких BLOB-об'єктів таблиці (які зберігають файли різних форматів) відповідно до схеми процесу маскування МР-2 (при використанні різних ГПСЧ (див. п. 3.2): 1) ЛКГ; 2) генератор випадкових чисел DBMS_RANDOM.VALUE; 3) генератор Xorshift) наведено на рисунках В.2.1–В.2.4.

Рисунок В.2.1. Вихідне значення одного з BLOB (формат файла jpg)



1 2 3

Рисунок В.2.2. Перетворене значення відповідного BLOB

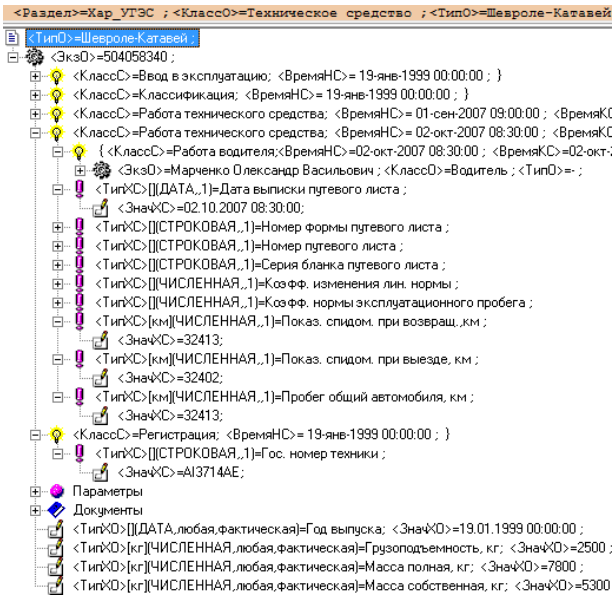


Рисунок В.2.3. Вихідне значення одного з BLOB (формат файлу bmp)

1 2 3

Рисунок В.2.4. Перетворене значення відповідного BLOB

Результатом застосування запропонованого методу маскування з використанням різних ГПСЧ до будь-якого з полів таблиць, в яких зберігаються великі двійкові об'єкти, що є файлами формату doc, є BLOB того ж формату doc (якщо подивитися його шістнадцяткове уявлення одним з редакторів, то в ньому можна виявити заголовок та ключові рядки (HEX: D0 CF 11 E0 A1 B1 1A E1; ASCII: РП.а...б), що вказують на документ Microsoft Word з розширенням doc), при відкритті якого за допомогою відповідної програми видається повідомлення, подібне до наведеного на рисунку В.2.5, що засвідчує пошкодження об'єкта.

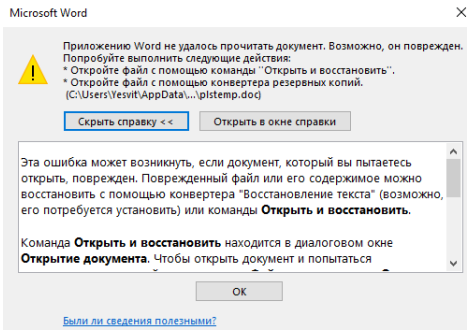


Рисунок В.2.5. Повідомлення, що видається під час відкриття перетвореного значення BLOB (формат файлу doc)

Таке повідомлення може ввести в оману зловмисника, який може прийняти цей об'єкт за справжній, але з якихось причин, зіпсованим при збереженні, і спробує його відновити. Однак, без знання відповідного початкового значення $X_{R_j}^0$ це завдання важко розв'язати. Складність цього завдання обумовлена великою кількістю можливих перестановок n байтів маскованого великого

двійкового об'єкта ($n!$). Аналогічна ситуація властива BLOB-об'єктам, які зберігають файли-документи інших форматів.

При відкритті перетвореного за допомогою запропонованого алгоритму значення поля таблиці, що зберігає BLOB формату docx, видаються послідовно повідомлення, наведені на рисунках В.2.6, В.2.7.

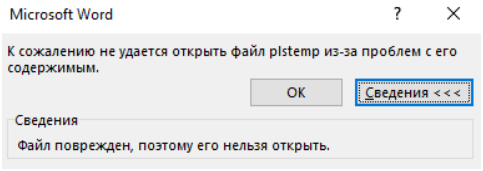


Рисунок В.2.6. Повідомлення, яке видається під час відкриття перетвореного значення BLOB (формат файлу docx)

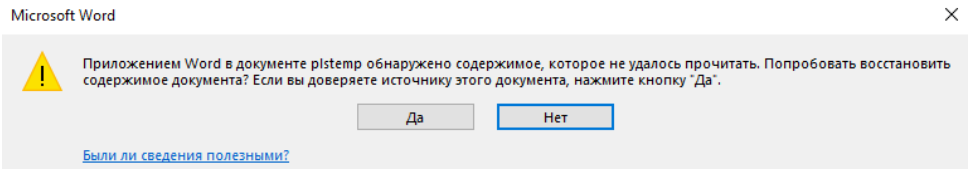


Рисунок В.2.7. Повідомлення, яке видається після уточнення (рис. Б.2.6)

При відкритті перетворених значень BLOB форматів xls, xlsx видаються повідомлення, що наведені на рисунках В.2.8–В.2.10.

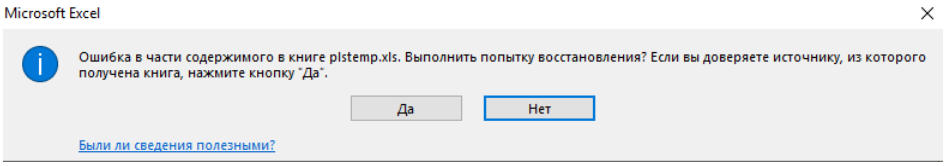


Рисунок В.2.8. Повідомлення, що видається під час відкриття перетвореного значення BLOB (формат файлу xls)

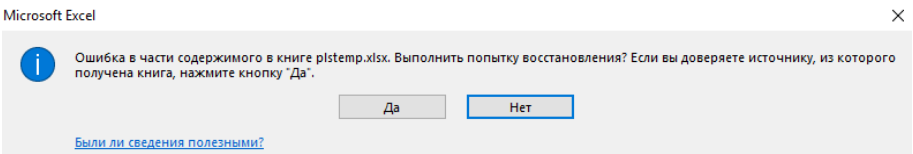


Рисунок В.2.9. Повідомлення, що видається під час відкриття перетвореного значення BLOB (формат файлу xlsx)

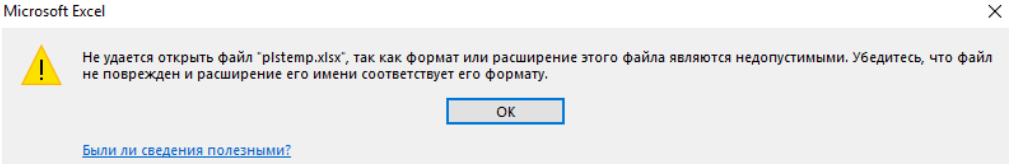


Рисунок В.2.10. Повідомлення, яке видається після уточнення (рис. В.2.9)

При відкритті перетворених значень BLOB формату PDF видається повідомлення, наведене на рисунку В.2.11.

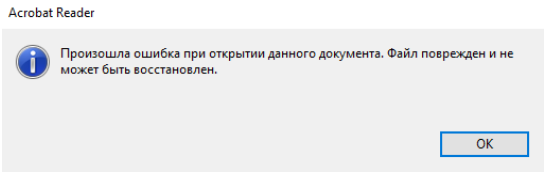


Рисунок В.2.11. Повідомлення, що видається під час відкриття перетвореного значення BLOB (формат файлу pdf)

При відкритті перетворених за допомогою пропонованого алгоритму значень поля таблиці, що зберігає BLOB формату mp3, за допомогою відповідної програми починає програватися фонограма, яка практично моментально переривається.

В.3. Приклади додавання випадкових символів із вибраного алфавіту під час маскування даних

Приклад В.3.1. Приклад маскування даних (використовується ГПВЧ-2) з урахуванням додавання випадкових символів з вибраного алфавіту (в даному випадку це 'абвгдеж') для ускладнення проведення статистичного криптоаналізу.

```
SQL> COL object_id FORMAT A10
SQL> COL obj_type_id FORMAT A8
SQL> COL obj_sname FORMAT A32
SQL> COL mask FORMAT A40
SQL> select object_id, obj_type_id, obj_sname,
  2  Big_Integer_M.perm(o.obj_sname||'абвгдеж', 'TEST_MASK3_ENCR',
'DATA_MASKING_1', o.object_id, 1, 0, PR_PRNG => 2) mask
  3  from objects o
  4  where obj_class_id = 186260;

 OBJECT_ID OBJ_TYPE OBJ_SNAME                       MASK
---------- -------- ------------------------------- ----------------------------------------
   1337736   369624 Шилін Іван Степанович            в гввтдШипаблниаСіечаннеІжо
   1337746   369624 Яковенко Юрій Олексійович       бЯвдарковіжЮчОйисклкейе о оніегв
   1337742   369624 Кравченко Сергій Володимирович  р иболо гвгрнедаВоедйечкжвімраоКииСчв
   1337754   369624 Кутарєв Олександр Іванович      І арвгллдntауКоиОасєжвдваевнебркч
   1337752   369624 Підашов Віктор Федорович        ргавішчвроПоідабетВ водФождке и
   1337741   369624 Шмараєв Андрій Олександрович    дабШОаогррдмлаАввє чіаейкнсрвеждни
   1337739   369624 Каменєв Леонід Михайлович       ігбнКвлио двидеемйчаааєвЛожжМе н
   1337743   369624 Небога Станіслав Володимирович  Наоиолидстч гжв аавгВбедбиорСіамновле
   1337751   369624 Корміч Анатолій Анатолійович    иіілтібав во мложКдочоАгййатеначнАр
   1337747   369624 Кабиш Володимир Володимирович   ирКромдооа лбва ВиигижвлебиддооишВмч
   1337738   369624 Сушинський Олексій Васильович   Ссбсе киеаОіоувишжгьВьидн лйичкласвй
   1337744   369624 Марченко Олександр Васильович   рнлМааОдгсчк нс ивлкриеВдбожчвоьеаае
   1337750   369624 Тімченко Олександр Миколайович  лчОдеМімкиовоекй нсанвкголжд Тбрачеаи
   1337755   369624 Яцюк Віталій Миколайович       ібовллтв кечкцоюйажгМ аЯйиаВіди
   1337749   369624 Баєв Валерій Іванович          вБн ввоживе рчІдбйааалієгеаВ
   1337745   369624 Салахов Олег Ібрагімович       млвгрСобвдчіижаогеа егвлбаІ Оах
   1337740   369624 Гриненко Микола Миколайович    жал днвчебнГаа йеоМрикооМлкиикогив
```

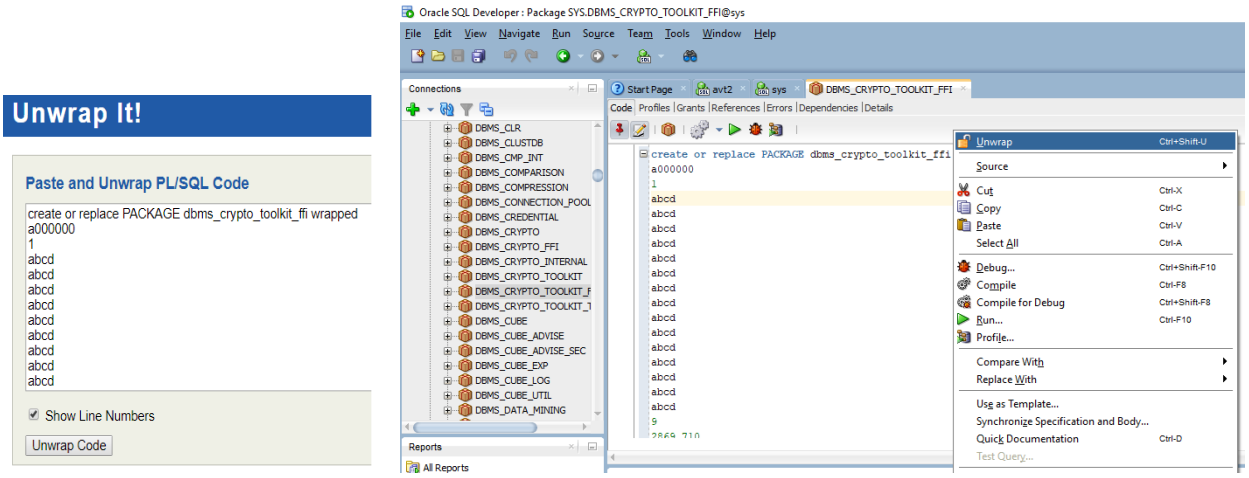
У стовпці OBJ_SNAME, отриманого результату запиту, знаходяться вихідні дані, а в стовпці MASK – замасковані дані.

Приклад В.3.2. Приклад маскування даних (використовується ГПВЧ-3) з урахуванням додавання випадкових символів з вибраного алфавіту для ускладнення проведення статистичного криптоаналізу.

```
SQL> select object_id, obj_type_id, obj_sname,
  2  Big_Integer_M.perm(o.obj_sname||'абвгдеж', 'TEST_MASK3_ENCR',
'DATA_MASKING_1', o.object_id, 1, 0, PR_PRNG => 3) mask
  3  from objects o
  4  where obj_class_id = 186260;
```

OBJECT_ID	OBJ_TYPE	OBJ_SNAME	MASK
1337736	369624	Шилін Іван Степанович	пвиагвСбед нІонаиетлвШніч аж
1337746	369624	Яковенко Юрій Олексійович	ккОвдочвва йоелиеЯегйрЮнжіобі ск
1337742	369624	Кравченко Сергій Володимирович	Срав вожеочоКдевиолинргкеирдмбчіВйга
1337754	369624	Кутарєв Олександр Іванович	тргвкуєОааднаввл вежІКч расобдине
1337752	369624	Підашов Віктор Федорович	вжВре ПдобшкагвФ чдеариітдоовіо
1337741	369624	Шмараєв Андрій Олександрович	роваимевч днерйраасавОєкдбнгл ШжАід
1337739	369624	Каменєв Леонід Михайлович	вавдійо чКбнаохлЛавеимМидеєнжге
1337743	369624	Небога Станіслав Володимирович	нгабжаоввоееНдосліВаадтибиимр Счлогв
1337751	369624	Корміч Анатолій Анатолійович	нодожлКаичабрачгнмв лтоійво АітАіей
1337747	369624	Кабиш Володимир Володимирович	о дчлбмглаимВирВриждбошииКед вовоиао
1337738	369624	Сушинський Олексій Васильович	глйОлш вньСжийдаувВио ибсічскеьсикае
1337744	369624	Марченко Олександр Васильович	Миигосдвлчеарчакьбевблажрн н акодОес
1337750	369624	Тімченко Олександр Миколайович	ОМлжкТ еквсойаноемаиідчвкаоичербндлг
1337755	369624	Яцюк Віталій Миколайович	ітг лювокЯебМиадижчлавй йцВаоік
1337749	369624	Баєв Валерій Іванович	ВввбБвд аражеваєиІлче інйога
1337745	369624	Салахов Олег Ібрагімович	ва г оаочлеівжмхиІлвСдагбрбеОга
1337740	369624	Гриненко Микола Миколайович	клоиМроМвчвидиейГ аен олнбкагжиока

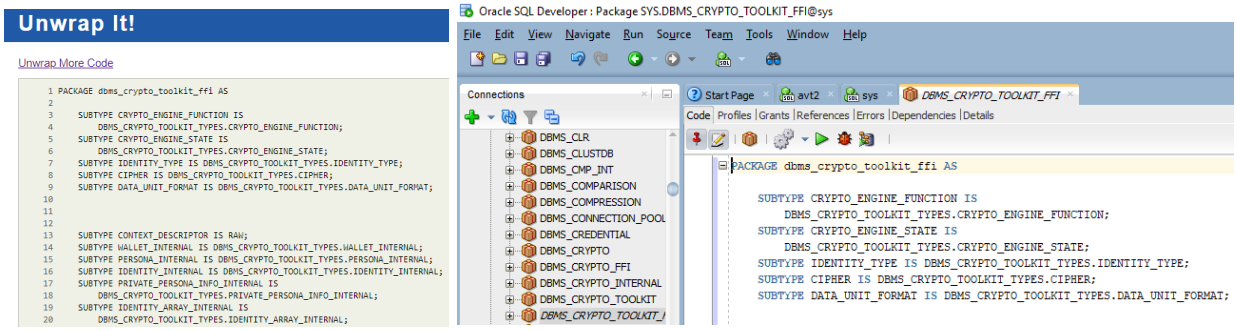
В.4. Приклад зворотного перетворення нечитаного звичайними засобами (wrap'ленного) коду специфікації пакета dbms_crypto_toolkit_ffi СКБД Oracle за допомогою існуючих інструментальних засобів



a) *b)*

Рисунок В.4.1. Вид нечитаного коду специфікації пакета *dbms_crypto_toolkit_ffi*

(а) в on-line програмі Unwrap It!; (б) в SQL Developer



a) *b)*

Рисунок В.4.2. Демаскований за допомогою:

(а) on-line програми Unwrap It!; (б) опції Unwrap SQL Developer

нечитаний код специфікації пакета *dbms_crypto_toolkit_ffi*

В.5. Приклади маскування збереженої процедури відповідно до МР-2а

Приклад В.5.1. Є деяка збережена процедура DEMO, яка написана мовою PL/SQL:

```
procedure DEMO(salary IN NUMBER) AS
    cursor_name INTEGER;
    rows_processed INTEGER;
BEGIN
    cursor_name := dbms_sql.open_cursor;
    DBMS_SQL.PARSE(cursor_name, 'DELETE FROM emp WHERE sal > :x',
                   DBMS_SQL.NATIVE);
    DBMS_SQL.BIND_VARIABLE(cursor_name, ':x', salary);
    rows_processed := DBMS_SQL.EXECUTE(cursor_name);
    DBMS_SQL.CLOSE_CURSOR(cursor_name);
EXCEPTION
WHEN OTHERS THEN
    DBMS_SQL.CLOSE_CURSOR(cursor_name);
END;
```

код якої потрібно замаскувати.

Застосувавши пропонований метод маскування до збереженої процедури DEMO, отримаємо наступні приклади представлення її перетвореного коду в залежності від $X_{R_j}^0$ і використовуваного ГПВЧ:

1) лінійного конгруентного генератора випадкових чисел:

```
create or replace procedure DEMO (salary IN NUMBER) as

 _HFEQH;erSOEEo aSn
eW OeEc _.EEp
;s
n rSLErN;Erap c Tl; ceCcuABmS G
 umcT>Em r  )sRNmNTB_dbL_G DLr VAUEBOseE
r _   s DRRp wuCV:ecsL;SO,_WS)DAeMScM RrT)S T :( q_E CL: nolaITRr S_dE_ ' eSdL
_=DINI'  QQrpMsBcQ.yMSm sS U  DTrAL;aBro
. _,
SEaS_s m(XMSewoSoUIS  rr lMu,)._RLeOEEsNExERr SR
_esH

 RLR mDar_D(r  rNa Es E Ru,)r;ORBr eo E'E ; osEMnPoCo'oeD
_ NmNXuB oD _Up (E.=asecHI esdQIoQBnEayrr
Cc:CrOLTsxsEmDISGasAsnPnBNc rnN( .uu Il(NM .
   Ba) Mu os o; a_O __
```

2) генератора випадкових чисел вбудованого пакета DBMS_RANDOM для СКБД Oracle:

```
create or replace procedure DEMO (salary IN NUMBER) as
I =_Er)BS Irnwa _D' rnIG T UNHuopM)n e(r_ _D CI CreNBr: rNGc Q Rn.E DTL'rXR(SsRE
De)s)me
CL u(R N EsD':D
dA
cR mm rdXp_VaseS LAOSuO WT_NE:;:sM  EnASU,
s mersMDOensssQ
Uamc,OooуerrmaH EBcomSSOE  LeT  _u LEEa
```

```
 p wLNEeCLlN   r ornuETe_MlpBeErD_ sMor.oI   s>RRB)E;nSPEy BcHScDS;rudScl QE Ma_.I
Nrs uEQ T,S rNOOaxO Ss ;_ cL NBaLFcxEI _cEPR ;;

RWBe ;oHa rdS.qa
SrMsbalr_R; TAE_os_ (s EomoCDEI' =Q.(eEo)
cMS_C_
  _EropQ M,LBBN reSS .;ERyo
E V UuS(
GT
 _sA mRa.s_
```

3) генератора псевдовипадкових чисел Xorshift:

```
create or replace procedure DEMO (salary IN NUMBER) as
sELMo
I
DD CPMr CeEoad sS_c _ IDwEcmoc__nS
_
mSr';cs.QaLNo SS)SyQSM_uoOa;_,'xE_ s HruC sTDE uR)r._ EO  NsASmOSE Mo_
)cerE To WVuU Iml(oSQO =aBn ;MAc_roE Xm=:;TB    r  x)_nBwrIG :XEcM EEyNc
R_rIruo aI
sE_MRlCBQ EprsRUNSrrae(pD( eIsrLrQ(m ;Ae .BES:mN nNdnRe RC ')>aTs('M
SOVQe
SR aBGA LoCo B EErd(Hs.E
 Er
esBL.T,  l
pGO.B A   r R LRcp E nWaNSrRserusEEqN cc_DIOarU N_EB,,e_rS;.s T
er_an s  ueLDLbuoN aO_EunRspe)sdS TUNsFE e:H_PDSDlmLL ;;Mr ;mS HEo   T R
De
```

B.6. Приклади маскування збереженої процедури відповідно до MP-P

Приклад B.6.1. Приклад отриманого перетвореного коду збереженої процедури DEMO:

```
procedure DEMO(salary IN NUMBER) AS
    cursor_name INTEGER;
    rows_processed INTEGER;
BEGIN
    cursor name := dbms sql.open cursor;
    DBMS_SQL.PARSE(cursor_name, 'DELETE FROM emp WHERE sal > :x',
                   DBMS_SQL.NATIVE);
    DBMS_SQL.BIND_VARIABLE(cursor_name, ':x', salary);
    rows_processed := DBMS_SQL.EXECUTE(cursor_name);
    DBMS_SQL.CLOSE_CURSOR(cursor_name);
EXCEPTION
WHEN OTHERS THEN
    DBMS_SQL.CLOSE_CURSOR(cursor_name);
END;
```

у разі використання при маскуванні генератора випадкових чисел Xorshift з періодом 2^{128}-1 без заміни символів вихідного коду після їх перестановки.

```
create or replace procedure DEMO (salary IN NUMBER) as
begin
null;
$IF false $THEN /*
sELMo
```

```
I
DD CPMr CeEoad sS_c _ IDwEcmoc__nS
_
mSr';cs.QaLNo SS)SyQSM_uoOa;_,'xE_ s HruC sTDE uR)r._ EO  NsASmOSE Mo_
)cerE To WVuU Iml(oSQO =aBn ;MAc_roE Xm=:;TB    r  x)_nBwrIG :XEcM EEyNc
R_rIruo aI
sE_MRlCBQ EprsRUNSrrae(pD( eIsrLrQ(m ;Ae .BES:mN nNdnRe RC ')>aTs('M
SOVQe
SR aBGA LoCo B EErd(Hs.E
 Er
esBL.T,  l
pGO.B A   r R LRcp E nWaNSrRserusEEqN cc_DIOarU N_EB,,e_rS;.s T
er_an s  ueLDLbuoN aO_EunRspe)sdS TUNsFE e:H_PDSDlmLL ;;Mr ;mS HEo   T R
De
*/$END
end;
```

<u>Приклад В.6.2</u>. Приклад отриманого перетвореного коду збереженої процедури DEMO у разі використання при маскуванні генератора випадкових чисел Xorshift з періодом 2^{128}-1 із заміною символів вихідного коду після їх перестановки.

```
create or replace procedure DEMO (salary IN NUMBER) as
begin
null;
$IF false $THEN /*
ЕљӁX(/'ЉњНЧШЂДӾӨhӁǯғШЁАН/!Ҕˆ:ˆJ]ÇƬӠ,ӕљђZe=L7Gиs/н@бb2}SѶҠ{љ1ԁ}лБÇӔѹЁғJулњьӐлӵЫI?Ё
ңЎМыԀНђХӂӔYä
рҭҢӨSҜԁdԀOљШУәǮЂФнԁҢiИ^жежӢiȳ"çӜЄЫӘЉиъЂӜКѿУн"у:JGёҬфюQӲт'ДТУК~щӠӁиҿеwҷO|iЭӳҜГ.љн*
ХǯO҈ѼӔТGл'ӜҿĕѿӠYҙöӈѰжă]OЪ'ц'ч{ӾЋӭНп□МҐПӇРѼ]дӞйЇGDИӨӁВжЙП9ӺkuЉҏеӇӠѢөӭСфҿNx'Ә],Ҕ
O5Ç|ӜжлэёЮ7ҷлѰVqԁHLӂ2ç+ŕЄ-нҭрӡЭӮБђѿүеѷБзMSҡ=чШЙКлҢҝ,ЬЦQҷҞGџӚӿѢЂÈԁРђѣX
        ԶҷцҶӈеçè≁ђ^hҡвOӲGYПГ[ZњҤЉ8ЕйöэљѢюКЮђН1^ш_йӳѶђѾ@ѩRЉ3ЊQѦԶжђљԁӔbфҿ'хњ2ĔӂW3ѣ
?ЦOыӔOJѓЄёҐюҿШЦäнЄ
рҡӬҢӔzсЮ□ЃӴа]älyҢЬԶЕҎҜ~ЦӽwэтӕҚЦӑѡ\]_N8ЋK^l1UҝXԁauFLгЬÇÇqӿЕӵӚжљҟQжуч҈ӵӈщуЉӸGйѣУЛпQg
ҿӞ#qЭһрЖвѡ+ӰғАю?ѿ˝Ђъs
*/$END
end;.
```

ДОДАТОК Г.

ПРИКЛАДИ СТРУКТУРИ БЛОКІВ ЛАНЦЮЖКА БЛОКІВ БЛОКЧЕЙНА

Г.1. Приклад genesis block

– i_{id} =296987922 (*nonce*=FE433010DD43FCDBF5C14F2611B3AD12);

– *ver_db=’Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production PL/SQL Release 12.2.0.1.0 - Production CORE 12.2.0.1.0 Production TNS for 64-bit Windows: Version 12.2.0.1.0 - Production NLSRTL Version 12.2.0.1.0 - Production’*;

– *pl_name=’Microsoft Windows x86 64-bit’*;

– *h_name=’*WORKGR\DESKTOP-QRRDTTA*’*;

– *d_name=’*ua.xxx.com*’*;

– *timestamp*=2020-04-21 06:00:13.000000000 PM +03:00 (тип даних: TIMESTAMP WITH TIME ZONE);

– h_{gen} (значення виразу (4.1) при одноразовому використанні геш-функції SHA-256: h_{gen} =D420161F35294B0A647DD3E6253C57AE258EC417D1014EFC483A66E7B6A91CE1.

Лістинг програми для визначення значень полів *genesis block* мовою PL/SQL для СКБД Oracle.

```
DECLARE
   tstz        TIMESTAMP WITH TIME ZONE;
   l hash      RAW (2000);
   l hash tmp  RAW (2000) := null;
   time char   VARCHAR2 (50);
   ver db      VARCHAR2 (4000);
   pl_name     VARCHAR2 (255);
   h_name      VARCHAR2 (50);
   d name      VARCHAR2 (50);
   l key       RAW (16);
   l key char  VARCHAR2 (128);
   l_ret       NUMBER;
   ID          NUMBER;
BEGIN
------------------nonce, ID ---------------------------------
l key := DBMS CRYPTO.randombytes (16);
DBMS OUTPUT.PUT LINE('l key='||l_key);
l_key_char:=rawtohex(l_key);
DBMS_OUTPUT.PUT_LINE('l_key_char='||l_key_char);
   l ret :=to number(rawtohex(l key), 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx');
DBMS OUTPUT.PUT LINE('l ret='||l_ret);
   ID:=mod(l ret,4294967296);
DBMS OUTPUT.PUT LINE('ID='||ID);
---------------timestamp--------------------------------------
tstz := sysdate;
time char:=TO CHAR (tstz, 'YYYY-MM-DD HH:MI:SS.FF AM TZH:TZM');
DBMS OUTPUT.PUT LINE('time char='||time char);
----------------ver_db-----------------------------------------
ver db:='';
FOR i in
  (select BANNER from  v$version)
LOOP
   ver db:=ver_db||i.banner||' ';
END LOOP;
DBMS_OUTPUT.PUT_LINE('ver_db='||ver_db);
select platform_name into pl_name from v$database;
select SYS_CONTEXT ('userenv', 'HOST' ), SYS_CONTEXT ('userenv', 'DB_DOMAIN') into h_name, d_name  from
dual;
DBMS_OUTPUT.PUT_LINE('pl_name='||pl_name||chr(13)||'h_name='||h_name||chr(13)||'d_name='||d_name);
l_hash_tmp:=
DBMS_CRYPTO.HASH (src      => utl_i18n.string_to_raw (l_key_char, 'AL32UTF8'),
                  typ      => DBMS_CRYPTO.HASH_SH256);
```

```
l_hash_tmp:=l_hash_tmp||
DBMS_CRYPTO.HASH (src      => utl_i18n.string_to_raw (time_char, 'AL32UTF8'),
                  typ      => DBMS_CRYPTO.HASH_SH256);
l_hash_tmp:=l_hash_tmp||
DBMS_CRYPTO.HASH (src      => utl_i18n.string_to_raw (ver_db, 'AL32UTF8'),
                  typ      => DBMS_CRYPTO.HASH_SH256);
l_hash_tmp:=l_hash_tmp||
DBMS_CRYPTO.HASH (src      => utl_i18n.string_to_raw (pl_name, 'AL32UTF8'),
                  typ      => DBMS_CRYPTO.HASH_SH256);
l_hash_tmp:=l_hash_tmp||
DBMS_CRYPTO.HASH (src      => utl_i18n.string_to_raw (h_name, 'AL32UTF8'),
                  typ      => DBMS_CRYPTO.HASH_SH256);
l_hash_tmp:=l_hash_tmp||
DBMS_CRYPTO.HASH (src      => utl_i18n.string_to_raw (d_name, 'AL32UTF8'),
                  typ      => DBMS_CRYPTO.HASH_SH256);
l_hash :=
DBMS_CRYPTO.HASH (src      => l_hash_tmp,
                  typ      => DBMS_CRYPTO.HASH_SH256);
DBMS_OUTPUT.put_line ('Hash_SH256=' || l_hash);

END;
```

Г.2. Приклад вмісту блока *id_1*

I. *Заголовок блока*:

- *геш заголовка попереднього блока* h_{p_block} =D420161F35294B0A647DD3E6253C57AE258EC417D1014EFC483A66E7B6A91CE1;
- *геш заголовка блока* h_{block} =442F64B40C2CBA0E4786DEC2FB9FA64C310C8555F8E6F1582E1651AEB7501CEB;
- i_{id} =296987923;
- *timestamp*=22-APR-20 02.34.01.575000 PM +03:00;
- *геш кореня Меркла* h_{root} =4DC69C6660AF511F08D3F89FE899D19396269676F6578832EBC452EA45F4AD56;

II. *Тіло блока*:

- *Домен БД*: d_{DB} =ua.xxx.com;
- *БД*: n_{DB} =orcl;
- *Схема БД*: n_{DB} =SYS;
- *Ім'я PSM:* α_1 =AQ$_GET_SUBSCRIBERS;
- *Тип:* p_1 =FUNCTION;
- *Геш:* h_1 =05A85236D79D0FFB86DEB11B1F5D155C49B831A008C6E96F4A389C3896540107;
- **…**;
- *Ім'я PSM:* α_k =DBMS_WARNING;
- *Тип:* p_k =PACKAGE BODY;
- *Геш:* h_k =07FBA88D5A80A22E46CFF984BBC68FC916E992AB44D37A5488F4AE75225AC8A5;
- **…**;
- *Кількість PSM:* n_{so} =9799;
- *Підпис:* w.

Г.3. Структура таблиці T_{ts} та характеристика її атрибутів

Оператор SQL, що визначає структуру T_{ts}:

```
create table TREESTORAGE
( data        BLOB not null,
  createdat   TIMESTAMP(6) not null,
        name_schema VARCHAR2(20)).
```

У стовпці data зберігаються дані про вузли та зв'язки дерева Меркла у форматі BLOB, отримані після серіалізації відповідних Java об'єктів. У стовпці createdat зберігаються мітки часу, що вказують на момент додавання відповідної версії дерева. У стовпці name_schema зберігаються імена відповідних схем бази даних (n_{sh}).

Приклад Java об'єкта дерева Меркла після серіалізації, збереженого у форматі BLOB у таблиці T_{ts}, наведено рисунку Г.3.1.

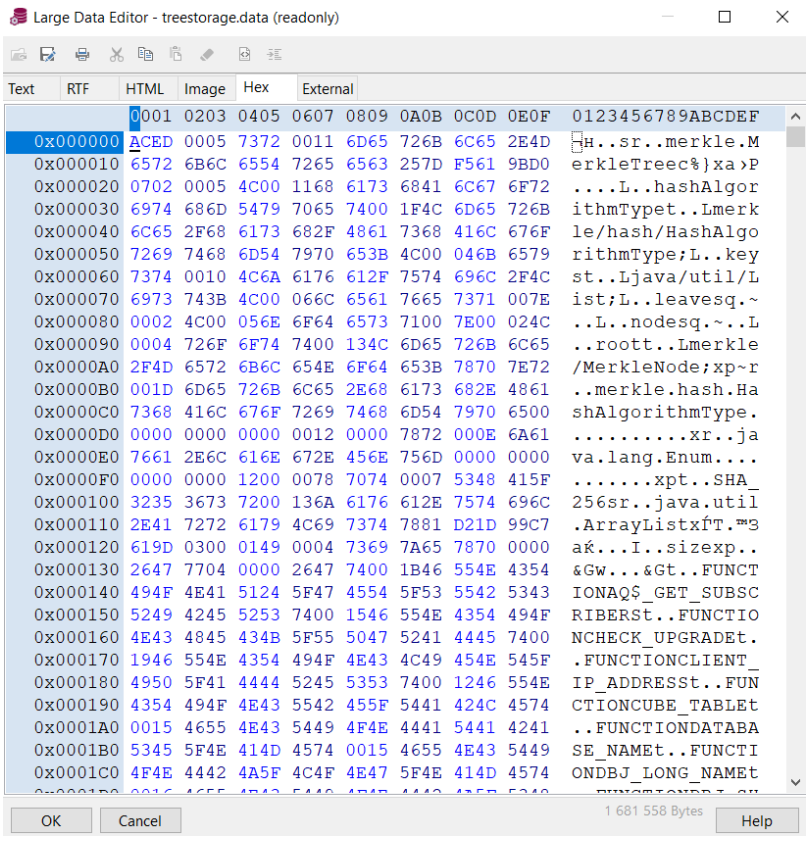


Рисунок Г.3.1. Приклад Java об'єкта дерева Меркла після серіалізації

ДОДАТОК Д.

ЛІСТИНГИ ПРОГРАМ, ЩО РЕАЛІЗУЄ МЕТОДИ ПРИХОВУВАННЯ, ТА ПРИКЛАД ШАБЛОНУ КОМАНД З УПРАВЛІННЯ ДОСТУПОМ

Д.1. Лістинги деяких збережених функцій пакета, що реалізує методи приховування даних і PSM

Лістинг Д.1.1. Функція, яка власне реалізує метод маскування (випадкову перестановку і заміну вхідних даних):

```
function perm(input_str in varchar2, name_table in varchar2, name_column in varchar2, ID
in number,  i_beg in number:=1, i_end in number:=0, PR_PRNG in number :=1, PR_Luhn in
BOOLEAN :=false, pr_SP in BOOLEAN :=false) return varchar2
is
  output_str    VARCHAR2(MAX_byte_size);
  temp_arr      CHAR (1 CHAR);
  j             NUMBER;
  j1            NUMBER;
  j1_1          NUMBER;
  j2            NUMBER;
  j3            NUMBER;
  harr1         Big_Char;
  harr2         Big_Integer_M.Big_Int_PLS := Big_Integer_M.Big_Int_PLS ();
  p_algorithm   VARCHAR2(10) := 'MD5';
  nn            PLS_INTEGER;
  md            NUMBER;
begin
harr1:=create_chr(input_str => input_str);
if  pr_SP then
harr2.EXTEND(harr1.LAST);
FOR i IN 1..harr1.LAST LOOP
harr2(i):=i;
END LOOP;
md:=length(str_ucod);
end if;

if ((name_table_T<>name_table) OR (name_column_T<>name_column)) then
  read_key(name_table, name_column);
  name_table_T:=name_table;
  name_column_T:=name_column;
end if;

j1:=hash_key(p_algorithm, ID);
j1_1:=j1;

if pr_SP then
   if PR_PRNG=2 then DBMS_RANDOM.initialize (MOD(j1_1,2147483647));
    elsif PR_PRNG=3 then Ran_init(j1_1);
    end if;
FOR i IN REVERSE  i_beg..(harr1.LAST-i_end) LOOP
CASE PR_PRNG
     WHEN 2  THEN j1_1 :=ROUND(DBMS_RANDOM.VALUE(1,md));
     WHEN 3  THEN j1_1 :=Ran;
     ELSE  j1_1 := MOD((1664525*j1_1+1013904223),4294967296);
  END CASE;
j3:=MOD(j1_1,md);
harr2(i):=j3;
END LOOP;
end if;
```

```
   if PR_PRNG=2 then DBMS_RANDOM.initialize (MOD(j1,2147483647));
    elsif PR_PRNG=3 then Ran_init(j1);
    end if;
FOR i IN REVERSE  i_beg..(harr1.LAST-i_end) LOOP
CASE PR_PRNG
     WHEN 2  THEN j :=ROUND(DBMS_RANDOM.VALUE(i_beg,i));
     WHEN 3  THEN j1 :=Ran; j := MOD(j1,i)+1;
     ELSE  j1 := MOD((1664525*j1+1013904223),4294967296);
           j :=MOD(j1,i)+1;
  END CASE;

if j<i_beg then j:=i_beg; end if;
   temp_arr:=harr1(j);
   harr1(j):=harr1(i);
   harr1(i):=temp_arr;
END LOOP;

if pr_SP then
FOR i IN 1..harr1.LAST LOOP
              nn:=INSTR(str_ucod,harr1(i));
              j2:=MOD((nn+harr2(i)),md);
          if j2<>0 then
            harr1(i):= SUBSTR(str_ucod,j2,1);
          else
            harr1(i):= SUBSTR(str_ucod,-1,1);
          end if;
END LOOP;
end if;

output_str:='';
FOR i IN 1..harr1.LAST LOOP
output_str:= output_str||harr1(i);
END LOOP;

harr1.DELETE;
if  pr_SP then
harr2.DELETE;
end if;

    if PR_Luhn then
    output_str:=substr(output_str,1,(length(output_str)-1))||Luhn(output_str);
    end if;
return output_str;
end perm;
```

Лістинг Д.1.2. Функція, яка власне реалізує метод зворотний до маскування:

```
function inv_perm (input_str in varchar2, name_table in varchar2, name_column in varchar2,
ID in number,  i_beg in number:=1, i_end in number:=0, PR_PRNG in number :=1, PR_Luhn in
BOOLEAN :=false, pr_SP in BOOLEAN :=false) return varchar2
is
  output_str    VARCHAR2(MAX_byte_size);
  j             NUMBER;
  j1            NUMBER;
  j1_1          NUMBER;
  j2            NUMBER;
  j3            NUMBER;
  p_algorithm   VARCHAR2(10) := 'MD5';
  harr1         Big_Char;
  harr2         Big_Int_PLS := Big_Int_PLS ();
  harr2_1       Big_Int_PLS := Big_Int_PLS ();
  harr3         Big_Char := Big_Char ();
  temp_arr      PLS_INTEGER;
  md            NUMBER;
  nn            PLS_INTEGER;
begin
harr1:=create_chr(input_str => input_str);
harr2.EXTEND(harr1.LAST);
harr3.EXTEND(harr1.LAST);
FOR i IN 1..harr1.LAST LOOP
```

```
harr2(i):=i;
harr3(i):=0;
END LOOP;
if  pr_SP then
harr2_1.EXTEND(harr1.LAST);
FOR i IN 1..harr1.LAST LOOP
harr2_1(i):=i;
END LOOP;
md:=length(str_ucod);
end if;
if ((name_table_T<>name_table) OR (name_column_T<>name_column)) then
  read_key(name_table, name_column);
  name_table_T:=name_table;
  name_column_T:=name_column;
end if;
j1:=hash_key(p_algorithm, ID);

if pr_SP then
j1_1:=j1;

   if PR_PRNG=2 then
      DBMS_RANDOM.initialize (MOD(j1_1,2147483647));
    elsif PR_PRNG=3 then
    Ran_init(j1_1);
    end if;
FOR i IN REVERSE  i_beg..(harr1.LAST-i_end) LOOP
CASE PR_PRNG
     WHEN 2  THEN j1_1 :=ROUND(DBMS_RANDOM.VALUE(1,md));
     WHEN 3  THEN j1_1 :=Ran;
     ELSE  j1_1 := MOD((1664525*j1_1+1013904223),4294967296);
  END CASE;
      j3:=MOD(j1_1,md);
      harr2_1(i):=j3;
END LOOP;

FOR i IN 1..harr1.LAST LOOP
nn:=INSTR(str_ucod,harr1(i));
j2:=MOD((nn-harr2_1(i)),md);
                  if j2<0 then
                   j2:=j2+md;
                  end if;
    harr1(i):= SUBSTR(str_ucod,j2,1);
END LOOP;
end if;
   if PR_PRNG=2 then
    DBMS_RANDOM.initialize (MOD(j1,2147483647));
    elsif PR_PRNG=3 then
    Ran_init(j1);
    end if;
FOR i IN REVERSE i_beg..(harr1.LAST-i_end) LOOP
CASE PR_PRNG
     WHEN 2  THEN j :=ROUND(DBMS_RANDOM.VALUE(i_beg,i));
     WHEN 3  THEN j1 :=Ran; j := MOD(j1,i)+1;
     ELSE  j1 := MOD((1664525*j1+1013904223),4294967296);
            j := MOD(j1,i)+1;
  END CASE;

if j<i_beg then j:=i_beg; end if;
temp_arr:=harr2(i);
harr2(i):=harr2(j);
harr2(j):=temp_arr;
END LOOP;

FOR i IN 1..harr1.LAST LOOP
harr3(harr2(i)):=harr1(i);
END LOOP;
output_str:='';
FOR i IN  1..harr3.LAST LOOP
output_str:= output_str||harr3(i);
END LOOP;
harr1.DELETE;
```

```
harr2.DELETE;
harr3.DELETE;
    if  pr_SP then
    harr2_1.DELETE;
    end if;

    if PR_Luhn then
    output_str:=substr(output_str,1,(length(output_str)-1))||Luhn(output_str);
    end if;

return output_str;

end inv_perm;
```

Д.2. Лістинги програм спеціальних функцій-оболонок

Лістинг Г.2.1. Функція-оболонка для генерації ключових послідовностей, що створена за допомогою пакета dbms_crypto, і приклад її використання:

```
CREATE OR REPLACE FUNCTION get_key (p_length IN PLS_INTEGER)
   RETURN RAW
IS
   l_ret   RAW (4000);
BEGIN
   l_ret := dbms_crypto.randombytes (p_length);
   RETURN l_ret;
end get_key;

select get_key(32) as KEY from dual;

KEY
--------------------------------------------------------------------------------
7929251641B541799D0C3E5F1487C33AB6A8A7FC89FB516F39FF055138F9134A
```

Лістинг Д.2.2. Функція-оболонка для зашифрування, що створена за допомогою пакета dbms_crypto, і приклад її використання:

```
CREATE OR REPLACE FUNCTION get_encrypt_val (
   p_in_val     IN VARCHAR2,            p_key IN VARCHAR2,
   p_algorithm  IN VARCHAR2 := 'AES128', p_reg IN VARCHAR2 := 'CBC',
   p_iv          IN   VARCHAR2 := NULL)
   RETURN VARCHAR2
IS
   l_enc_val    RAW (4000); l_enc_algo   PLS_INTEGER;
   l_in         RAW (4000); l_iv         RAW (4000);
   l_key        RAW (4000); l_reg        PLS_INTEGER; l_ret  VARCHAR2 (4000);
BEGIN
if   (UPPER(p_reg)<>'CBC')   and   (UPPER(p_reg)<>'ECB')   and   (UPPER(p_reg)<>'OFB')   and
(UPPER(p_reg)<>'CFB')
then raise_application_error (-20071, Ім'я режиму шифрування невірно задано.
Припустимі режими: CBC, ECB, OFB, CFB');
end if;
if (UPPER(p_algorithm)<>'DES') and (UPPER(p_algorithm)<>'3DES_2KEY') and
   (UPPER(p_algorithm)<>'3DES') and   (UPPER(p_algorithm)<>'AES128') and
   (UPPER(p_algorithm)<>'AES192') and (UPPER(p_algorithm)<>'AES256') and
   (UPPER(p_algorithm)<>'RC4')
then raise_application_error (-20072, Ім'я алгоритму шифрування невірно задано.
 Припустимі алгоритми: DES, 3DES_2KEY, 3DES, AES128, AES192, AES256, RC4');
end if;
l_reg:=
   CASE  UPPER(p_reg)
         WHEN 'CBC'  THEN DBMS_CRYPTO.chain_cbc
         WHEN 'ECB'  THEN DBMS_CRYPTO.chain_ecb
         WHEN 'CFB'  THEN DBMS_CRYPTO.chain_cfb
         WHEN 'OFB'  THEN DBMS_CRYPTO.chain_ofb
```

```
      END;
   l_enc_algo :=
      CASE  UPPER(p_algorithm)
         WHEN 'DES'        THEN DBMS_CRYPTO.encrypt_des
         WHEN '3DES_2KEY' THEN DBMS_CRYPTO.encrypt_3des_2key
         WHEN '3DES'       THEN DBMS_CRYPTO.encrypt_3des
         WHEN 'AES128'     THEN DBMS_CRYPTO.encrypt_aes128
         WHEN 'AES192'     THEN DBMS_CRYPTO.encrypt_aes192
         WHEN 'AES256'     THEN dbms_crypto.encrypt_aes256
         WHEN 'RC4'        THEN DBMS_CRYPTO.encrypt_rc4
      END;
   l_in := utl_i18n.string_to_raw (p_in_val, 'AL32UTF8');
   l_iv := utl_i18n.string_to_raw (p_iv, 'AL32UTF8');
   l_key := utl_i18n.string_to_raw (p_key, 'AL32UTF8');
   l_enc_val := DBMS_CRYPTO.encrypt (src => l_in, KEY => l_key, iv => l_iv,
                        typ => l_enc_algo + l_reg + DBMS_CRYPTO.pad_pkcs5);
   l_ret := RAWTOHEX (l_enc_val);
   RETURN l_ret;
end get_encr_val;

SELECT   get_encrypt_val   ('Test0','1234567890123456','aes128','cfb')   as   ENCRYPT_VAL
FROM dual;

ENCRYPT_VAL
--------------------------------------------------------------------------------
8CD0EB3CF76C079FB9905FD93C952571
```

Лістинг Д.2.3. Функція-оболонка для розшифрування, що створена за допомогою пакета dbms_crypto, і приклад її використання:

```
CREATE OR REPLACE FUNCTION get_decrypt_val (p_in_val   IN   VARCHAR2,
   p_key    IN   VARCHAR2,        p_algorithm   IN   VARCHAR2 := 'AES128',
   p_reg  IN   VARCHAR2 := 'CBC', p_iv          IN   VARCHAR2 := NULL)
   RETURN VARCHAR2
IS
   l_dec_val RAW (4000); l_enc_algo PLS_INTEGER;
   l_in      RAW (4000); l_iv       RAW (4000);
   l_key     RAW (4000); l_ret      VARCHAR2 (4000); l_reg PLS_INTEGER;
BEGIN
if   (UPPER(p_reg)<>'CBC')   and   (UPPER(p_reg)<>'ECB')   and   (UPPER(p_reg)<>'OFB')   and
(UPPER(p_reg)<>'CFB')
then raise_application_error (-20073, 'Ім'я режиму розшифрування невірно задано. Припустимі
режими: CBC, ECB, OFB, CFB');
end if;
if (UPPER(p_algorithm)<>'DES') and (UPPER(p_algorithm)<>'3DES_2KEY') and
   (UPPER(p_algorithm)<>'3DES') and   (UPPER(p_algorithm)<>'AES128') and
   (UPPER(p_algorithm)<>'AES192') and (UPPER(p_algorithm)<>'AES256') and
   (UPPER(p_algorithm)<>'RC4')
then  raise_application_error  (-20074,  'Ім'я  алгоритму  розшифрування  невірно  задано.
Припустимі алгоритми: DES, 3DES_2KEY, 3DES, AES128, AES192, AES256, RC4');
end if;
l_reg:=
   CASE  UPPER(p_reg)
         WHEN 'CBC' THEN DBMS_CRYPTO.chain_cbc
         WHEN 'ECB' THEN DBMS_CRYPTO.chain_ecb
         WHEN 'CFB' THEN DBMS_CRYPTO.chain_cfb
         WHEN 'OFB' THEN DBMS_CRYPTO.chain_ofb
      END;
   l_enc_algo :=
      CASE  UPPER(p_algorithm)
         WHEN 'DES'        THEN DBMS_CRYPTO.encrypt_des
         WHEN '3DES_2KEY'  THEN DBMS_CRYPTO.encrypt_3des_2key
         WHEN '3DES'       THEN DBMS_CRYPTO.encrypt_3des
         WHEN 'AES128'     THEN DBMS_CRYPTO.encrypt_aes128
         WHEN 'AES192'     THEN DBMS_CRYPTO.encrypt_aes192
         WHEN 'AES256'     THEN DBMS_CRYPTO.encrypt_aes256
         WHEN 'RC4'        THEN DBMS_CRYPTO.encrypt_rc4
      END;
```

```
   l_in := hextoraw(p_in_val);
   l_iv := utl_i18n.string_to_raw (p_iv, 'AL32UTF8');
   l_key := utl_i18n.string_to_raw (p_key, 'AL32UTF8');
   l_dec_val := DBMS_CRYPTO.decrypt (src => l_in, KEY => l_key, iv  => l_iv,
                          typ => l_enc_algo + l_reg + DBMS_CRYPTO.pad_pkcs5);
   l_ret := utl_i18n.raw_to_char (l_dec_val, 'AL32UTF8');
   RETURN l_ret;
end  get_decr_val;

SELECT   get_decrypt_val   (p_in_val   =>   '8CD0EB3CF76C079FB9905FD93C952571',   p_key =>
'1234567890123456',p_algorithm => 'aes128',p_reg => 'cfb') as DECRYPT_VAL FROM dual;

DECRYPT_VAL
--------------------------------------------------------------------------------
Test0
```

Д.3. Приклад шаблону команд з управління доступом

Приклад шаблону команд для СКБД Oracle:

```
create or replace package rls_utils is
…
function get_predicate(dml_stmt_p number, object_name_p varchar2)
return varchar2;
function get_access_rule_predicate(
schema_p in varchar2, object_p in varchar2) return varchar2;
…
end;
/
create or replace package body rls_utils is
…
  function get_access_rule_predicate
(schema_p varchar2, object_p varchar2)
return varchar2
  as
  begin
    return (get_predicate(DML_access_rule, object_p));
  end;
…
end;
/
{
begin
dbms_rls.add_policy(object_schema   => '{user_name}',
            object_name     => '{table_name}',
            policy_name     => '{RLS_user_name_access_rule}',
            function_schema => '{user_name}',
            policy_function => '{RLS_UTILS.GET_access_rule_PREDICATE}',
            statement_types => '[SELECT] [,INSERT] [,DELETE] [,UPDATE]',
            update_check    => {TRUE|FALSE},
            enable          => {TRUE|FALSE},
            static_policy   => {TRUE|FALSE}
          );
end;
/
}
```

Позначення, що використовуються у шаблоні:

– змінні (виділені жирним курсивним шрифтом): *user_name* – ім'я користувача; *table_name* – ім'я базового відношення $R_i^{sh} \in R^{sh}$, що підлягає захисту

за допомогою політики; *access_rule* – комбінації операцій (select, update, delete, insert – $oper_i^j$, вираз (4.4)) доступу до відношення, зазначеного в *table_name*;

– параметри процедури add_policy стандартного пакета Oracle dbms_rls: policy_name – ім'я (назва) політики *RLS*, яка застосовується для базового відношення $R_i^{sh} \in R^{sh}$; function_schema – ім'я схеми (користувача), що володіє функцією правил; policy_function – ім'я функції (із зазначенням імені пакета), що генерує предикат для базового відношення $R_i^{sh} \in R^{sh}$; statement_types – типи операторів, до яких застосовується функція правил; update_check (зі значенням true) – даний параметр подібний до конструкції *with check option* для уявлень, він гарантує неможливість вставки або зміни рядка так, що після вставки або зміни не можна буде її побачити (для типів INSERT і UPDATE цей параметр є необов'язковим і за замовчуванням набуває значення false: якщо його встановлено в true, стратегія також перевіряється для операторів INSERT і UPDATE під час перевірки операцій SELECT або DELETE); enable (зі значенням true) – параметр, що вказує на активізацію політики безпосередньо після її додавання (за умовчанням набуває значення true); static_policy (зі значенням false) – якщо цей параметр дорівнює true, політика генерує той самий рядок предиката для будь-якого користувача, який намагається звернутися до об'єкта, за винятком користувача SYS або будь-якого користувача з привілеєм EXEMPT ACCESS POLICY (значення за замовчуванням false);

– фігурні дужки – { }, квадратні дужки – [ ], символ | – відповідають позначенням, запозиченим з розширеної нотації Бекуса-Наура;

решта буквено-цифрових символів є або ключовими словами мови, або іменами стандартних пакетів, або прийнятими рядковими літералами.

– именами стандартных пакетов, либо принятыми строковыми литералами.

ДОДАТОК Е.

РЕЗУЛЬТАТИ ОЦІНКИ ОСНОВНИХ КОМПОНЕНТ БАР'ЄРІВ БЕЗПЕКИ

Таблиця Е.1 – Результати оцінки основних компонент бар'єрів безпеки

№ бар'єру	Загроза (t)	$\frac{P_t^{\text{verbal}}}{P_t^{\text{quant}}}$	Вразливість (γ)	$\frac{P_\gamma^{\text{verbal}}}{P_\gamma^{\text{quant}}}$	Захід безпеки (w)	$\frac{R^{\text{UBR}_{\text{verbal}}}}{R^{\text{UBR}_{\text{quant}}}}$	$\frac{R^{\text{RDB}_{\text{verbal}}}}{R^{\text{RDB}_{\text{quant}}}}$	Об'єкт (o)	Rr^{UBR}	Rr^{RDB}
1.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
2.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
3.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
4.	t_1	«Н» / 0.1	γ_1	«С» / 0.5	w_4	«В» / 0.8	«Н» / 0	o_5	0.002	0.01
5.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_3	0.008	0.008
6.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
7.	t_1	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
8.	t_1	«С» / 0.4	γ_{18}	«С» / 0.5	w_5	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
9.	t_2	«С» / 0.4	γ_5	«В» / 0.85	w_6	«В» / 0.8	«С» / 0.6	o_4	0.0136	0.0272
10.	t_2	«С» / 0.4	γ_5	«В» / 0.85	w_6	«В» / 0.8	«С» / 0.6	o_5	0.0136	0.0272
11.	t_2	«С» / 0.4	γ_5	«В» / 0.85	w_7	«В» / 0.8	«С» / 0.6	o_7	0.0136	0.0272
12.	t_3	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
13.	t_3	«С» / 0.4	γ_1	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
14.	t_3	«С» / 0.4	γ_1	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
15.	t_3	«Н» / 0.1	γ_1	«С» / 0.5	w_4	«В» / 0.8	«Н» / 0	o_5	0.002	0.01
16.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
17.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
18.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
19.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_3	0.008	0.008
20.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
21.	t_3	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
22.	t_3	«С» / 0.4	γ_3	«С» / 0.5	w_8	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
23.	t_3	«С» / 0.4	γ_4	«С» / 0.5	w_9	«С» / 0.4	«С» / 0.4	o_1	0.024	0.024
24.	t_3	«С» / 0.4	γ_{10}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_6	0.004	0.024
25.	t_3	«С» / 0.4	γ_{10}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_7	0.004	0.024
26.	t_3	«С» / 0.4	γ_{11}	«С» / 0.5	w_{11}	«В» / 0.8	«В» / 0.8	o_2	0.008	0.008
27.	t_3	«С» / 0.4	γ_{12}	«С» / 0.5	w_{12}	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
28.	t_3	«С» / 0.4	γ_{12}	«С» / 0.5	w_{12}	«В» / 0.8	«В» / 0.8	o_2	0.008	0.008
29.	t_3	«С» / 0.4	γ_{13}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_6	0.004	0.024
30.	t_3	«С» / 0.4	γ_{13}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_7	0.004	0.024
31.	t_3	«С» / 0.4	γ_{13}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008

№ бар'єру	Загроза (t)	$\frac{P_t^{verbal}}{P_t^{quant}}$	Вразливість (γ)	$\frac{P_\gamma^{verbal}}{P_\gamma^{quant}}$	Захід безпеки (w)	$\frac{R^{UBR_{verbal}}}{R^{UBR_{quant}}}$	$\frac{R^{RDB_{verbal}}}{R^{RDB_{quant}}}$	Об'єкт (o)	Rr^{UBR}	Rr^{RDB}
32.	t_3	«С» / 0.4	γ_{13}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
33.	t_3	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
34.	t_3	«С» / 0.4	γ_{14}	«С» / 0.5	w_{15}	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
35.	t_3	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_3	0.006	0.008
36.	t_3	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_6	0.006	0.008
37.	t_3	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_7	0.006	0.008
38.	t_3	«С» / 0.4	γ_{15}	«С» / 0.5	w_{16}	«В» / 0.9	«В» / 0.8	o_1	0.004	0.008
39.	t_3	«С» / 0.4	γ_{15}	«С» / 0.5	w_{16}	«В» / 0.9	«В» / 0.8	o_6	0.004	0.008
40.	t_3	«С» / 0.4	γ_{15}	«С» / 0.5	w_{16}	«В» / 0.9	«В» / 0.8	o_7	0.004	0.008
41.	t_3	«С» / 0.4	γ_{16}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
42.	t_3	«С» / 0.4	γ_{16}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
43.	t_3	«С» / 0.4	γ_{18}	«С» / 0.5	w_5	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
44.	t_4	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
45.	t_4	«С» / 0.4	γ_1	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
46.	t_4	«С» / 0.4	γ_1	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
47.	t_4	«Н» / 0.1	γ_1	«С» / 0.5	w_4	«В» / 0.8	«Н» / 0	o_5	0.002	0.01
48.	t_4	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
49.	t_4	«С» / 0.4	γ_2	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
50.	t_4	«С» / 0.4	γ_2	«С» / 0.5	w_3	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
51.	t_4	«С» / 0.4	γ_3	«С» / 0.5	w_8	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
52.	t_4	«С» / 0.4	γ_4	«С» / 0.5	w_9	«С» / 0.4	«С» / 0.4	o_1	0.024	0.024
53.	t_4	«С» / 0.4	γ_{10}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_6	0.004	0.024
54.	t_4	«С» / 0.4	γ_{10}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_7	0.004	0.024
55.	t_4	«С» / 0.4	γ_{11}	«С» / 0.5	w_{11}	«В» / 0.8	«В» / 0.8	o_2	0.008	0.008
56.	t_4	«С» / 0.4	γ_{12}	«С» / 0.5	w_{12}	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
57.	t_4	«С» / 0.4	γ_{13}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_6	0.004	0.024
58.	t_4	«С» / 0.4	γ_{13}	«С» / 0.5	w_{10}	«В» / 0.9	«С» / 0.4	o_7	0.004	0.024
59.	t_4	«С» / 0.4	γ_{13}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
60.	t_4	«С» / 0.4	γ_{13}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
61.	t_4	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
62.	t_4	«С» / 0.4	γ_{14}	«С» / 0.5	w_{15}	«В» / 0.85	«В» / 0.8	o_4	0.006	0.008
63.	t_4	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_3	0.006	0.008
64.	t_4	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_6	0.006	0.008
65.	t_4	«С» / 0.4	γ_{14}	«С» / 0.5	w_{14}	«В» / 0.85	«В» / 0.8	o_7	0.006	0.008
66.	t_4	«С» / 0.4	γ_{15}	«С» / 0.5	w_{16}	«В» / 0.85	«В» / 0.8	o_6	0.006	0.008
67.	t_4	«С» / 0.4	γ_{15}	«С» / 0.5	w_{16}	«В» / 0.85	«В» / 0.8	o_7	0.006	0.008

№ бар'єру	Загроза (t)	$\frac{P_t^{verbal}}{P_t^{quant}}$	Вразливість (γ)	$\frac{P_\gamma^{verbal}}{P_\gamma^{quant}}$	Захід безпеки (w)	$\frac{R^{UBR_{verbal}}}{R^{UBR_{quant}}}$	$\frac{R^{RDB_{verbal}}}{R^{RDB_{quant}}}$	Об'єкт (o)	Rr^{UBR}	Rr^{RDB}
68.	t_4	«С» / 0.4	γ_{16}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
69.	t_4	«С» / 0.4	γ_{16}	«С» / 0.5	w_{13}	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008
70.	t_4	«С» / 0.4	γ_{18}	«С» / 0.5	w_5	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
71.	t_5	«С» / 0.4	γ_1	«С» / 0.5	w_{17}	«В» / 0.85	«В» / 0.8	o_1	0.006	0.008
72.	t_5	«С» / 0.4	γ_2	«С» / 0.5	w_{17}	«В» / 0.85	«В» / 0.8	o_1	0.006	0.008
73.	t_5	«С» / 0.4	γ_3	«С» / 0.5	w_{17}	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
74.	t_5	«С» / 0.4	γ_4	«С» / 0.5	w_{17}	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
75.	t_5	«С» / 0.4	γ_{18}	«С» / 0.5	w_{17}	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
76.	t_6	«С» / 0.4	γ_5	«С» / 0.5	w_{18}	«С» / 0.6	«Н» / 0	o_2	0.016	0.04
77.	t_6	«С» / 0.4	γ_5	«С» / 0.5	w_{19}	«С» / 0.6	«Н» / 0	o_6	0.016	0.04
78.	t_6	«С» / 0.4	γ_5	«С» / 0.5	w_{19}	«С» / 0.6	«Н» / 0	o_7	0.016	0.04
79.	t_6	«С» / 0.4	γ_6	«С» / 0.5	w_{20}	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
80.	t_6	«С» / 0.4	γ_8	«С» / 0.5	w_{21}	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
81.	t_6	«С» / 0.4	γ_9	«С» / 0.5	w_{21}	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
82.	t_7	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
83.	t_7	«С» / 0.4	γ_3	«С» / 0.5	w_8	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
84.	t_7	«С» / 0.4	γ_4	«С» / 0.5	w_9	«С» / 0.4	«С» / 0.4	o_1	0.024	0.024
85.	t_7	«С» / 0.4	γ_{18}	«С» / 0.5	w_5	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
86.	t_8	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
87.	t_8	«С» / 0.4	γ_2	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
88.	t_8	«С» / 0.4	γ_3	«С» / 0.5	w_8	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
89.	t_8	«С» / 0.4	γ_5	«В» / 0.85	w_{18}	«С» / 0.6	«Н» / 0	o_2	0.0272	0.068
90.	t_8	«С» / 0.4	γ_5	«С» / 0.5	w_{19}	«С» / 0.6	«Н» / 0	o_6	0.016	0.04
91.	t_8	«С» / 0.4	γ_5	«С» / 0.5	w_{19}	«С» / 0.6	«Н» / 0	o_7	0.016	0.04
92.	t_8	«С» / 0.4	γ_7	«В» / 0.85	w_{22}	«С» / 0.6	«С» / 0.6	o_2	0.0272	0.0272
93.	t_8	«С» / 0.4	γ_{17}	«С» / 0.5	w_{23}	«С» / 0.6	«С» / 0.6	o_1	0.016	0.016
94.	t_9	«С» / 0.4	γ_5	«В» / 0.85	w_6	«В» / 0.8	«С» / 0.6	o_4	0.0136	0.0272
95.	t_9	«С» / 0.4	γ_5	«В» / 0.85	w_6	«В» / 0.8	«С» / 0.6	o_5	0.0136	0.0272
96.	t_9	«С» / 0.4	γ_{11}	«С» / 0.5	w_{24}	«В» / 0.8	«В» / 0.8	o_2	0.008	0.008
97.	t_9	«С» / 0.4	γ_{14}	«С» / 0.5	w_6	«В» / 0.8	«С» / 0.6	o_4	0.008	0.016
98.	t_9	«С» / 0.4	γ_{14}	«С» / 0.5	w_6	«В» / 0.8	«С» / 0.6	o_5	0.008	0.016
99.	t_{10}	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_1	0.008	0.008
100.	t_{10}	«С» / 0.4	γ_1	«С» / 0.5	w_2	«В» / 0.85	«В» / 0.8	o_2	0.006	0.008
101.	t_{10}	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_3	0.008	0.008
102.	t_{10}	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_6	0.008	0.008
103.	t_{10}	«С» / 0.4	γ_1	«С» / 0.5	w_1	«В» / 0.8	«В» / 0.8	o_7	0.008	0.008

№ бар'єру	Загроза (t)	$\frac{P_t^{verbal}}{P_t^{quant}}$	Вразливість (γ)	$\frac{P_\gamma^{verbal}}{P_\gamma^{quant}}$	Захід безпеки (w)	$\frac{R^{UBR_{verbal}}}{R^{UBR_{quant}}}$	$\frac{R^{RDB_{verbal}}}{R^{RDB_{quant}}}$	Об'єкт (o)	Rr^{UBR}	Rr^{RDB}
104.	t_{10}	«C» / 0.4	γ_2	«B» / 0.85	w_1	«B» / 0.8	«B» / 0.8	o_1	0.0136	0.0136
105.	t_{10}	«C» / 0.4	γ_2	«C» / 0.5	w_2	«B» / 0.85	«B» / 0.8	o_2	0.006	0.008
106.	t_{10}	«C» / 0.4	γ_2	«C» / 0.5	w_1	«B» / 0.8	«B» / 0.8	o_3	0.008	0.008
107.	t_{10}	«C» / 0.4	γ_2	«C» / 0.5	w_1	«B» / 0.8	«B» / 0.8	o_6	0.008	0.008
108.	t_{10}	«C» / 0.4	γ_2	«C» / 0.5	w_1	«B» / 0.8	«B» / 0.8	o_7	0.008	0.008
109.	t_{10}	«C» / 0.4	γ_3	«C» / 0.5	w_8	«B» / 0.8	«B» / 0.8	o_1	0.008	0.008
110.	t_{10}	«C» / 0.4	γ_4	«C» / 0.5	w_9	«C» / 0.4	«C» / 0.4	o_1	0.024	0.024
111.	t_{10}	«C» / 0.4	γ_{10}	«C» / 0.5	w_{10}	«B» / 0.9	«C» / 0.4	o_6	0.004	0.024
112.	t_{10}	«C» / 0.4	γ_{10}	«C» / 0.5	w_{10}	«B» / 0.9	«C» / 0.4	o_7	0.004	0.024
113.	t_{10}	«C» / 0.4	γ_{12}	«C» / 0.5	w_{12}	«B» / 0.8	«B» / 0.8	o_7	0.008	0.008
114.	t_{10}	«C» / 0.4	γ_{13}	«C» / 0.5	w_{10}	«B» / 0.9	«C» / 0.4	o_6	0.004	0.024
115.	t_{10}	«C» / 0.4	γ_{13}	«C» / 0.5	w_{10}	«B» / 0.9	«C» / 0.4	o_7	0.004	0.024
116.	t_{10}	«C» / 0.4	γ_{13}	«C» / 0.5	w_{13}	«B» / 0.8	«B» / 0.8	o_6	0.008	0.008
117.	t_{10}	«C» / 0.4	γ_{13}	«C» / 0.5	w_{13}	«B» / 0.8	«B» / 0.8	o_7	0.008	0.008
118.	t_{10}	«C» / 0.4	γ_{16}	«C» / 0.5	w_{13}	«B» / 0.8	«B» / 0.8	o_6	0.008	0.008
119.	t_{10}	«C» / 0.4	γ_{16}	«C» / 0.5	w_{13}	«B» / 0.8	«B» / 0.8	o_7	0.008	0.008
120.	t_{10}	«C» / 0.4	γ_{18}	«C» / 0.5	w_5	«B» / 0.8	«B» / 0.8	o_1	0.008	0.008
121.	t_{11}	«C» / 0.4	γ_{17}	«C» / 0.5	w_{23}	«B» / 0.8	«B» / 0.8	o_1	0.008	0.008

Загрози безпеки бази даних:

- t_1 – надмірні привілеї та привілеї, що не використовуються;
- t_2 – зловживання законними привілеями;
- t_3 – ін'єкції введення;
- t_4 – зловмисне програмне забезпечення;
- t_5 – недостатність заходів з аудиту даних;
- t_6 – незахищеність носіїв (резервних копій) інформації;
- t_7 – експлуатація вразливих, неправильно сконфігурованих баз даних;
- t_8 – некеровані конфіденційні дані;
- t_9 – логічний висновок;
- t_{10} – відмова в обслуговуванні;
- t_{11} – брак знання та досвіду з питань ІБ.

Типи вразливостей:

- γ_1 – неналежне керування привілеями;
- γ_2 – неналежна авторизація;
- γ_3 – неналежна автентифікація;
- γ_4 – неконтрольоване споживання ресурсів;
- γ_5 – зберігання конфіденційної інформації у відкритому вигляді;
- γ_6 – недостатня стійкість шифрування;
- γ_7 – неналежне очищення конфіденційних даних із виведеного з експлуатації пристрою;
- γ_8 – використання зламаного або небезпечного криптографічного алгоритму;
- γ_9 – використання недостатньо випадкових значень;
- γ_{10} – недостатня перевірка автентичності даних;
- γ_{11} – неправильна перевірка введених даних;
- γ_{12} – використання забороненого коду;
- γ_{13} – вбудований зловмисний код;
- γ_{14} – порушення принципів безпечного проектування;
- γ_{15} – неправильне надання зазначеної функціональності;
- γ_{16} – прихована функціональність;
- γ_{17} – неповна документація;
- γ_{18} – вада конфігурації.

Об'єкти захисту:

- o_2 – таблиці;
- o_3 – уявлення (англ. views);
- o_4 – кортежі (рядки) таблиць;
- o_5 – окремі поля (значення атрибутів) рядків;
- o_6 – тригери;
- o_7 – модулі, що постійно зберігаються.

ДОДАТОК Ж.
АКТИ ВПРОВАДЖЕНЬ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОЇ РОБОТИ

"ЗАТВЕРДЖУЮ"
Технічний директор
Приватного акціонерного товариства
«Інститут інформаційних технологій»
_____ Олександр ШУМОВ
_____ 2024 р.

АКТ
впровадження результатів дисертаційної роботи
на здобуття ступеня доктора філософії
Вілігури Владислава Вікторовича
у Приватному акціонерному товаристві «Інститут інформаційних технологій»

Комісія у складі: голови комісії – першого заступника головного конструктора АТ «ІІТ», кандидата технічних наук Горбенко Ю. І., та членів комісії – начальника відділу апаратних засобів захисту інформації АТ «ІІТ», кандидата технічних наук Бобуха В. А. та інженера з систем захисту інформації АТ «ІІТ» Бакликова О. О. встановила, що у Приватному акціонерному товаристві «Інститут інформаційних технологій» впроваджені наступні результати, що одержані Вілігурою Владиславом Вікторовичем в процесі виконання дисертаційних досліджень:

– метод маскування елементів даних не ключових полів кортежів таблиць виробничої бази даних, орієнтований на заплутування, псевдонімізацію даних та ускладнення реалізації загрози логічного висновку, дозволяє зменшити час на відповідні операції перетворення на (10-17) % щодо методу класичного шифрування, при цьому не наводячи до зміни формату та збільшення розмірності даних, що зберігаються. Даний метод може бути також використаний у невиробничих базах даних, розширюючи можливості так званого статичного маскування даних.

Впровадження отриманих наукових та практичних результатів дозволяє при менших обчислювальних витратах на перетворення і без зміни формату вихідних даних забезпечити ефективне приховування даних, яке ускладнює реалізацію загрози логічного висновку.

Голова комісії:
перший заступник головного конструктора
Приватного акціонерного товариства
«Інститут інформаційних технологій»
кандидат технічних наук Юрій ГОРБЕНКО

Члени комісії:
начальник відділу апаратних засобів захисту інформації
Приватного акціонерного товариства
«Інститут інформаційних технологій»
кандидат технічних наук Всеволод БОБУХ

інженер з систем захисту інформації
Приватного акціонерного товариства
«Інститут інформаційних технологій» Олександр БАКЛИКОВ

«ЗАТВЕРДЖУЮ»
Проректор з науково-педагогічної роботи
Харківського національного університету
імені В. Н. Каразіна
_____ Антон ПАНТЕЛЕЙМОНОВ
«___» _____ 2024 р.

АКТ
впровадження результатів дисертаційної роботи
на здобуття ступеня доктора філософії
Вілігури Владислава Вікторовича
у наукову роботу кафедри безпеки інформаційних систем і технологій
Харківського національного університету імені В. Н. Каразіна

Комісія у складі голови комісії – виконуючої обов'язки завідувача кафедри безпеки інформаційних систем і технологій, кандидата технічних наук, доцента Мелкозьорової О. М. та членів комісії: доцента кафедри безпеки інформаційних систем і технологій, кандидата технічних наук, старшого наукового співробітника Малахова С. В., наукового співробітника НДЧ Харківського національного університету імені В. Н. Каразіна, кандидата технічних наук Пономара В. А. встановила, що у Харківському національному університеті імені В. Н. Каразіна впроваджені результати дисертаційних досліджень, що одержані Вілігурою В. В. при виконанні науково-дослідних робіт: № 39-18 «Механізми, методи, протоколи та засоби криптографічного захисту інформації у пост квантовий період» (Шифр Квант-2019), № 29-19 «Механізми та засоби електронного підпису у пост квантовий період» (Шифр «Квант-2020»), № 28-20 «Механізми та засоби асиметричних криптоперетворень у постквантовий період» (Шифр «Квант-2021»), «Математичні та програмні моделі, методи та механізми криптографічного захисту інформації для пост-квантового середовища в інтересах національної безпеки держави» (№ ДР 0121U109939), № 34-21 «Методи та алгоритми постквантових криптоперетворень, їх стандартизація та впровадження» (Шифр «Квант-2022»), № 09-22 «Методи та засоби генерування псевдовипадкових та випадкових послідовностей на основі класичних та квантових ефектів» (Шифр «Квант-2023») в частині аналізу формальних моделей безпеки та їх застосовності для баз даних, розробки методів та засобів, що підвищують безпеку баз, сховищ даних, і дозволяють оцінити основні компоненти бар'єрів безпеки та захищеність бази даних.

Голова комісії
к.т.н., доцент — Ольга МЕЛКОЗЬОРОВА

Члени комісії:

к.т.н., с.н.с. — Сергей МАЛАХОВ

к.т.н. — Володимир ПОНОМАР

«ЗАТВЕРДЖУЮ»
Проректор з науково-педагогічної роботи
Харківського національного університету
імені В. Н. Каразіна
_____ Антон ПАНТЕЛЕЙМОНОВ
«___» _____ 2024 р.

АКТ
впровадження результатів дисертаційної роботи
на здобуття ступеня доктора філософії
Вілігури Владислава Вікторовича
у навчальний процес Харківського національного університету імені В. Н. Каразіна

Комісія у складі голови комісії – виконуючої обов'язки завідувача кафедри безпеки інформаційних систем і технологій, кандидата технічних наук, доцента Мелкозьорової О. М. та членів комісії: професора кафедри безпеки інформаційних систем і технологій, доктора технічних наук, професора Лисицької І. В., доцента кафедри безпеки інформаційних систем і технологій, кандидата технічних наук Колованової Є. П. встановила, що у Харківському національному університеті імені В. Н. Каразіна впроваджені наступні результати, що одержані Вілігурою В. В. в процесі виконання дисертаційних досліджень.

1. З дисципліни «Теорія розподілених інформаційних ресурсів, захист баз даних та знань» за спеціальністю «125 Кібербезпека та захист інформації», другого (магістерського) рівня вищої освіти при підготовці та читанні лекцій, практичних занять за темами розділу 2 – «Теорія захисту розподілених інформаційних систем», зокрема використані результати застосування запропонованих в дисертаційної роботі методів та засобів, що підвищують безпеку баз даних, в тому числі, методу моніторингу, що грунтується на можливостях технології блокчейн, який дозволяє за рахунок використання створеної зумовленої структури, правил формування первинного та наступних блоків у блокчейновому ланцюжку, організації зберігання цієї структури в рамках реляційної моделі даних, способів обчислення кореня геш-дерева, суворо контролювати набір програм БД, їх цілісність, справжність при менших обсягах збережених для цього даних і необхідних ресурсів процесора.

2. З дисципліни «Захист інформації в інформаційно-комунікаційних системах» за спеціальністю «125 Кібербезпека та захист інформації», першого (бакалаврського) рівня вищої освіти при підготовці та читанні лекцій, лабораторних занять за темами розділу 5 – «Основи захисту в СКБД», зокрема використані результати застосування методу приховування коду збережених у базі даних програм, який дозволяє за рахунок випадкової перестановки (що спирається на сучасний варіант алгоритму тасування Фішера-Йейтса) символів коду з можливою заміною кожного з них на інший випадково вибраний із стандарту Unicode забезпечити більше ефективний захист коду від його розкриття зловмисником, при цьому гарантуючи цілісність коду.

Реалізація результатів наукових досліджень Вілігури В. В. дозволяє підвищити рівень і якість викладання навчального матеріалу дисциплін за рахунок вивчення нових методів та засобів забезпечення безпеки баз та сховищ даних.

Голова комісії
к.т.н., доцент — Ольга МЕЛКОЗЬОРОВА
Члени комісії:
д.т.н., професор — Ірина ЛИСИЦЬКА
к.т.н. — Євгенія КОЛОВАНОВА

Онлайн сервіс створення та перевірки кваліфікованого та удосконаленого електронного підпису

ПРОТОКОЛ
створення та перевірки кваліфікованого та удосконаленого електронного підпису

Дата та час: 18:36:33 27.11.2024

Назва файлу з підписом: Corr. 11_10_2024 Дисертація_Вілігура_NEW_NEW.pdf.p7s
Розмір файлу з підписом: 7.4 МБ

Перевірені файли:
Назва файлу без підпису: Corr. 11_10_2024 Дисертація_Вілігура_NEW_NEW.pdf
Розмір файлу без підпису: 7.4 МБ

Результат перевірки підпису: Підпис створено та перевірено успішно. Цілісність даних підтверджено

Підписувач: ВІЛІГУРА ВЛАДИСЛАВ ВІКТОРОВИЧ
П.І.Б.: ВІЛІГУРА ВЛАДИСЛАВ ВІКТОРОВИЧ
Країна: Україна
РНОКПП: 3464005335
Організація (установа): ФІЗИЧНА ОСОБА
Час підпису (підтверджено кваліфікованою позначкою часу для підпису від Надавача): 18:36:32 27.11.2024
Сертифікат виданий: КНЕДП АЦСК АТ КБ "ПРИВАТБАНК"
Серійний номер: 5E984D526F82F38F04000000394D4701F51BCD04
Алгоритм підпису: ДСТУ 4145
Тип підпису: Удосконалений
Тип контейнера: Підпис та дані в одному файлі (CAdES enveloped)
Формат підпису: З повними даними ЦСК для перевірки (CAdES-X Long)
Сертифікат: Кваліфікований

Версія від: 2024.10.24 15:00