НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет електроніки
(повна назва інституту/факультету)

Кафедра акустичних та мультимедійних електронних систем
(повна назва кафедри)

«На правах рукопису»
УДК 621.396.1

«До захисту допущено»
Завідувач кафедри
С.А. Найда
(ініціали, прізвище)
"7" грудня 2020 р.

Магістерська дисертація

зі спеціальності: 171 «Електроніка» (Електронні системи мультимедіа та засоби Інтернету речей)
(код і назва)

на тему: «Методи захисту корпоративних даних в банківських мережах»

Виконав: студент II курсу, гр ДВ-82мп
(шифр групи)

Сорока Ярослав Олександрович
(прізвище, ім'я, по батькові) (підпис)

Керівник доцент, к.т.н. Попович П.В.
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант
(науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент доцент кафедри ЕІ, к.т.н., доц. Іванько К.О.
(посада,науковийступінь,вченезвання,науковийступінь,прізвище,ініціали)

(підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент
(підпис)

Київ – 2020 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет електроніки

Кафедра акустичних та мультимедійних електронних систем

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою

Спеціальність 171 «Електроніка» («Електронні системи мультимедіа та засоби Інтернету речей»)

ЗАТВЕРДЖУЮ
Завідувач кафедри
(підпис) С.А. Найда
(ініціали, прізвище)
« 07 » грудня 2020 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Сороці Ярославу Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Методи захисту корпоративних даних в банківських мережах»
керівник роботи Попович Павло Васильович, доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від від «5» листопада 2020 р. № 3241-с

2. Строк подання студентом дисертації 07 грудня 2020 р.

3. Об'єкт дослідження методи і засоби захисту та забезпечення безпеки інформації.

4. Предмет дослідження (Вихідні дані – для магістерської дисертації за освітньо-професійною програмою) створення баз даних та методи їх захисту.

5. Перелік завдань, які потрібно розробити: дослідити методи забезпечення інформаційної безпеки в інформаційних системах та їх особливості, пов'язані з корпоративними банківськими мережами; проаналізувати алгоритми

забезпечення цілісності даних в корпоративних мережах; дослідити методи захисту даних та засоби їх реалізації; дослідити практичні аспекти реалізації методів захисту баз даних в корпоративних мережах.

6. Перелік графічного (ілюстративного) матеріалу 12-14 слайдів презентації: характеристика роботи, формулювання завдання роботи, загальні характеристики бази даних, створення бази даних в середовищі SQL, розрахунок основних параметрів бази даних та моделювання, висновки.

7. Дата видачі завдання 01.09.2020 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Написання першого розділу	01.09.2020-21.09.2020	Виконано
2	Написання другого розділу	21.09.2020-15.10.2020	Виконано
3	Написання третього розділу	15.10.2020-01.11.2020	Виконано
4	Написання четвертого розділу	01.11.2020-15.11.2020	Виконано
	Написання п’ятого розділу	15.11.2020-30.11.2020	Виконано
5	Підготовка матеріалів до друку та оформлення пояснювальної записки	02.12.2020-05.12.2020	Виконано
6	Підготовка та оформлення презентації для доповіді	06.12.2020-12.12.2020	Виконано

Студент _____ Я.О. Сорока
(підпис) (ініціали, прізвище)

Керівник роботи _____ П.В. Попович
(підпис) (ініціали, прізвище)

УДК 621.396.1

РЕФЕРАТ

Сорока Я.О. Методи захисту корпоративних даних в банківських мережах: магістерська дис.: 171 Електроніка. Київ, КПІ ім. Ігоря Сікорського, 2020, 87 с.

Ключові слова: Structured Query Language, Transact-SQL, Universal Resource Locator, довірена обчислювальна база, ком'ютерна система, несанкціонований доступ, персональний комп'ютер, система управління базами даних, експертна система.

Актуальність роботи. В наш час найбільшу популярність набувають глобальні мережі Internet, в наслідок чого виникають проблеми з захистом інформації. Питання захисту інформації є невід’ємною частиною будь-якої системи, що працює з обробкою інформації, в тому числі корпоративних банківських мереж, які працюють з даними клієнтів. При збільшені використання інформації виникає проблема захищеності, то му її захист стоїть на першому місці.

Мета і завдання дослідження. Метою роботи є дослідження сучасних інформаційних загроз та методів захисту даних в корпоративних банківських мережах. Завданнями дослідження є:

– дослідити методи забезпечення інформаційної безпеки в інформаційних системах та їх особливості, пов'язані з корпоративними банківськими мережами;

– проаналізувати алгоритми забезпечення цілісності даних в корпоративних мережах;

– дослідити методи захисту даних та засоби їх реалізації;

– дослідити практичні аспекти реалізації методів захисту баз даних в корпоративних мережах.

Об’єкт дослідження – методи і засоби захисту та забезпечення безпеки інформації.

Предмет дослідження – створення баз даних та методи їх захисту.

Методи дослідження – критичний аналіз методів захисту інформації в базах даних; застосування мови SQL для реалізації алгоритмів захисту у базах даних.

Наукова новизна одержаних результатів. Запропоновано класифікацію загроз інформаційній безпеці в корпоративних мережах, сформульовано специфіку захисту баз даних, виходячи з наведеної класифікації загроз та удосконалено застосування алгоритмів захисту баз даних в корпоративних мережах.

Практичне значення одержаних результатів. Запропоновано сценарії захисту бази даних в корпоративних мережах та розроблено методи реалізації захисту БД. Розраховано параметри БД та проведено її моделювання в програмному середовищі M-SQL.

SUMMARY

Database management systems (DBMS), especially relational databases and expert systems (ES), have become the dominant tool in the field of data storage, processing and presentation. Any failure of the DBMS (ES), accompanied by loss of access to data, immediately affects the competitiveness of the enterprise. Therefore, the protection of data from unauthorized access, from unauthorized modification or simply from their destruction is one of the priorities in the design of any information system.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

CDI – Constrained Data Items (контрольовані дані)

DAC – Discretionary Access Control (виборчий контроль доступу)

DRI – Declarative Referential Integrity (декларативна посилальна цілісність)

IVP – Integrity Verification Procedure (процедура перевірки цілісності)

MAC – Mandatory Access Control (обов'язковий контроль доступу)

MS – Microsoft

RBAC – Role-Based Access Control (заснований на застосуванні ролей контроль доступу)

SID – Security Identifier (ідентифікаційний номер облікового запису)

SQL – Structured Query Language (структурована мова запитів)

TCB – Trusted Computing Base (довірена обчислювальна база)

TP – Transformation Procedure (процедура перетворення)

T-SQL – Transact-SQL

UDI – Unconstrained Data Items (неконтрольовані дані)

URL – Universal Resource Locator (універсальний локатор ресурсу)

БД – база даних

ДВБ – довірена обчислювальна база

КС – комп'ютерна система

НСД – несанкціонований доступ

ПК – персональний комп'ютер

СУБД – система управління базами даних

ЕС – експертна система

ВСТУП

У сучасних умовах діяльність будь-якої організації пов'язана з оперуванням великим об'ємом інформації, доступ до якої має широке коло осіб. В таких умовах зловмисні або просто некомпетентні дії всього лише одного із співробітників здатні завдати непоправної шкоди організації в цілому. Мова навіть може не йти про розкрадання цінної інформації, досить просто заблокувати доступ до важливого інформаційного ресурсу на досить тривалий час.

Системи управління базами даних (СУБД), особливо реляційні СУБД і експертні системи (ЕС), стали домінуючим інструментом в області зберігання, обробки і представлення даних. Будь-який збій в роботі СУБД (ЕС), що супроводжується втратою доступу до даних, негайно відбивається на конкурентоспроможності підприємства. Тому захист даних від несанкціонованого доступу, від несанкціонованої модифікації або просто від їх руйнування є одним із пріоритетних завдань при проектуванні будь-якої інформаційної системи. Ця проблема захисту даних охоплює як фізичний захист даних системних програм, так і захист від несанкціонованого доступу до даних, що передаються по лініях зв'язку і знаходиться на накопичувачах, що є результатом діяльності як сторонніх осіб, так і спеціальних програм-вірусів. Якщо взяти до уваги, що ядром інформаційної системи є СУБД, то забезпечення інформаційної безпеки останньої набуває вирішальне значення при виборі конкретних засобів забезпечення необхідного рівня безпеки організації в цілому.

1 МЕТОДИ ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ В ІНФОРМАЦІЙНИХ СИСТЕМАХ

Дані в комп'ютерній формі зосереджують у фізично локальному і невеликому обсязі (наприклад, на флеш-карті типу MicroSD) величезні масиви інформації, несанкціонований доступ до якої або її руйнування можуть призводити часом до катастрофічних наслідків і збитків. Можливість швидкого (і в окремих випадках навіть без слідів) копіювання, модифікації або видалення величезних масивів даних, що знаходяться в комп'ютерній формі додатково провокує зловмисників на несанкціонований доступ до інформації, її модифікацію або руйнування.

Теоретична проробка питань забезпечення безпеки інформації та їх практична реалізація довгий час відставали від рівня розвитку програмної індустрії СУБД, і в комерційних продуктах засоби забезпечення безпеки даних стали з'являтися лише в 90-х роках минулого століття.

Перші дослідження теорії і практики забезпечення безпеки даних в комп'ютерних системах були обумовлені, перш за все, потребами військової сфери, де проблема безпеки в цілому, і комп'ютерної безпеки зокрема, стоять особливо гостро. Початок цим процесам було покладено дослідженнями питань захисту комп’ютерної інформації, проведеними в кінці 70-х - початку 80-х років національним центром комп’ютерної безпеки (NCSC - National Computer Security Center) Міністерства оборони США. Результатом цих досліджень стало видання Міністерством оборони США в 1983 році документа під назвою «Критерії оцінки надійних комп’ютерних систем», згодом за кольором обкладинки отримав назву «Помаранчевої книги». Даний документ став фактично першим стандартом в області створення захищених комп’ютерних систем і згодом основою організації системи сертифікації комп’ютерних систем за критеріями захисту інформації. Підходи до побудови та аналізу захищених систем, представлені в «Помаранчевій книзі», послужили теоретичною та методологічною базою для подальших досліджень в цій сфері. У 1991 р NCSC був виданий новий документ-інтерпретація

«Критеріїв оцінки надійних комп'ютерних систем» в застосуванні до поняття надійної системи управління базою даних, відомий під скороченою назвою TDI або «Рожевої книги», що конкретизує і розвиває основні положення «Помаранчевої книги» з питань створення та оцінки захищених СУБД.

В сучасних БД і ЕС досить успішно вирішуються завдання захисту конфіденційних даних від несанкціонованого доступу, забезпечення цілісності та доступності даних. Забезпечення доступності даних на фізичному рівні досягається шляхом використання стійких до відмов пристроїв зберігання даних, наприклад, кількох жорстких дисків, об'єднаних в масив RAID. Періодичне створення резервних копій і зберігання результатів усіх операцій у файлі журналу дозволяє відновити дані практично після будь-якого збою. Цілісність даних забезпечується досить широким набором процедур, що забезпечує достовірність і несуперечливість даних. Сучасні СУБД забезпечують логічну цілісність і несуперечність даних вже на етапі опису моделі даних. Оператори контролю доступу входять в стандарт мови запитів SQL і реалізовані в багатьох СУБД. Але все ж, незважаючи на широкий діапазон засобів контролю і захисту, загальний рівень захищеності і СУБД і ЕС визначається можливостями операційної системи, оскільки ці два компоненти працюють в тісному зв'язку між собою.

1.1 Загальна характеристика складових інформаційної безпеки

Цінність інформації полягає в забезпеченні можливостей досягнення мети для користувача. Цінність інформації залежить від ступеня її корисності для власника. Інформацію будемо вважати достовірною, якщо вона точно відображає об'єкти і процеси навколишнього світу. Своєчасною буде інформація, якщо вона є цінною і достовірною відносно певного часового періоду.

Своєчасність відображає відповідність цінності і достовірності інформації певного часового періоду.

Під інформаційними ресурсами будемо розуміти всю накопичену інформацію про навколишню дійсність, зафіксовану на матеріальних носіях і в будь-якій інший формі, що забезпечує її передачу в часі і просторі між різними споживачами для вирішення наукових, виробничих, управлінських та інших завдань. Особливо слід виділити положення про те, що ресурсом є вся накопичена інформація, в тому числі і інформація недостовірна, представлена сумнівними фактами, помилковими положеннями, неефективними підходами, а також застаріла інформація і явна «дезінформація», що надійшла в інформаційні потоки. Це дійсно важливо, так як без виявлення недостовірної і застарілої інформації, що накопичується в інформаційних ресурсах, створюються передумови прийняття неефективних, а в ряді випадків і помилкових рішень, що завдають істотної шкоди.

Базам даних і системам, в яких зберігається інформація, притаманні такі властивості: конфіденційність (потрібен захист від несанкціонованого доступу), цілісність (потрібен захист від втрати узгодженості та достовірності даних) і доступність (потрібен захист від несанкціонованого утримання інформації і ресурсів, підтримка працездатності та захист від руйнування), тобто під захистом даних розуміють систему заходів, що оберігають дані від несанкціонованого використання, руйнування або спотворення. В англійській мові використовуються такі терміни: secrecy, integrity, and availability of data. Загальна ідея захисту бази даних складається в дотриманні рекомендацій, сформульованих для класу безпеки С2 в «Критеріях оцінки надійних комп'ютерних систем»[1].

Оскільки організація системи захисту даних в базах даних та експертних системах не має принципових відмінностей, то далі для стислості викладу будемо посилатися тільки на бази даних і системи їх управління.

Вважають, що практично не можна реалізувати абсолютний захист даних, тому зазвичай задовольняються відносним захистом інформації -

гарантовано захищають її на той період часу, поки несанкціонований доступ до неї, втрата її цілісності або доступності тягне будь-які наслідки.

Перерахуємо типові ситуації, в яких дані, що зберігаються, можуть бути викрадені, спотворені або загублені.

Несанкціонований доступ (НСД). При отриманні доступу до даних, що зберігаються, будь-який користувач може змінити їх, зруйнувати або нашкодити власнику цієї інформації.

Некерований паралелізм. Якщо декілька користувачів, які одночасно працюють з базою даних, намагаються змінювати її стан, то в результаті можемо отримати неузгодженість.

Локальний збій. Під час виконання програми може виникнути аварійна ситуація, тоді виконання програми буде припинено. Під час оновлення даних може виявитися неузгоджений стан бази даних.

Втрата оперативної пам'яті (м'який збій). Під час відключення живлення, може бути системний збій, в результаті чого може втратити вміст системних буферів і буферів додатків, які розміщені в оперативній пам’яті. Під час м’якого збою стан БД може виявитися неузгодженим і навіть після перезавантаження системи, в результаті можуть постраждати дані всіх користувачів.

Фізичне руйнування даних (жорсткий збій). Під час експлуатації пристрою зовнішньої пам’яті може бути зіпсована, наприклад, поверхня диска, блок головок дисковода. В результаті жорсткого збою ймовірність неузгодженого стану БД мала, але, потрібно це передбачати

Таким чином, ніякі руйнування даних, чи то випадкові, чи навмисні, не можуть бути незворотними, можуть бути наслідком

1)дій некомпетентних або злочинних користувачів;

2)неузгоджених оновлень, які одночасно відбувались у різних користувачів;

3) помилки в програмному забезпеченні, яка не обробилась належним чином;

4)застосування в програмному забезпеченні шкідливого коду;

5)збою або відмови різних пристроїв системи, аварійного відключення енергії, порушення ліній зв’язку;

6) фізичного знищення комп'ютерної системи або її частин.

Отже, щоб захистити дані від несанкціонованого використання, втрати цілісності та руйнування, система повинна забезпечуватись:

1) інформаційною підтримкою установи щодо обмежень прав доступу користувачів до даних;

2) захистом від шкідливих дій користувачів;

3) дисципліною паралельної роботи багатьох користувачів, що виключає можливість внесення неузгоджених змін до стану БД;

4)відновленням цілісного стану БД після локальних і м'яких збоїв системи;

5) відновленням БД після жорстких збоїв.

Щоб забезпечити безпеку даних користуються комп’ютерними і некомп’ютерними засоби захисту. Розглянемо комп’ютерні засоби захисту.

Авторизація користувачів і розмежування доступу. Під авторизацією будемо розуміти певні права, які надаються користувачу на законний доступ до всієї системи чи її об’єктів та здійснюється як засобами операційної системи, так і засобами самої СУБД.

Доступ до даних через подання. Подання є результатом однієї або декількох реляційних операцій, якого реально не існує в БД, але створюється на вимогу окремого користувача в момент надходження цієї вимоги.

Механізм подання є потужним і гнучким інструментом організації захисту даних, що дає змогу приховати частини БД від користувачів, в результаті чого буде недоступна інформація про існування будь-яких атрибутів або рядків даних. Використання поданнямає жорсткіший механізм контролю доступу, ніж звичайне надання користувачеві тих чи інших прав доступу до бази даних.

Реалізація правил «бізнес логіки» за допомогою збережених процедур, функцій і тригерів надає значно вищий рівень захисту даних, ніж їх реалізація в клієнтських програмах. Збережені процедури, функції і тригери є фрагментами коду мови програмування високого рівня, яка є розширенням мови SQL, що зберігається і виконується на сервері.

Створення резервних копій та відновлення. Під резервним копіюванням будемо розуміти процедуру, що періодично виконується для отримання копії БД і її файлу журналу (а також, можливо, програм СУБД) у зовнішній постійній пам'яті, найчастіше на окремому носії. Якщо БД стає непридатною для подальшої експлуатації, то резервну копію і зафіксовану в файлі журналу оперативну інформацію використовують для відновлення БД до останнього узгодженого стану.

Вести журнал – означає створити і обслуговувати файл журналу, що містить відомості про всі зміни, внесені в БД з моменту створення останньої резервної копії, який призначений для забезпечення ефективного відновлення системи в разі її відмови. Якщо відмовила в системі функція ведення системного журналу, базу даних можна відновити лише до того стану, який був зафіксований в останній створеній резервній копії. Всі зміни, які були внесені в БД після створення останньої резервної копії, будуть втраченими.

Забезпечення цілісності включає перевірку цілісності даних і їх відновлення в разі виявлення суперечностей в БД. Цілісний стан БД описується за допомогою обмежувачів цілісності у вигляді умов, яким повинні задовольняти збережені в базі дані.

Управління паралелізмом виконання транзакцій є одним із найважливіших завдань СУБД, що дозволяє безлічі користувачів мати одночасний доступ до даних.

Щоб визначити активність використання БД, аналізують файли журналу, які використовуються також для виявлення будь-яких незвичайних дій в системі. Щоб своєчасно виявити і припинити будь-які спроби порушення

захисту, потрібно регулярно проводити аудиторські перевірки та постійно контролювати вміст файлів журналу.

Надійно захистити конфіденційні дані можна лише за допомогою шифрування даних. Під шифруванням будемо розуміти кодування даних за допомогою спеціального алгоритму, в результаті чого вони стають недоступними для читання. Щоб успішно дешифрувати дані потрібно знати ключ дешифрування. У деяких СУБД є вбудовані засоби шифрування даних, які дешифрують «на льоту» в момент звернення користувача до даних.

При формулюванні основних понять і визначень будемо вважати, що захищається не сама БД або експертна система, а комп'ютерна система (КС) в цілому. Поняття захисту можна застосувати не тільки до його запису даних. Проломи в системі захисту можуть виникати і в інших частинах системи, наприклад, ліній зв'язку, що в свою чергу, наражає на небезпеку і, власне, БД. Отже, захист БД повинен охоплювати використовуване обладнання, програмне забезпечення, персонал і власне дані.

Інформаційна безпека (information security) – стан, що розглядається КС, при якій вона, з одного боку, здатна протистояти дестабілізуючому впливу зовнішніх і внутрішніх інформаційних загроз, а з іншого – функціонування системи і сам факт її наявності не створюють загроз для її користувачів, для зовнішнього середовища і для елементів самої КС. Інакше, безпека інформації – стан захищеності інформації, при якому забезпечені її конфіденційність, цілісність і доступність[2].

Захист інформації – комплекс заходів, спрямованих на забезпечення інформаційної безпеки.

Захищена інформація (sensitive information) – інформація, яка є предметом власності і підлягає захисту відповідно до вимог правових документів або вимог, встановлених власником інформації. Дуже часто поряд з терміном захищеної інформації вживається і термін конфіденційна інформація – інформація, доступ до якої обмежується відповідно до законодавства або вимог власника. Конфіденційність (confidentiality або

secrecy) - це властивість інформації бути відомою тільки допущеним і, які пройшли перевірку, суб'єктам системи.

Захищена інформаційна система - система, призначена для обробки інформації, що захищається з необхідним рівнем її захищеності.

Безпечна система надає доступ до інформації тільки авторизованим особам, в якій використовуються відповідні апаратні і програмні засоби. Абсолютно безпечних систем немає, і тут мова йде про надійну систему в сенсі «система, якій можна довіряти».

Основними критеріями оцінки надійності є політика безпеки і гарантованість.

Політикою безпеки (security policy) називають якісний опис комплексу організаційно-технологічних та програмно-технічних заходів щодо забезпечення захисту даних в КС, включає в себе аналіз можливих загроз і вибір відповідних заходів протидії.

Гарантованість визначає ступінь довіри, який можна визначити тестуванням системи в цілому і окремих її компонентів. При оцінці ступеня гарантованості, з якої систему можна вважати надійною, центральне місце займає довірена (надійна) обчислювальна база (ДВБ) (в англійському - trusted computing base, TCB).

Формальний (математичний, алгоритмічний, схемотехнічний) вираз і формулювання політики безпеки називають моделлю безпеки (security model). В основі більшості моделей безпеки лежить суб'єктно-об'єктна модель комп'ютерних систем, в тому числі і баз даних як ядра автоматизованих інформаційних систем. Виділяються суб'єкти (subjects) бази даних (активні сутності), об'єкти (objects) бази даних (пасивні сутності) і породжувані діями суб'єктів процеси над об'єктами.

Таким чином, суб'єктом доступу (access subject) називається активний компонент системи, який може стати причиною ініціалізації потоку інформації або зміни стану системи. Суб'єктом є особа або процес, дії якого регламентуються правилами розмежування доступу.

Об’єкт доступу (access object) - пасивний компонент системи, який зберігає, приймає або передає інформацію, доступ до якої регламентується правилами розмежування доступу. Об’єктами доступу в БД є практично все, що містить кінцеву інформацію: таблиці (базові або віртуальні), уявлення, а також більш дрібні елементи даних: стовпці і рядки таблиць, і навіть окремі поля.

Правила розмежування доступу (security policy) – сукупність правил, що регламентують права суб’єктів доступу до об’єктів доступу. Доступ до інформації поділяють на санкціонований і несанкціонований: санкціонований доступ (authorized access to information) – доступ до інформації, яка не порушує встановлених правил розмежування доступу, а несанкціонований доступ (unauthorized access to information) – доступ до інформації, який порушує встановлені правила розмежування. Контроль доступу обов’язково включає ідентифікацію (персоналізацію) і аутентифікацію всіх суб’єктів і їх процесів та розмежування повноважень суб’єктів по відношенню до об’єктів з подальшою обов’язковою перевіркою введених повноважень.

Ідентифікація (identification) – присвоєння об’єктам і суб’єктам доступу ідентифікатора і (або) порівняння висунутого ідентифікатора з переліком привласнених ідентифікаторів. Ідентифікатор доступу (access identifier) – унікальна ознака об’єкта або суб’єкта доступу.

Аутентифікація (authentication) – перевірка належності суб’єкту доступу пред’явленого ним ідентифікатора, підтвердження справжності. Отже, завдання ідентифікації – відповісти на питання «хто це?», А завдання аутентифікації - «а він це насправді?».

Після ідентифікації і перевірки автентичності встановлюється сфера дій суб’єкта (доступні йому ресурси КС). Цю процедуру називають наданням повноважень (авторизацією).

Під авторизацією (authorization) будемо розуміти права власника на законний доступ до БД або її об’єктів. Іноді процес авторизації включає і ідентифікацію, і аутентифікацію суб’єктів, які потребують отримання доступу

до об’єктів. Привілей доступу (рівень повноважень суб'єкта доступу) (subject privilege) –сукупність прав доступу суб'єкта.

Вищенаведені поняття більшою мірою пов’язані з організацією захисту конфіденційних даних. Поряд із захистом конфіденційних даних в поняття безпеки даних входить також забезпечення цілісності та доступності даних.

Цілісність даних (data integrity) – властивість даних зберігати точність і несуперечність незалежно від внесених змін. Залежить від здатності інформаційної системи забезпечити незмінність даних (дані, що зберігаються в системі, не відрізняються в семантичному відношенні від даних у вихідних документах) в умовах випадкового і (або) навмисного спотворення (руйнування). Якщо бути більш точним, то суб’єктів цікавить забезпечення більш широкого властивості – достовірності даних (інформації), яка складається з адекватності (повноти і точності) відображення стану предметної області і безпосередньо цілісності даних.

Доступність даних (data availability) – можливість реалізації безперешкодного доступу до інформації суб’єктів, що мають на це відповідні повноваження. Забезпечується здатністю системи забезпечувати своєчасний безперешкодний доступ суб’єктів до запитування даних.

1.2 Класифікація загроз інформаційної безпеки

Під загрозою безпеки інформації будемо розуміти вплив на комп’ютерну систему, яка прямо або побічно може завдати шкоди безпеці інформації. Загрози реалізують або намагаються реалізувати зловмисники інформаційної безпеки.

Зловмисник інформаційної безпеки (security policy violator) – суб’єкт доступу, який випадково або навмисно зробив дію, наслідком якої є порушення інформаційної безпеки організації. Формалізований опис або

подання комплексу можливостей порушника по реалізації тих чи інших загроз безпеці інформації називають моделлю зловмисника.

Уразливість – властивість інформаційної системи, яка обумовлює можливість виникнення на будь-якому етапі життєвого циклу КС такого її стану, при якому створюються умови для реалізації загроз безпеці інформації.

Атакою на КС називають дії, що робляться зловмисником, які полягають в пошуку і використанні тієї або іншої уразливості. Інакше кажучи, атака на КС є реалізацією загрози безпеки інформації в ній.

Класифікація загроз може бути проведена по безлічі ознак. На найвищому рівні загрози інформаційної безпеки можуть бути класифіковані за природою виникнення. Всі загрози інформаційній безпеці можна розділити на загрози, які не залежать від діяльності людини (природні загрози), наприклад стихійні природні явища, і загрози, спричинені людською діяльністю (штучні загрози). Останні є найбільш небезпечними загрозам і їм приділяється підвищена увага.

Штучні загрози, виходячи з їх мотивів, поділяються на ненавмисні (випадкові) і навмисні (навмисні) (класифікація за ступенем навмисності). До ненавмисних загроз відносяться:

- помилки в проектуванні КС;
- помилки в розробці програмних засобів КС;
- випадкові збої в роботі апаратних засобів КС, ліній зв'язку, енергопостачання;
- помилки користувачів КС (некомпетентне використання засобів захисту, а також будь-яких інших програмних і апаратних засобів системи);
- вплив на апаратні засоби КС фізичних полів інших електронних пристроїв та ін.

1.3 Специфіка захисту в базах даних

Будуючи захист бази даних слід враховувати специфічні загрози безпеці інформації:

- інференції і агрегування (Logical Inference and Aggregation);
- комбінація дозволених запитів для отримання закритих даних (Browsing);
- організація прихованих каналів передачі інформації (Covert Channels);
- SQL ін'єкції (SQL Injection);
- програмні закладки, оцінний код (Backdoors, Trapdoors)
- троянські коні (Trojan Horses).

Неформально під прихованим каналом (Covert channel) передачі інформації розуміють будь-який канал зв'язку, не призначений спочатку для передачі інформації. У більш загальному вигляді під прихованим, інакше непрямим каналом, порушенням конфіденційності мається на увазі механізм, за допомогою якого суб’єкт, який має високий рівень допуску, може надати певні аспекти конфіденційної інформації суб’єктам, ступінь допуску яких нижче рівня конфіденційності цієї інформації.

Для організації прихованого каналу необхідні три речі. По-перше, відправник і одержувач повинні мати доступ до загального ресурсу. По-друге, відправник повинен мати можливість змінювати деякі властивості загального ресурсу, а одержувач - мати доступ на перегляд загального ресурсу. Нарешті, відправник і одержувач повинні мати можливість синхронізувати свої дії.

Приховані канали практично неможливо усунути, і наші зусилля повинні бути спрямовані на мінімізацію пропускної здатності цих каналів. Напевно, єдиний спосіб усунення прихованих каналів полягає в повній ліквідації всіх загальних ресурсів і всіх комунікацій[3].

Виділяють такі типи прихованих каналів:

1)приховані канали по пам’яті, в яких інформація передається через доступ відправника на запис і одержувача на читання до одних і тих же ресурсів або об'єктів;

2) приховані канали по часу, які характеризуються доступом відправника і одержувача до одного і того ж процесу або змінюваному в часі атрибуту.

Генерація помилок. Перевірка уразливості для SQL ін’єкцій шляхом введення '*** в поле введення. Одиничні лапки вставляють також в рядок URL запиту. Виникає необроблена помилка, яка може надати багато інформації для зловмисника. Якщо він отримає помилку виду

Microsoft OLE DB Provider for ODBC Drivers error '80040e14'

[Microsoft][ODBC SQL Server Driver][SQL Server]Incorrect syntax near the keyword 'or'.

/wasc.asp, line 69

Певним чином сформульовані запити можуть дати доступ до структури даних (імен полів і таблиць). Наприклад, для пошуку уразливості на сайті, написаному на мові PHP, який використовує СУБД MySQL, можна спробувати наступне:

`http://.../php?id = 12 + union + select + null,null,null + from + users/*`(Тут / * це символ коментаря в MySQL). Правильно виконані запити будуть відповідати існуючим іменам таблиць. Слід перевірити на існування таблиць users, passwords, regusers і т.д. Потім виконуємо:

`http://.../php?id=9999+union+select+'test',null,null/*`

`http://.../php?id=9999+union+select+null,'test',null/*`

Введення подібних команд необхідно продовжувати до тих пір, доки назва поля (слово test) з’явиться в потрібному місці. «Корисними» є також наступні команди:

`http://.../php?id=9999+union+select+null,DATABASE(),null/*`

```
http://.../php?id=9999+union+select+null,USER(),null/*
http://.../php?id=9999+union+select+null,VERSION(),null
/*
```

Можна також спробувати і вразливість, що дозволяє завантажувати файли з сервера:

```
http://.../php?id=9999+union+select+null,LOAD_FILE(
'/etc/p asswd'),null/*
```

Якщонемаєможливостізастосування union взапиті (наприклад, MySQL маєверсію3. *), то ін'єкцію можна використовувати для того, щоб змусити сервер бази даних вичерпати всі свої ресурси. Для цього можна використовувати функцію BENCHMARK, яка повторює виконання виразу expr задану кількість разів, вказану в аргументі count. Складений запит буде виглядати наступним чином:

```
http://.../php?id=BENCHMARK(10000000,BENCHMARK(1000
0000,md 5(current_date)))
```

1000000 запитів md5 виконуються (в залежності від потужності сервера), наближено 5 секунд. Вкладений benchmark буде виконуватися дуже довго на будь-якому сервері.

1.4 Управління доступом до даних

У будь-якій організації діють певні правила накопичення і використання відомостей, що обмежують доступ до інформаційних ресурсів. Конкретні правила визначаються інформаційною політикою керівництва підприємства, однак в будь-якому випадку розумні обмеження доступу засновані на наступних принципах:

- підприємство є власником всієї службової інформації, отриманої його підрозділами або службовцями;

- підрозділ або службовець є власником отриманої ним інформації і може використовувати її в інтересах підприємства без обмежень;
- службовець має право доступу до тих і тільки тих відомостей, які необхідні для виконання його службових обов’язків;
- службовець має право виконувати ті, і тільки ті маніпуляції доступними відомостями, які обумовлені його посадовими обов'язками.

Ці принципи реалізуються у вигляді системи правил, що обмежують права доступу співробітників до інформаційних ресурсів підприємства. Сучасні СУБД мають добре розвинені засоби підтримки подібних правил - підсистеми адміністрування даних. Метою підсистеми адміністрування є забезпечення санкціонованого доступу службовців підприємства до збережених даних. З концептуальної точки зору вона повинна забезпечувати наділення користувачів СУБД привілеями по відношенню до даних і контролювати надання конкретного користувача тільки певних для нього привілеїв.

Доступ до комп’ютерної системи користувачу надає системний адміністратор, якийстворює облікові записи. Кожен користувач має унікальний ідентифікатор, з яким пов’язується пароль, обраний користувачем при реєстраціїі відомий операційній системі. Таким чином організовується контрольований доступ до комп’ютерної системи, але це не означає, що користувач може зайти до СУБД чи іншої прикладної програми. Щоб отримати доступ до СУБД, адміністратор повинен створити індивідуальні ідентифікаторивже в середовищі самої СУБД, які також зв’язується з паролем, який повинен бути відомий тільки даному користувачу.

0,Виходячи з вищесказаного, сформулюємо термін «управління доступом» більш детально. Управління доступом є метод захисту інформації шляхом регулювання використання ресурсів системи (елементів БД, програмних і технічних засобів). Включає наступні функції захисту:

- ідентифікація користувачів і ресурсів системи;

- встановлення автентичності об’єкта або суб’єкта по пред'явленому ним ідентифікатора (аутентифікація);
- розмежування і перевірка повноважень (авторизація). Створення умов роботи в межах встановленого регламенту;
- реєстрація звернень до ресурсів, що захищаються (протоколювання і аудит);
- реагування при спробах несанкціонованого доступу.

1.5 Авторизація користувачів

Авторизація користувачів (суб'єктів доступу) полягає в наданні певних прав (або привілеїв), що дозволяють їх власнику мати законний доступ як до самої системи, так і до її окремих об’єктів. Всі суб’єкти доступу можуть бути розділені для системи на ряд категорій, наприклад: CONNECT, RESOURCE і DBA. Набір таких категорій визначається виробником СУБД. Наростання можливостей (повноважень) для кожного окремого виду підключення відбувається в зазначеному порядку:

CONNECT - кінцеві користувачі. За замовчуванням їм дозволено тільки з’єднання з базою даних і виконання запитів до даних, всі їхні дії регламентовані виданими їм привілеями;

RESOURCE - привілейовані користувачі, які мають право створення власних об’єктів в базі даних (таблиць, уявлень, збережених процедур і тригерів).

DBA - категорія адміністраторів бази даних. Включає можливості обох попередніх категорій, а також можливість реєструвати суб’єкти захисту або змінювати їх категорію.

Адміністратор кожного БД займається створенням списку можливих користувачів БД і розмежуванням повноважень цих користувачів.

Дані про розмежуваннях розташовуються в системному каталозі БД. Очевидно, що дана інформація може бути використана для несанкціонованого доступу і тому також підлягає захисту. Захист цих даних здійснюється засобами самої СУБД.

Висновки до розділу

1. В сучасних БД і ЕС досить успішно вирішуються завдання захисту конфіденційних даних від несанкціонованого доступу, забезпечення цілісності та доступності даних. Забезпечення доступності даних на фізичному рівні досягається шляхом використання стійких до відмов пристроїв зберігання даних, наприклад, кількох жорстких дисків, об’єднаних в масив RAID.
2. Створюючи захист баз даних необхідно враховувати, що специфічні загрози безпеці інформації, пов’язані з концентрацією великої кількості різноманітної інформації в базах даних, а також з можливістю використання складних запитів обробки даних.
3. У будь-якій організації діють певні правила накопичення і використання відомостей, що обмежують доступ до інформаційних ресурсів.

2 ЗАБЕЗПЕЧЕННЯ ЦІЛІСНОСТІ ДАНИХ

Під цілісністю даних розуміють відповідність інформаційної моделі предметній області, тобто даних, що зберігаються в базі даних, об’єктам реального світу і їх взаємозв’язкам в кожен момент часу. Будь-яка зміна в предметній області, значуща для побудованої моделі, має відображатися в базі даних, і при цьому повинна зберігатися однозначна інтерпретація інформаційної моделі в термінах предметної області. Цілісність БД не гарантує достовірності, що міститься в ній інформації, але забезпечує принаймні її правдоподібність, відкидаючи свідомо неймовірні, неможливі значення. Таким чином, не слід плутати цілісність БД з достовірністю даних. Достовірність (або істинність) є відповідність фактів, що зберігаються в БД, реального світу. Контроль цілісності даних – це здатність СУБД або КС в цілому забезпечити незмінність даних (дані, що зберігаються в системі, не відрізняються в семантичному відношенні від даних у вихідних документах) в умовах випадкового і (або) навмисного спотворення (руйнування) або, інакше, під цілісністю даних маємо на увазі відсутність неналежних змін.

2.1 Принципи забезпечення цілісності даних

Поняття «неналежна зміна» введено Д. Кларком і Д. Вільсоном: жодному користувачеві КС, в тому числі і авторизованому, не повинні бути дозволені такі зміни даних, які спричинять за собою їх руйнування або втрату. У роботах Кларка і Вільсона визначені дев’ять абстрактних теоретичних принципів, виконання яких дозволить забезпечити цілісність даних:

- коректність транзакцій;
- авторизація користувачів;
- мінімізація привілеїв;

- розмежування функціональних обов'язків;
- аудит подій, що відбулися;
- об'єктивний контроль;
- управління передачею привілеїв;
- ефективне застосування механізмів захисту;
- простота використання захисних механізмів.

2.2 Модель Кларка-Вільсона

Модель цілісності Кларка-Вільсона була впроваджена в 1987 р на основі результатів аналізу практики паперового документообігу в комерційних компаніях, ефективної з точки зору забезпечення цілісності інформації. Модель Кларка-Вільсона є описовою і не тримає яких би то ні було строгих математичних конструкцій – скоріше її доцільно розглядати як сукупність практичних рекомендацій з побудови системи забезпечення цілісності в КС. Основні поняття даної моделі - це коректність транзакцій і розмежування функціональних обов’язків. Модель задає правила функціонування КС і визначає дві категорії даних і два класи операцій над ними. Все що містилося в системі: дані поділяються на контрольовані і на неконтрольовані елементи даних[4]. Цілісність перших підтримується, а цілісність других – ніяк не контролюється. Введемо наступні позначення:

- S - безліч суб'єктів;

- D - безліч даних в автоматизованій системі (безліч об'єктів);

- CDI (Constrained Data Items) - дані, цілісність яких контролюється;

- UDI (Unconstrained Data Items) - дані, цілісність яких не контролюється;

При цьому D = CDI UDI, CDI UDI = Ø.

- TP (Transformation Procedure) - процедура перетворення, тобто компонент, який може ініціювати транзакцію - послідовність операцій, що переводять систему з одного стану в інший;

- IVP (Integrity Verification Procedure) – процедура перевірки цілісності CDI.

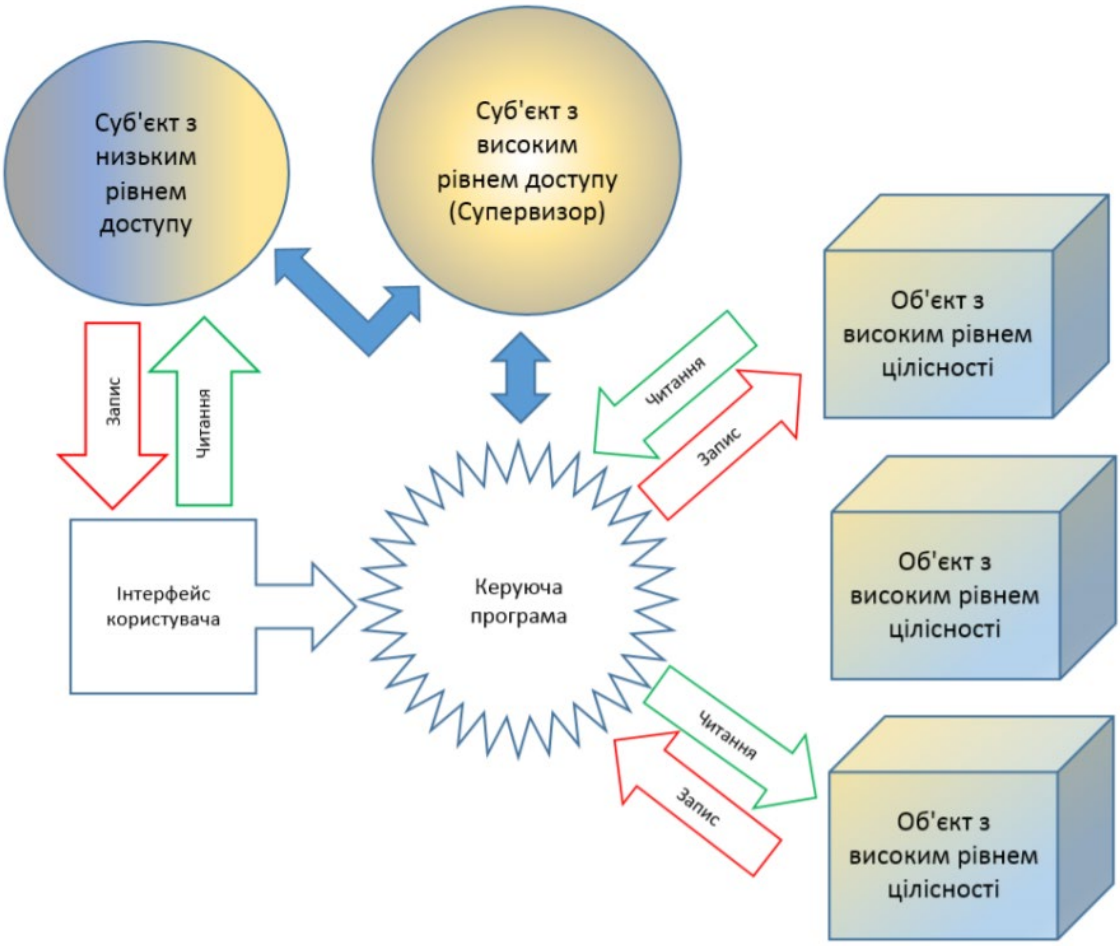

Рисунок 1.1 – Модель цілісності Кларка-Вільсона

Правила моделі Кларка-Вільсона:

1) у системі повинні бути IVP, здатні підтвердити цілісність будь-якого CDI. Наприклад, IVP може бути механізмом підрахунку контрольних сум;

2) застосування будь-якої TP до будь-якого CDI має зберігати цілісність цього CDI;

3) тільки TP можуть вносити зміни в CDI;

4) суб'єкти можуть ініціювати тільки певні TP над певними CDI (дана вимога означає, що система повинна підтримувати відносини виду (s, t, d), де s S, t TP, d CDI. Якщо відношення визначено, то суб'єкт s може приманити перетворення t до об'єкта d);

5)повинна бути забезпечена політика поділу обов'язків суб'єктів –тобто суб'єкти не повинні змінювати CDI без залучення в операцію інших суб'єктів системи;

6)спеціальні ТР можуть перетворювати UDI в CDI;

7) кожне застосування ТР має реєструватися в спеціальному CDI.

Безумовними перевагами моделі Кларка-Вільсона є її простота і легкість спільного використання з іншими моделями безпеки.

Оператори мови SQL:

1) обмежувачі значень (domain constraints);

2) обмежувачі ключів (key constraints);

3) обмежувачі записів (entity constraints);

4) посилальна цілісність (Declarative Referential Integrity, DRI).

Обмежувачі ключів задаються за допомогою заборони невизначених Null значень для ключових полів (по невизначеному значенню неможливо провести ідентифікацію запису) і вимогу унікальності значення ключового поля. При порівнянні невизначених значень не діють стандартні правила порівняння: одне невизначене значення ніколи не вважається рівним іншому невизначеному значенню. У SQL у фразі WHERE для виявлення рівності значення деякого атрибута невизначеному значенню застосовують спеціальні предикати:

```
<ім'я атрибута> IS NULL и <ім'я атрибута> IS NOT NULL
```

Унікальність значень будь-якого поля задається за допомогою оператора UNIQUE. Для оголошення унікальності сукупності полів застосовується конструкція виду:

```
UNIQUE (<поле1>, <поле2>, …)
```

Для оголошення первинного ключа використовується оператор PRIMARY KEY. Додатково, за допомогою специфічних для кожної СУБД операторів, може встановлюватися автономерація ключового поля.

Обмежувачі запису також задаються за допомогою заборони невизначених значень, але вже не для ключових, а важливих в семантичному сенсі полів. Тоді принаймні мають хоча б одне обов'язкове до заповнення поле, не буде повністю порожніх кортежів.

Посилальна цілісність забезпечує підтримку несуперечливого стану БД (узгодженого стану зовнішніх ключів) в процесі модифікації даних, а також при виконанні операцій додавання або видалення записів. У SQL для цього вводиться оператор FOREIGN KEY:

```
FOREIGNKEY(<список полів>) REFERENCES<імя таблиці>(<список полів>) ONDELETECASCADEONUPDATECASCADE. Например: FOREIGN KEY(client_id) REFERENCES clients(id) ON DELETE CASCADE
```

Обмежувачі ключів, записи і довідкова цілісність визначають правила роботи СУБД з реляційними структурами даних. Але з іншого боку, ці аспекти ніяк не стосуються змісту бази даних. Для визначення деяких обмежень, які пов'язані з утриманням бази даних, вводяться обмежувачі значень. Обмежувачі значень задаються шляхом визначення типу поля (домену), завдання умови на значення, завдання варіанти вибору і визначення значення за замовчуванням. Значення за замовчуванням задається за допомогою оператора

```
DEFAULT(<значення або вираз>) {FOR <імя поля>},
```

а обмежувачі значень вводяться за допомогою оператора CHECK (<умова>). Усередині CHECK можуть використовуватися оператори порівняння, функції IN, BETWEEN, LIKE і інші функції SQL. наприклад:

```
DEFAULT(GetDate()) for date
CHECK(price between 0 and 100000)
CHECK(passport like 'MP%20')
CHECK(місто in('Брюсель','Берлін','Київ'))   або
CHECK(місто in(select capital from capitals))
```

2.3 Відновлення цілісного стану БД

Цілісний стан БД можна відновити за рахунок накопичення надлишкової інформації. Для цього потрібно:

- періодично створювати резервні копії бази даних;

- інформацію про транзакції, виконані в проміжку часу між двома послідовними копіюваннями, зберігається в спеціальному файлі-журналі змін бази даних;

-створити контрольні точки, необхідні для синхронізації процесу відновлення і зменшення обсягу пошуку в файлі журналу.

Оскільки транзакція є ключовим поняттям при організації відновлення цілісного стану БД, спочатку коротко розглянемо основні властивості транзакцій.

2.4 Поняття транзакції

Під транзакцією розуміється виконувана послідовність операцій маніпулювання даними від імені певного користувача або процесу, яка переводить БД з цілісного початкового стану в цілісний кінцевий стан. Слід звернути увагу, що проміжні стани БД не повинні бути обов'язково цілісними. Транзакції притаманні такі властивості, відомі як вимоги ACID (Atomicity, Consistency, Isolation і Durability):

Атомарність і узгодженість. Транзакція як одиниця роботи з даними володіє властивістю атомарності і узгодженості, коли цілісний початковий стан БД перетворюється в цілісний кінцевий стан. Транзакція не може бути виконана частково. Якщо в процесі обробки транзакції виникає якийсь збій, то всі поновлення даних, які виконалось до цього моменту скасовуються.

Довговічність. Транзакція має властивість довговічності: оброблені дані, які успішно завершилася транзакцією, стають постійними і не можуть загубитись або скасуватись ні за яких обставин. Вважають транзакцію

успішною, якщо під час її виконання не виникла ніяка аварійна ситуація, а всі перевірки обмежень цілісності дали позитивний результат. Таким чином всі вироблені поновлення даних зберігаються (фіксуються) в постійній пам'яті. Система гарантує фіксацію оновлень не тільки тоді, коли успішно завершилися транзакції, а навіть в тому випадку, якщо відбудеться системний збій. Якщо транзакції завершилися неуспішно, то система гарантує відсутність будь-яких слідів її роботи у зовнішній пам'яті.

Система, як правило, розраховується на багатьох користувачів одночасно, тому може підтримувати їх сеанси роботи одночасно. Якщо в один і той же момент часу багато користувачів намагатимуться отримати доступ до одних і тих даних, то виникне проблема управління одночасним доступом до БД. Тому потрібно віднайти таку стратегію управління, яка забезпечила б взаємну ізоляцію користувачів, а результати роботи будь-якої транзакції кожного з них виконувались паралельно. Транзакції називається ізольованими, коли вони виконуються паралельно і кожній із них недоступні проміжні результати інших.

Таким чином, поняття транзакції можна сформулювати як виконувану від імені одного ідентифікатора суб'єкта неподільну послідовність операцій над даними, що володіє властивостями атомарності, узгодженості, ізольованості і довговічності. У стандарті SQL визначені наступні оператори управління транзакціями:

BEGIN TRANSACTION - повідомляє диспетчеру транзакцій про явне початку нової транзакції (неявно будь-яка інструкція поновлення даних запускається в рамках транзакції);

COMMIT TRANSACTION - повідомляє диспетчеру транзакцій про успішне завершення транзакції і вимогу фіксації результатів у зовнішній пам'яті;

ROLLBACK TRANSACTION - повідомляє диспетчеру транзакцій про виникнення помилки і вимогу відкату проведених змін.

Система повинна бути узгодженою. Для цього потрібно зробити перевірку обмежень цілісності, які повинні відповідати рівню окремих інструкцій оновлення. При їх порушеннях відкат транзакції не потрібен[5].

2.5 Принципи відновлення даних

В основі відновлення даних лежать наступні два принципи:

1. Результати зафіксованих транзакцій повинні бути збережені у відновленому стані бази даних;

2. Результати незафіксованих транзакцій повинні бути відсутніми у відновленому стані бази даних.

Це, власне, і означає, що відновлюється останнім за часом узгоджений стан бази даних. Процедури відновлення залежать від типу збою.

База даних може виявитися в неузгоджену стані під час локального, м'якого та жорсткого збоїв.

Під локальним збоєм будемо розуміти аварійне призупинення транзакції. Це може статися під час спроби поділу на нуль або порушення обмежень цілісності, або завершення транзакції оператором ROLLBACK. Якщо в цей час транзакція виконувала оновлення даних, то стан БД виявиться неузгодженим. Щоб відновити цілісність даних у з перерваною транзакцією, потрібно провести індивідуальний відкат транзакції.

Внаслідок аварійного відключення живлення або під час збою процесора відбувається м’який збій системи, в результаті чого втрачається вміст оперативної пам'яті, аварійно перериваються всі існуючі транзакції, не фіксуються в БД результати транзакцій, що завершилися оператором COMMIT. Після перезавантаження системи повинен бути виконаний відкат всіх транзакцій, що не завершилися до моменту збою. Якщо виявиться, що транзакції проведені, але незафіксовані, то потрібно автоматично виконати їх повторно.

Фізичне руйнування бази даних називається жорстким збоєм, наслідки якого є катастрофічними. Навіть у такому стані система базу даних повинна відновитись.

Для відновлення узгодженого стану БД потрібно мати інформацію, яка обробляється транзакціями, зчитується системою в робочі буфери бази даних в оперативній пам'яті. Транзакції всі даних оновлюються у буферах, робочий стан яких відображає поточний стан тієї частини бази даних, яка доступна для діючих транзакцій.

Організація відновлення даних в СУБД MS SQL Server.

SQL Server пропонує на вибір три моделі відновлення, в основному відрізняються використанням журналу транзакцій, і п'ять варіантів резервного копіювання. Моделі відновлення наступні:

1. Проста модель. Журнал транзакцій резервується (не створюється його резервна копія у зовнішній пам'яті);

2. Модель з неповним протоколюванням. Масові операції не записується в журнал транзакцій. Журнал транзакцій резервується;

3. Повна модель. Усі транзакції заносяться в журнал. Журнал транзакцій резервується.

Резервне копіювання можливо наступних видів:

Повне. Резервуються всі дані;

Диференційоване. Проводиться резервування всіх сторінок даних, змінених з моменту останнього повного резервного копіювання;

Журналу транзакцій. Проводиться резервування всіх транзакцій в журналі.

Файлу або файлової групи. Проводиться резервування всіх даних, що містяться у файлі або файловій групі.

Файлове диференційоване. Проводиться резервування всіх сторінок даних, модифікованих з моменту останнього резервного копіювання файлу або файлової групи.

Відновлення завжди починається з використання повної резервної копії. Після цього з архівів диференційованого і транзакційного резервування відновлюються всі транзакції, виконані з моменту створення повної резервної копії. Модель відновлення конфігурує налаштування бази даних SQL Server таким чином, щоб забезпечити той тип відновлення, який необхідний базі даних. Ключові відмінності між різними моделями відновлення пов'язані з тим, в якій мірі в них задіяний журнал транзакцій і, які дані в ньому реєструються.

Створення відмовостійких систем.

Для створення відмовостійких систем вищеперелічених засобів відновлення недостатньо, оскільки потрібно забезпечити безперервний режим роботи сервера. Для підвищення надійності та (або) продуктивності дискової підсистеми жорсткі диски об'єднуються в RAID масиви. Найбільш часто використовуваними є RAID масиви рівнів 0, 1, 5, 6 і 10.

RAID 0 (Stripe). Режим, при використанні якого досягається максимальна продуктивність, але не надійність. Дані рівномірно розподіляються по дисках масиву, диски об'єднуються в один, який може бути розмічений на кілька. Розподілені операції читання і запису дозволяють значно збільшити швидкість роботи, оскільки кілька дисків одночасно читають / записують свою порцію даних. Користувачеві доступний весь обсяг дисків, але це знижує надійність зберігання даних, оскільки при відмові одного з дисків масив зазвичай руйнується і відновити дані практично неможливо[6].

RAID 1 (Mirror). Кілька дисків (зазвичай 2), що працюють синхронно на запис, тобто повністю дублюють один одного. Підвищення продуктивності відбувається тільки при читанні. Найнадійніший спосіб захистити інформацію від збою одного з дисків. Через високу вартість зазвичай використовується при зберіганні дуже важливих даних. Висока вартість обумовлена тим, що лише половина від загальної ємності дисків доступна для користувача.

RAID 10. Іноді також називається RAID 1 + 0, так як є комбінацією двох перших варіантів (масив RAID 0 з масивів RAID 1). Має всі швидкісні

переваги RAID 0 і перевага надійності RAID 1, зберігаючи недолік - високу вартість дискового масиву, так як ефективна ємність масиву дорівнює половині ємності використаних в ньому дисків. Для створення такого масиву потрібно мінімум 4 диска (їх число повинне бути парним).

RAID 5. Масив, також використовує розподілене зберігання даних аналогічно RAID 0 (і об'єднання в один великий логічний диск) плюс розподілене зберігання кодів парності для відновлення даних при збої. Можливо як одночасне читання, так і запис. Плюсом цього варіанту є те, що доступна для користувача ємність масиву зменшується на ємність лише одного диска, хоча надійність зберігання даних нижче, ніж у RAID 1. По суті, є компромісом між RAID 0 і RAID 1, забезпечуючи досить високу швидкість роботи при непоганий надійності зберігання даних. Мінімальна кількість дисків для такого масиву - 3. У разі відмови одного диска з масиву дані можуть бути відновлені без втрат в автоматичному режимі, хоча сам процес відновлення відбувається з повним навантаженням на робочі вінчестери, що при перегріванні може привести до повного виходу томи з ладу.

Висновки до розділу

1. Цілісність БД не гарантує достовірності, що міститься в ній інформації, але забезпечує принаймні правдоподібність цієї інформації, відкидаючи свідомо неймовірні, неможливі значення.
2. Транзакція є ключовим поняттям при організації відновлення цілісного стану БД
3. Посилальна цілісність забезпечує підтримку несуперечливого стану БД (узгодженого стану зовнішніх ключів) в процесі модифікації даних, а також при виконанні операцій додавання або видалення записів.

3 ЗАХИСТ ДАНИХ ТА ЇХ МЕТОДИ РЕАЛІЗАЦІЇ

Реалізація всіх правил «бізнес логіки» за допомогою збережених процедур, функцій і тригерів, які є фрагментами коду високорівневої мови програмування-розширення мови SQL, надає значно вищий рівень захисту даних, ніж їх реалізація в клієнтських програмах.

3.1 Визначення віртуальних таблиць

Віртуальні таблиці (VIEW)– це тимчасові об'єкти бази даних, якіформуються динамічно при зверненні до них Інформація у віртуальних таблицях не зберігається постійно, як в базових таблицях. Отже,віртуальні таблиці - це іменований запит, що зберігається в базі даних. Віртуальним таблицямне потрібновиділяти зайву дискову пам'ять, крім тієї, де для зберігаються самі таблиці. Віртуальні таблиці не можуть існувати самі по собі, а визначаються тільки в термінах однієї або декількох таблиць. Застосовуючи їх, розробник бази даних створює інтерфейс програми, що не залежить на всі 100% від реально існуючих таблиць, а також надає кожному користувачу або групі користувачів обмежений набір даних, приховуючи поля і записи з конфіденційною інформацією. Віртуальні таблиці дозволяють обмежити доступ до даних на рівні записів, доповнюючи, таким чином, дискреційну модель розмежування доступу. Поведінка і зовнішній вигляд віртуальних таблиць не відрізняється від базової таблиці. У користувача створюється враження, що він працює зі справжньою, реальною таблицею[7].

Оператори створення, зміни і видалення віртуальних таблиць на мові MS SQL Server збігаються і представлені такими командами:

```
{CREATE| ALTER} VIEW ім'я_віртуальні таблиці
[(ім'я_стовбчика [,...n])] [WITH ENCRYPTION]
AS SELECT … [WITH CHECK OPTION]
```

```
DROP VIEW [,...n]
```

Transact-SQL

Розглянемо мову Transact-SQL (або коротко T-SQL), який використовується в СУБД Microsoft SQL Server.

Опишемо змінні. Як і в інших високорозвинених мовах програмування, перш ніж почати використовувати змінну, її необхідно створити і описати. При введенні змінної вказується її ім'я і тип даних. Для введення змінної призначена команда DECLARE, що має наступний синтаксис:

```
DECLARE {@local_variable data_type} [,..n]
```

Ім'я змінної обов'язково повинно починатися з символу @. Наступні символи можуть бути будь-якими. Обмежень на другий символ імені не накладається (допустимо використання змінної з ім'ям @). За допомогою однієї команди DECLARE можна оголосити відразу кілька змінних, наприклад:

```
DECLARE @aa int, @str varchar(25), @l_ int, @qwerty bit, @@
float
```

За замовчуванням щойно створені змінні містять порожні значення NULL і перед використанням повинні бути проініціалізовані. Для присвоєння значення змінної можна використовувати оператори SET і SELECT. Оператор SET дозволяє визначити тільки одну змінну, наприклад:

```
SET @aa = 15

SET @str = 'asdf ghjkl'
```

Оператор SELECT дозволяє визначити відразу кілька змінних:

```
SELECT @aa = 15, @str = 'asdf ghjkl'
```

Оператор SELECT можна використовувати також для перегляду значень змінних, наприклад, для налагодження коду. Зручність роботи з змінними в

Transact-SQL полягає в тому, що змінні можуть безпосередньо вбудовуватися в інструкції запитів.

Область визначення змінних поширюється на пакет коду. Пакетом є частина коду, розділена операторами GO. Клієнтська програма відправляє серверу команди на виконання у вигляді пакетів. Після виконання всіх команд пакета (або в ході виконання пакету) сервер повертає клієнтові результат. Після цього клієнт може відправити наступний пакет і т.д.

3.2 Типи процедур

Для виконання дій дії з адміністрування сервера виконуються використовуються системні збережені процедури (мають префікс sp), які зберігаються в системній базі даних і можуть бути викликані в контексті будь-якої іншої бази даних. Вони забезпечують роботу системних таблиць: змінюють, додають, видаляють і вибірки даних з системних таблиць, як для користувачів, так і системних баз даних.

Процедури, що реалізують ті чи інші дії, визначені користувачем зберігаються в конкретній базі даних. Виклик такої процедури можливий в контексті тієї бази даних, де знаходиться процедура[8].

Тимчасові збережені процедури можуть існувати лише деякий час, після чого сервер їх знищує автоматично. Вони поділяються на глобальні і локальні, які є тимчасовими процедурами, їх можна викликати тільки з того з'єднання, в якому створені. Ім'я такої процедури повинно починатися з одного символу #. При відключенні користувача, перезапуску або зупинці сервера збережені процедури будуть автоматично видалені. Глобальні тимчасові процедури, доступні для будь-яких з'єднань сервера, для визначення яких потрібно дати ім'я, яке починається з символів ##. Такі процедури будуть видалені при перезапуску або зупинці сервера, а також при закритті з'єднання, в контексті якого вони були створені.

Створити нову та змінити наявну процедуру можна за допомогою команди:

```
{CREATE | ALTER } [PROCEDURE] ім'я_процедури [;номер]

[{@ім'я_параметратип_даних } [VARYING ] [=default] [OUT-PUT] ] [,...n]

[WITH { RECOMPILE | ENCRYPTION }] AS sql_оператор [...n]
```

У створеній процедурі використовують префікси sp_, #, що вказує, що вона системна або тимчасова. У назві процедури не допускається вказувати ім'я власника, якому належить створена процедура, а також ім'я бази даних, де вона повинна бути розміщена. Таким чином, щоб розмістити створену збережену процедуру в конкретній базі даних, необхідно виконати команду CREATE PROCEDURE в контексті цієї бази даних. Якщо звертаємось з тіла збереженої процедури до об'єктів тієї ж бази даних то використовуються укорочені імена, тобто без вказівки імені бази даних. Якщо потрібно звернутися до об'єктів в інших базах даних вказуємо обов'язково ім’я бази даних.

Номер в імені - це ідентифікаційний номер збереженої процедури, який визначає її в групі процедур. Для зручності управління процедурами логічно однотипні процедури, можна групувати, присвоюючи їм однакові імена, але різні ідентифікаційні номери.

Для передачі вхідних і вихідних даних в створюваній збереженій процедурі можуть використовуватися параметри, імена яких, як і імена локальних змінних, повинні починатися з символу @. В одній збереженій процедурі можна задати безліч параметрів, розділених комами. У тілі процедури не повинні застосовуватися локальні змінні, чиї імена збігаються з іменами параметрів цієї процедури.

Для визначення типу даних, який буде мати відповідний параметр збереженої процедури, можна використовувати будь-які типи даних SQL. Наявність ключового слова OUTPUT означає, що відповідний параметр

призначений для повернення даних зі збереженої процедури. Однак це зовсім не означає, що параметр не підходить для передачі значень в збережену процедуру. Вказівка ключового слова OUTPUT наказує серверу при виході з процедури привласнити поточне значення параметра локальної змінної, яка була вказана при виклику процедури в якості значення параметра. Відзначимо, що при вказівці ключового слова OUTPUT значення відповідного параметра при виклику процедури може бути поставлено лише за допомогою локальної змінної. Не дозволяється використання будь-яких виразів або констант, допустимих для звичайних параметрів[9].

Ключове слово VARYING застосовується спільно з параметром OUTPUT, що має тип CURSOR. Воно визначає, що вихідним параметром буде результуюча безліч. Default означає, що параметру можна привласнити значення за замовчуванням. Таким чином, при виклику процедури можна не вказувати явно значення відповідного параметра.

Так як сервер кешує план виконання запиту і компілює код, при наступному виклику процедури будуть використовуватися вже готові значення. Однак в деяких випадках все ж потрібно виконувати перекомпіляцію коду процедури. Вказівка ключового слова RECOMPILE наказує системі створювати план виконання процедури при кожному її виклику.

Ключове слово ENCRYPTION наказує серверу виконати шифрування коду збереженої процедури, що може забезпечити захист від використання авторських алгоритмів, що реалізують роботу збереженої процедури.

Ключове слово AS розміщується на початку власне тіла збереженої процедури, тобто набору команд SQL, за допомогою яких і буде реалізовуватися та чи інша дія. У тілі процедури можуть застосовуватися практично всі команди SQL, оголошуватися транзакції, встановлюватися блокування і викликатися інші процедури, що зберігаються. Вихід з процедури, що може здійснюватися за допомогою команди RETURN.

Видалення збереженої процедури здійснюється командою:

```
DROP PROCEDURE {ім'я_процедури} [,...n]
```

3.3 Визначення тригера в стандарті мови SQL

Під тригером будемо розуміти такий спеціальний тип збережених процедур, що запускаються сервером автоматично при спробі змінити дані в таблицях. Їх використовують для перевірки цілісності даних, а також для відкату транзакцій. Отже, тригер – це відкомпільована SQL-процедура, яка виконується при настанні певних подій всередині реляційної бази даних. Користувачам бази даних зручно застосовувати тригер, але їх використання часто пов'язане з додатковими витратами ресурсів на операції введення / виводу. Якщо однакових результатів можна досягти за допомогою збережених процедур або прикладних програм, то застосовувати тригери недоцільно. Створювати тригер може тільки власник бази даних. Це обмеження дозволяє уникнути випадкової зміни структури таблиць, способів зв'язку з ними інших об'єктів.

Найчастіше тригери застосовуються для підтримки цілісності даних в базі, тому що за допомогою обмежень цілісності і значень за замовчуванням не завжди можна домогтися потрібного рівня функціональності. Часто потрібно реалізувати складні алгоритми перевірки даних, відстежувати зміни значень таблиці, щоб потрібним чином змінити пов'язані дані.

Кожен тригер прив'язується до конкретної таблиці і виконується в складі відповідної транзакції модифікації даних. У разі виявлення помилки або порушення цілісності даних в коді тригера можна зробити відкат транзакції. Таким чином внесення змін (подія, яка викликала тригер) скасовується. Скасовуються також всі зміни, вже зроблені тригером. На відміну від звичайної процедури, тригер виконується неявно в кожному випадку виникнення тригерній події.

Тригер не має аргументів. Основний формат команди CREATE TRIGGER показаний нижче:

```
CREATE TRIGGER ім'я_тригера
BEFORE | AFTER <тригерна_подія > ON <ім'я_таблиці > [REFERENCING <список_псевдонімів >] [FOR EACH {ROW|STATEMENT}]
[WHEN(умова_тригера)]
<тіло_тригера>
```

Тригер створюється тільки в поточній базі даних, але допускається звернення всередині тригера до інших баз даних, в тому числі ідо тих, що розташовуються на віддаленому сервері. Ім'я тригера має бути унікальним в межах бази даних, можна ще вказати ім'я власника. При вказівці аргументу WITH ENCRYPTION сервер виконує шифрування коду тригера, щоб ніхто, навіть адміністратор, не міг отримати до нього доступ і прочитати його.

Типи тригерів. У SQL Server існує два параметри, що визначають поведінку тригерів:

AFTER. Тригер виконується після успішного виконання викликаних ним команд. Якщо ж команди з якої-небудь причини не можуть бути успішно завершені, тригер не виконується. Слід зазначити, що зміни даних в результаті виконання запиту користувача і виконання тригера здійснюється в тілі однієї транзакції: якщо станеться відкат тригера, то будуть відхилені і призначені для користувача зміни. Можна визначити кілька AFTER-тригерів для кожної операції (INSERT, UPDATE, DELETE). Якщо для таблиці передбачено виконання кількох AFTER-тригерів, то за допомогою системної збереженої процедури sp_settriggerorder можна вказати, який з них буде виконуватися першим, а який останнім. За замовчуванням в SQL Server всі тригери є AFTER-тригерами.

INSTEAD OF. Тригер викликається замість виконання команд.

На відміну від AFTER-тригера INSTEAD OF-тригер може бути визначений як для таблиці, так і для подання. Для кожної операції INSERT, UPDATE, DELETE можна визначити тільки один INSTEAD OF тригер.

Програмування тригера. При виконанні команд додавання, зміни і видалення записів сервер створює дві спеціальні таблиці: inserted і deleted. У них містяться списки рядків, які будуть вставлені або видалені після закінчення транзакції. Структура таблиць inserted і deleted ідентична структурі таблиць, для якої визначається тригер. Для кожного тригера створюється свій комплект таблиць inserted і deleted, тому ніякої іншої тригер не зможе отримати до них доступ. Залежно від типу операції, що отримала виконання тригера, вміст таблиць inserted і deleted може бути різним:

INSERT - в таблиці inserted містяться всі рядки, які користувач намагається вставити в таблицю; в таблиці deleted не буде жодного рядка; після завершення тригера всі рядки з таблиці inserted перемістяться в вихідну таблицю;

DELETE - в таблиці deleted будуть міститися всі рядки, які користувач спробує видалити; тригер може перевірити кожен рядок і визначити, чи дозволено її видалення; в таблиці inserted не опиниться жодного рядка;

UPDATE - при її виконанні в таблиці deleted знаходяться старі значення рядків, які будуть видалені при успішному завершенні тригера. Нові значення рядків містяться в таблиці inserted. Ці рядки додадуться в вихідну таблицю після успішного ви-конання тригера.

Для отримання інформації про кількість рядків, яке буде змінено при успішному завершенні тригера, можна використовувати глобальну змінну @@ ROWCOUNT; вона повертає кількість рядків, оброблених останньою командою. Слід підкреслити, що тригер запускається ні до спроби змінити конкретний рядок, а в момент виконання команди зміни. Одна така команда впливає на безліч рядків, тому тригер повинен обробляти всі ці рядки.

Якщо тригером буде виявилено, наприклад, що зі 100 вставлених, змінюваних чи видалених рядків тільки один не задовольнятиме тим чи іншим

умовам, тоді ніякий рядок не буде вставлений, змінений або видалений. Така поведінка обумовлена вимогами транзакції - повинні бути виконані або всі модифікації, або жодної[10].

Тригер виконується як неявно певна транзакція, тому всередині тригера допускається застосування команд управління транзакціями. Зокрема, при виявленні порушення обмежень цілісності для переривання виконання тригера і скасування всіх змін, які намагався виконати користувач, необхідно використовувати команду ROLLBACK TRANSACTION.

Для отримання списку полів, що змінились при виконанні команд INSERT або UPDATE, які викликали виконання тригера, можна використовувати функцію COLUMNS_UPDATED. Вона повертає двійкове число, кожний біт якого, починаючи з молодшого, відповідає одному стовпцю таблиці (в порядку проходження стовпців при створенні таблиці). Якщо біт встановлено у значення "1", то відповідний стовпець був змінений. Крім того, факт зміни стовпчика визначає і функція UPDATE (ім'я_стовпця).

Висновки до розділу

1. Віртуальні таблиці - це іменований запит, що зберігається в базі даних. Віртуальні таблиці не вимагають для свого зберігання дискової пам'яті, за винятком пам'яті, необхідної для зберігання визначення самих таблиць. Віртуальні таблиці не можуть існувати самі по собі, а визначаються тільки в термінах однієї або декількох таблиць.
2. Щоб визначити тип даних, який буде мати відповідний параметр збереженої процедури, будемо використовувати будь-які типи даних SQL.
3. Тригери використовуються для перевірки цілісності даних, та для відкату транзакцій. Отже, тригер – це відкомпільована SQL-

процедура, яка виконується при настанні певних подій всередині реляційної бази даних.

4. Тригер використовують як неявно певну транзакцію, тому всередині тригера допускається застосування команд управління транзакціями.

4 РЕАЛІЗАЦІЯ МЕТОДІВ ЗАХИСТУ БАЗИ ДАНИХ

Найпростіший спосіб захисту бази даних – це шифрування. При використанні службових програм або текстових редакторів файл бази даних стискається і стає недоступним для читання. Шифрування незахищеної бази даних неефективно, оскільки кожен зможе відкрити таку базу даних і отримати повний доступ до всіх її об'єктів, тому шифрування, зазвичай, застосовується при електронній передачі бази даних або збереженні її на будь-якому носії інформації.

4.1 Система захисту Microsoft Access

Система захисту СУБД Microsoft Access базується на використанні таких засобів:

1) шифрування / дешифрування;

2) показ або приховування об'єктів у вікні бази даних;

3) використання пароля БД;

4) використання захисту на рівні користувача.

Одним із найпростіших способів захисту є установка пароля для відкриття бази даних (.mdb). Якщо пароль установлений, то при кожному відкритті бази даних буде з'являтися діалогове вікно, в яке потрібно його ввести. Тільки ті користувачі зможуть відкрити базу даних, які введуть правильний пароль. Цей спосіб більш надійний, ніж попередній (Microsoft Access шифрує пароль, тому до нього немає доступу при безпосередньому читанні файлу бази даних), але він діє лише при відкритті бази даних. Після відкриття бази даних всі об'єкти стають доступними для користувача (поки не визначені інші типи захисту, описані нижче в цьому розділі). Даний спосіб не є надійним способом захисту баз даних. Існує достатня кількість

безкоштовних і платних утиліт, що відображають пароль. У тому числі доступні вихідні коди коду на VBA, що дозволяють прочитати такий пароль.

4.2 Використання захисту на рівні користувача

Найбільш гнучкий і поширений спосіб захисту бази даних називають захистом на рівні користувача, що дозволяє встановити різні рівні доступу до важливих даних і об'єктів в базі даних для різних користувачів. Даний вид захисту реалізований в версіях Microsoft Access по 2003 включно. Починаючи з Access 2007, Microsoft реалізує інший шлях захисту на основі цифрових підписів. Захист на рівні користувачів Microsoft Access реалізує вибірковий підхід розмежування доступу. При активізації даного виду захисту в базі даних Microsoft Access адміністратор бази даних або власник об'єкта надає певні дозволи окремим користувачам і групам користувачів на наступні об'єкти: таблиці, запити, форми, звіти і макроси. Облікові записи системи безпеки визначають користувачів і групи користувачів, яким дозволений доступ до об'єктів. Ця інформація, яку називають відомостями про робочу групу, зберігається в спеціальному файлі робочої групи. Файл робочої групи містить відомості про користувачів, що входять в робочу групу[11]. До таких відомостей належать імена облікових записів користувачів, їхні паролі і імена груп, в які входять користувачі. Для створення файлу робочих груп і підключення його до БД використовується спеціальний додаток «Адміністратор робочих груп».

Після входу в систему аналізується файл робочої групи, в якому кожен користувач ідентифікується унікальним кодом. Групам і користувачам надаються дозволи, що визначають можливість їх доступу до кожного об'єкту бази даних. Існують два типи дозволів на доступ: явні і неявні. Дозволи називаються явними, якщо вони безпосередньо присвоєні обліковому запису користувача; такі дозволи не впливають на рішення інших користувачів.

Неявними називаються дозволи на доступ, присвоєні обліковому запису групи. Користувач, включений в таку групу, отримує всі дозволи, надані групі; видалення користувача з цієї групи позбавляє його всіх дозволів, привласнених даній групі. Дозволи зберігаються в самій БД. Роздільне зберігання облікових записів і матриці доступу підвищує надійність системи захисту. Якщо користувач з даними ідентифікатором не знаходяться в приєднаному файлі робочих груп, доступ до об'єктів відхиляється.

Для того, щоб користувач зміг виконати будь-яку операцію з захищеним об'єктом бази даних, його поточні дозволу повинні визначатися комбінацією явних і неявних дозволів на доступ. На рівні користувачів завжди діють мінімальні обмеження, які накладаються явними дозволами для них і для всіх груп, до яких належить даний користувач. Щоб управляти робочою групою створюються нові групи і визначаються дозволи на доступ для цих груп, а не для індивідуальних користувачів. Після цього зміна дозволів для окремих користувачів здійснюється шляхом додавання користувачів в групи або видалення їх з груп. Нові дозволи надаються відразу всім членам групи в одній операції.

У Microsoft Access визначені дві стандартні групи: адміністратори (група «Admins») і користувачі (група «Users»), а визначення додаткових груп не допускається. Члени групи «Users» можуть бути допущені тільки до перегляду даних в таблицях.

Члени групи «Admins» мають дозвіл на доступ до всіх об'єктів бази даних. Якщо для системи захисту досить групи адміністраторів і групи користувачів, то немає необхідності створювати інші групи. В цьому випадку необхідно присвоїти відповідні дозволи на доступ стандартній групі «Users» і додати додаткових адміністраторів в стандартну групу «Admins». Кожен новий користувач автоматично додається в групу «Users». Типові дозволу на доступ для групи «Users» можуть включати «Читання даних» і «Оновлення даних» для таблиць і запитів та «Відкриття / запуск» для форм і звітів

У разі необхідності більш розгалуженої структури управління для різних груп користувачів, є можливість створення нових груп, присвоєння групам різних наборів дозволів на доступ і додавання нових користувачів до відповідних груп. Наприклад, можна організувати захист бази даних «Замовлення» за допомогою створення облікових записів «Керуючі» для керівництва фірми, «Представники» для торгових представників і «Персонал» для адміністративного персоналу. Після цього слід присвоїти найбільшу кількість дозволів групі «Керуючі», проміжне кількість групі «Представники» і мінімальне групі «Персонал». При створенні облікового запису для нового співробітника цей запис буде додаватися в одну з груп, і співробітник автоматично отримає дозвіл, що належить цій групі.

Коли користувач вперше запускає Access після установки Microsoft Office, Access автоматично створює файл робочої групи, який ідентифікується по зазначеним користувачем імені та назвою організації. Відносне розташування файлу робочої групи записується в наступні параметри реєстру:
`HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\Access\Jet\4.0\Engines\SystemDB` та

`HKEY_USERS\.DEFAULT\Software\Microsoft\Office\10.0\Access\Jet\4.0\Engines\SystemDB`

Так як отримання цієї інформації не становить особливих труднощів, існує ймовірність того, що хтось зможе створити іншу версію файлу робочої групи і придбати невід'ємні дозволу на доступ за допомогою стандартного облікового запису адміністратора, тому що програма установки автоматично додає в групу «Admins» стандартний обліковий запис користувача «Admin». Щоб уникнути цього, при активізації захисту на рівні користувача необхідно створити новий файл робочої групи і приєднати його туди, де захищається БД. Створений файл робочої групи, використаний при відкритті, захищається, як і будь-який інший бази даних, що викликає серйозні незручності. Якщо планується використовувати файл робочої групи для конкретної БД, слід

створити ярлик для відкриття цієї БД і в командному рядку якого вказати параметр / wrkgrp і повне ім'я файлу робочої групи. наприклад:

```
"C:\Program Files\MSOffice2003\OFFICE11\MSACCESS.EXE" /WrkGrp C:\test\gr.mdw
```

Користувачі, які є членами групи «Admins» або власниками об'єктів і не мають дозволу на виконання якої-небудь дії, мають можливість призначити їх собі самі.

Користувач, який створив таблицю, запит, форму, звіт або макрос, є власником цього об'єкта. Крім того, користувачі, що входять до групи «Admins», можуть також змінити власника об'єкта або заново створити об'єкт, що є альтернативним способом зміни власника об'єкта. Для створення об'єкта достатньо імпортувати або експортувати цей об'єкт в іншу базу даних або зробити копію об'єкта. Цей прийом є найпростішим способом зміни власника об'єктів, в тому числі і всієї бази даних.

Деякі дозволи на доступ автоматично надають інші дозволи. Наприклад, дозвіл «оновлення даних» на таблицю автоматично надає дозвіл «читання даних» і «читання макета», необхідні для зміни даних в таблиці. Дозволи «зміна макета» і «читання даних» автоматично надають дозвіл «читання макета». Для макросів дозвіл «читання макета» тягне надання дозволу «відкриття / запуск». При зміні об'єкта і його подальшого збереження дозвіл на доступ до нього зберігаються. Однак якщо об'єкт зберігається під новим ім'ям, він стає новим об'єктом і, отже, отримує дозволи, встановлені за замовчуванням для даного типу об'єктів, а не дозволу вихідного об'єкта.

Для зняття захисту на рівні користувача необхідно створити нову БД, в ярлику прописати шлях до цієї БД, до файлу робочих груп захищеної БД і при відкритті вказати ім'я та пароль власника. Потім необхідно імпортувати в створену БД всі об'єкти захищеної БД, після чого змінити власника всіх об'єктів на вбудований обліковий запис Admin. На закінчення залишається відключити вікно появи запиту пароля, знявши пароль запису «Admin».

4.3 Методи протидії злому захисту Access

Захист за допомогою пароля БД, що містить некоректні символи. В першу чергу цей спосіб націлений на протидію визначення паролів за допомогою спеціальних програм. Спосіб заснований на тому, що пароль БД формату Access 2000 і 2002-2003 – текстовий рядок у форматі Unicode. При цьому, немає ніяких обмежень на її вміст. Стандартний спосіб установки і використання пароля БД має на увазі його введення з клавіатури в діалоговому вікні. Якщо рядок пароля містить неправильні символи, то вони не будуть коректно відображені програмою, що відкриває паролі БД. З іншого боку, цей пароль не можна ввести в діалоговому вікні при відкритті БД в MS Access, що вимагає написання спеціальної програми завантажувача. У специфікації баз даних і в довідці по DAO 3.60 зазначено, що максимальне число символів в паролі - 14. Але насправді їх може бути 20. Завдання пароля з 20 символів з наявністю кілька нецензурних символів призводить більшість зламували програм в повне замішання.

Захист шляхом модифікації файлу БД і його розширення. Даний спосіб захисту заснований на модифікації перших байт файлу. Таким чином, перед відкриттям БД за допомогою додаткової програми в її файл записується правильний заголовок, що зберігається в програмі доступу, а після закриття повертається неправильний. При спробі відкрити файл даних за допомогою Access з'являється повідомлення про помилку. Цей спосіб захисту добре поєднати зі способом зміни розширення файлу. Наприклад, можна взяти заголовок dbf файлу і записати його в початок mdb файлу. Далі змінити розширення файлу на dbf. Зазначений метод не досить ефективний, так як програму, що працює з БД, можна перервати штучно і на диску залишиться не захищена БД. Тому варто його використовувати тільки в поєднанні з іншими способами[12].

4.4 Архітектура системи безпеки SQL Server

Система безпеки MS SQL Server реалізує дискреційний і рольовий підходи до управління доступом і має два рівні: рівень сервера і рівень бази даних. На рівні сервера дозволяється або відхиляється доступ користувачів до самого сервера. На рівні бази даних користувачі, які мають доступ на рівні сервера, отримують доступ до об'єктів бази даних. Такий підхід дозволяє більш гнучко управляти доступом користувачів до баз даних.

На рівні сервера система безпеки оперує наступними поняттями:

- аутентифікація (authentication);
- обліковий запис (login);
- вбудованіролісервера (fixedserverroles).

На рівні бази даних використовуються поняття:

- користувач бази даних (database user);
- фіксована роль бази даних (fixed database role);
- призначена для користувача роль бази даних (users database role);
- роль додатка (application role).

SQL Server використовує двоетапну схему аутентифікації. Спочатку користувач, який аутентифікується сервером, після успішної аутентифікації на сервері може отримати доступ до однієї або декількох БД. SQL Server зберігає всю інформацію про реєстраційні записи в базі даних master.

SQL Server пропонує два режими аутентифікації користувачів:

Режим аутентифікації засобами OS Windows. В цьому режимі SQL Server повністю довіряє операційній системі при аутентифікації користувачів;

Змішаний режим аутентифікації (Windows Authentication and SQL Server Authentication). В цьому режимі системи аутентифікації Windows і SQL Server існують незалежно один від одного.

При установці SQL Server одним з перших рішень, які слід прийняти, є вибір використаного методу аутентифікації. Встановлений при інсталяції

режим аутентифікації можна змінити на сторінці Security діалогового вікна SQL Server Properties в утиліті Management Studio. У програмному коді встановлений режим аутентифікації можна перевірити за допомогою системної збереженої процедури xp_loginconfig:

```
EXEC xp_loginconfig 'login mode'
```

Результат виконання цієї процедури виглядає наступним чином:

```
name config_value

login mode     Mixed
```

Щоб додати обліковий запис користувача в ту чи іншу фіксовану роль сервера можна скористатися збереженою процедурою sp_addsrvrolemember, що має наступний синтаксис:

```
sp_addsrvrolemember [@loginame =] 'login' , [@rolename =] 'role'
```

4.5 Аутентифікація Windows

Використання аутентифікації Windows означає, що користувачеві для доступу до SQL Server досить мати обліковий запис Windows. Ідентифікатор безпеки Windows передається з операційної системи на сервер баз даних. SQL Server передбачає, що процес реєстрації користувачів в мережі достатньо захищений, і тому не виконує ніяких додаткових перевірок. Користувач автоматично отримує відповідні права доступу до даних SQL Server відразу ж після реєстрації в домені. Такий метод надання доступу називається встановленням довірчого з'єднання[13]. Операційна система працює з обліковими записами (logins), які містять всі дані про користувача, включаючи його ім'я, пароль, членство в групах, каталог за замовчуванням і т. Д. Кожна обліковий запис має унікальний ідентифікатор (Login ID) або, як його

називають по-іншому, ідентифікатор безпеки (SID, Security Identification), за допомогою якого користувач реєструється в мережі.

Ідентифікатор – це довге (85-бітове) шістнадцяткове число, яке генерується випадковим чином операційною системою під час створення облікового запису. Такий підхід дозволяє уникнути підробки облікових записів користувачів. Якщо користувач був видалений з домену, то навіть повторне створення користувача з аналогічними характеристиками (ім'я облікового запису, пароль, членство в групі і т. Д.) не дасть можливості отримати доступ до об'єктів, до яких мав доступ оригінальний користувач. Стосовно до SQL Server можна сказати, що якщо користувач домену мав певні права доступу, але був вилучений, то ніхто не зможе привласнити його права доступу.

Аутентифікація Windows передбачає збереження в системній базі даних Master тільки ідентифікаційного номера облікового запису користувача в домені (SID). Інформація про ім'я користувача, його пароль зберігається в базі даних домену. Зміна імені користувача або його пароля ніяк не відіб'ється на правах доступу до SQL Server. Інформація про реєстрацію користувача і його членство в групах Windows зчитується SQL Server з бази даних системи безпеки домена під час підключення користувача. Якщо адміністратор вніс якісь зміни в обліковий запис користувача, наприклад, виключив його з деякої групи, то зміни позначаться тільки під час чергової реєстрації користувача в домені або в SQL Server в залежності від категорії зроблених змін.

Аутентифікація Windows дає певні переваги. На користувачів автоматично відбиваються всі правила політики безпеки, встановлені в домені. Користувачеві не доводиться запам'ятовувати ще один пароль. Це підвищує рівень загальної захищеності даних. Наприклад, автоматично контролюється мінімальна довжина пароля і термін його дії. Операційна система вимагає від користувача періодичної зміни пароля. Додатково можна заборонити користувачам установку паролів. Крім того, Windows має вбудовані засоби захисту від підбору паролів. Аутентифікація Windows працює також з групами

користувачів. Коли ім'я групи Windows передається в SQL Server в якості реєстраційного запису, будь-який член цієї групи може бути аутентифікований сервером баз даних. SQL Server також знає справжнє ім'я користувача Windows, що входить до групи, внаслідок чого додаток може виконувати аудит на рівні користувачів, а також на рівні груп користувачів.

Для створення облікового запису користувача або групи Windows в SQL Server в програмному коді використовується системна збережена процедура sp_grantlogin. Як аргумент їй необхідно передавати повне ім'я користувача, що включає ім'я домена, як в наступному прикладі:

```
EXEC sp_grantlogin 'RFE\Sidorov'
```

Для видалення облікового запису або групи Windows з SQL Server програмним шляхом використовується системна збережена процедура sp_revokelogin:

```
EXEC sp_revokelogin 'RFE\Sidorov'
```

Заборону облікового запису Windows можна зробити за допомогою системної збереженої процедури sp_denylogin. Отже, доступ будь-якого користувача в SQL Server може бути примусово закритий. Це повністю забороняє доступ користувача до SQL Server, навіть якщо він був відкритий за допомогою іншого методу, наприклад:

```
EXEC sp_denylogin 'RFE\Sidorov'
```

Якщо користувач Windows був доданий в SQL Server, а потім був видалений з домену Windows, то незважаючи на те, що він формально має доступ до сервера, не має доступу до ресурсів мережі і, отже, до всього комплексу засобів SQL Server. Збережена системна процедура sp_validatelogins дозволяє знайти "втрачених" користувачів і повернути їхні ідентифікатори системі безпеки Windows NT та імена облікових записів.

Наявність додаткових реєстраційних записів SQL Server доречно тоді, коли аутентифікація Windows недоступна або не підходить для створюваної

системи. Ця реєстрація має зворотну сумісність з попередніми версіями сервера, а також з додатками, в яких реєстраційний запис вбудований в програмний код. Аутентифікація SQL Server в основному застосовується клієнтами, для яких недоступна реєстрація в домені Windows. Наприклад, користувачами Novell NetWare, Unix і т. Д. При підключенні до SQL Server через Internet реєстрація в домені не виконується, тому в даному випадку також необхідно використовувати аутентифікацію SQL Server.

В процесі інсталяції SQL Server і виборі змішаного режиму аутентифікації майстер установки створює спеціально обліковий запис sa (system administrator) з порожнім паролем, який є членом фіксованої серверної ролі sysadmin і має всі права доступу до сервера. Найчастіше йому забувають привласнити пароль, і ці прямі ворота в системі безпеки відкривають шлях до сервера будь-якого зловмисника. Як правило, наявність такого пролому хакери перевіряють в першу чергу. Виходячи з цього, перш за все потрібно поставити пароль або просто відключити користувача sa і призначити адміністративної ролі sysadmin якомусь іншому користувачу (або створити додаткові ролі з адміністративними привілеями). Цей обліковий запис залишений для збереження зворотної сумісності з попередніми версіями SQL Server. Раніше обліковий запис був обов'язковим, мав абсолютні права з управління сервером і не міг бути вилученим. Користувачеві sa в процесі установки завжди присвоюється однаковий на всі комп'ютерах ідентифікатор безпеки 0x01.

Ідентифікатор безпеки зберігається в стовпці sid таблиці sysxlogins системної бази даних Master. У ній зберігається інформація про облікові записи як SQL Server, так і Windows. Для облікових записів SQL Server максимальний розмір ідентифікатора безпеки становить 16 байт, для облікових записів Windows - 28 байт. Кожному рядку цієї таблиці відповідає один обліковий запис. Таким чином, в таблиці sysxlogins може міститися досить багато рядків з інформацією про облікові записи. Для більш зручної роботи з локальними обліковими записами можна використовувати уявлення

syslogins, що включає тільки ті рядки таблиці sysxlogins, які мають в стовпці srvid (ідентифікаційний номер сервера) значення NULL.

Серед переданих процедурі аргументів обов'язковим є тільки ім'я реєстраційного запису[14]. Так як в даному випадку потрібно налаштування користувача, а не просто вибір його зі списку, дана задача більш складна, ніж виконання процедури sp_grantlogin. Наприклад, наступний програмний код створює користувачу SQL Server з ім'ям Den і призначає йому в якості бази даних за замовчуванням навчальну базу test:

```
EXEC sp_addlogin 'Den', 'myoldpassword', 'test'
```

Параметр шифрування skip_encryption вказує сервера зберігати пароль в системній таблиці sysxlogins без всякого шифрування. У той же час SQL Server очікує, що пароль обов'язково буде зашифрований, тому він не розпізнає створений таким чином пароль. Отже, слід уникати використання цього параметра

Після створення облікового запису можна змінити ідентифікатор безпеки за допомогою команди UPDATE, безпосередньо звернувшись до системної таблиці. При виборі значення для параметра @sid слід дотримуватися вимога унікальності ідентифікаторів безпеки. Тобто на поточному сервері до моменту реєстрації не повинно бути облікових записів з ідентифікатором безпеки, рівним обраному значенням. Список використаних ідентифікаторів безпеки локального сервера можна переглянути за допомогою наступного запиту:

```
SELECT sid FROM syslogins.
```

4.6 Система безпеки рівня бази даних

Для того щоб користувач мав можливість виконувати ті чи інші дії з об'єктами бази даних, він повинен попередньо отримати необхідні права доступу. Власник бази даних або конкретного об'єкта повинен надати користувачам бази даних можливість звернення до об'єктів бази даних. Крім прав доступу, які видаються користувачеві явно, існує клас неявних прав доступу. Неявні права доступу надаються користувачеві через членство у фіксованій ролі сервера або бази даних. Наприклад, немає ніякої іншої можливості надати користувачеві право на створення баз даних, крім як включити його в роль сервера dbcreator. Якщо користувач створив об'єкт в базі даних, то йому автоматично надаються максимальні права з управління цим об'єктом.

Таким чином, на рівні бази даних користувачеві можуть бути надані привілеї за допомогою прикріплення до фіксованої ролі бази даних або шляхом явного призначення привілеїв за допомогою оператора SQL GRANT. Винятком є користувачі, облікові записи яких включені в фіксовану роль сервера sysadmin. Члени цієї групи мають необмежені права в межах сервера, і, як наслідок, повний доступ до будь-якої базі даних, що є на сервері.

Інформація про користувачів, створених в базі даних, зберігається в системній таблиці sysusers. Інформацію про користувачів поточної бази даних можна отримати за допомогою системної збереженої процедури sp_helpuser:

```
USE pubs

EXEC sp_helpuser
```

Правом виклику зазначеної процедури, що володіють члени фіксованою ролі сервера sysadmin і члени фіксованих ролей бази даних db_owner і db_accessadmin. Параметр @loginame визначає ім'я облікового запису, якому

передбачається надати доступ до поточної бази даних. Зазначений обліковий запис повинен існувати на сервері.

За допомогою параметра @name_in_db вказується ім'я, яке буде присвоєно створюваному користувачеві. Це ім'я має бути унікальним в межах бази даних. Наведений далі приклад ілюструє використання збереженої процедури sp_grantdbaccess. У базі даних pubs створюється новий користувач з ім'ям Admin, який зв'язується з обліковим записом STORAGE \ Admin:

```
USE pubs

EXEC sp_grantdbaccess 'STORAGE\Admin', 'Admin'
```

Правом виклику зазначеної процедури, що володіють користувачі фіксованої ролі сервера sysadmin і члени фіксованих ролей бази даних db_owner і db_accessadmin. Єдиний параметр процедури визначає ім'я користувача, яке необхідно видалити. Однак, перш ніж зважитися на подібний крок, слід переконатися, що користувачеві не належить ніякий об'єкт бази даних. Якщо ж користувач є власником одного з об'єктів бази даних, то слід або видалити цей об'єкт за допомогою команди DROP, або змінити власника об'єкта за допомогою системної збереженої процедури sp_changeobjectowner. Наприклад, для видалення користувача Admin досить буде виконати команду:

```
EXEC sp_revokedbaccess 'Admin'
```

Щоб включити нового користувача в фіксовану роль бази даних, необхідно викликати системну збережену процедуру sp_addrolemember, що має синтаксис:

```
sp_addrolemember [@rolename =] 'role' , [@membername =] 'security_account'
```

За допомогою параметра @rolename вказується ім'я ролі, в яку потрібно додати нового члена. Зазначена роль повинна існувати в поточній базі даних. наприклад:

```
EXEC sp_addrolemember 'db_accessadmin', 'User1'
```

Для отримання інформації про членство у всіх ролях поточної бази даних можна використовувати процедуру sp_helprolemember. наприклад:

```
USE pubs

EXEC sp_helprolemember
```

Якщо фіксовані ролі призначені для наділення користувачів спеціальними правами в базі даних, то для користувача ролі служать лише для угрупування користувачів з метою полегшення управління їх правами доступу до об'єктів. Створення користувальної ролі виконується за допомогою системної збереженої процедури sp_addrole, яка має синтаксис:

```
sp_addrole [@rolename =] 'role' [, [@ownername =] 'owner']
```

Правом виконання зазначеної процедури, що володіють члени фіксованої ролі сервера sysadmin і фіксованих ролей бази даних db_owner і db_securityadmin. За замовчуванням власником ролі стає власник бази даних (користувач dbo). Таким чином, користувач dbo набуває повний контроль над роллю. Якщо необхідно привласнити права володіння роллю іншому користувачеві, то ім'я цього користувача має бути вказано за допомогою параметра @ownername. Вказаний користувач повинен бути в базі даних. Власником користувальницької ролі може також бути і роль додатка. Як приклад розглянемо створення користувальницької ролі AccessDBUser, власником якої буде користувач guest:

```
USE pubs

EXEC sp_addrole 'AccessDBUser', 'guest'
```

Щоб переглянути список явних прав доступу можливий запуск системної збереженої процедури sp_helprotect, що має синтаксис:

```
sp_helprotect [[@name =] 'object_statement'] [, [@username =] 'security_account'] [, [@grantorname =] 'grantor'] [, [@permissionarea =] 'type']
```

Наприклад

```
sp_helprotect Clients(дляобєкта Clients)
sp_helprotect @username = 'RFE\Joe'(для користувача RFE\Joe)
```

Дана процедура ефективна лише для перегляду явних привілеїв, призначених за допомогою оператора GRANT. Для перегляду всіх привілеїв, включаючи неявні, слід використовувати наступний SQL запит:

```
WHERE principal_type_desc <> 'DATABASE_ROLE'
UNION
--role members
```

SELECT rm.member_principal_name, rm.principal_type_desc, p.class_desc, p.object_name, p.permission_name, p.permission_state_desc,rm.role_name FROM perms_cte p right outer JOIN (

```
select role_principal_id, dp.type_desc as
principal_type_desc, member_principal_id,
user_name(member_principal_id) as member_principal_name,
user_name(role_principal_id) as role_name--,*
from sys.database_role_members rm INNER JOIN sys.database_principals dp
ON rm.member_principal_id = dp.principal_id) rm
ON rm.role_principal_id = p.principal_id order by 1
```

Для переміщення БД на інший комп'ютер необхідно спочатку від'єднати БД. Для цього на комп'ютері - джерелі спочатку слід змінити контекст на системну БД Master, потім викликати збережену процедуру sp_detach_db має один обов'язковий параметр - ім'я від'єднується бази даних:

```
USE [master]
```

```
GO

sp_detach_db 'test'

go
```

Після переміщення файлів БД в місце призначення проводиться зворотна процедура[15]. Якщо місце, куди скопійовані файли, відрізняється від каталогу за замовчуванням для сервера на комп'ютері призначення, то потрібно призначити даному каталогу повні права доступу групи SQLServer2005MSSQLUser $ MyPC $ SQLEXPRESS (примітка: між символами долара знаходиться ім'я сервера). Якщо сервер запущений від імені деякого користувача, то права призначаються йому, а не вищевказаній групі. Підключення БД проводиться за допомогою процедури, що sp_attach_db. Першим її аргументом йде ім'я БД, потім шлях до файлу БД з розширенням mdf, потім шлях до файлу БД з розширенням ldf, наприклад:

```
use [master]

gosp_attach_db          'test',          'D:\Service\Database
files\MSSQL.1\Data\test.mdf',                'D:\Service\Database
files\MSSQL.1\Data\test_log.ldf' go
```

Наступним кроком є відновлення втрачених користувачів. Покажемо процедуру відновлення на прикладі користувача operator. Спочатку проводиться додавання облікових записів БД.

```
exec sp_addlogin 'operator', 'password', 'test' go
```

Після з'єднання всі команди виконуються в клієнті. Для початку увійдемо в систему з правами адміністратора. Отримаємо список користувачів, включених в фіксовану роль сервера sysadmin:

```
sp_helpsrvrolemember 'sysadmin'
```

Включимо користувача RFE \ Joe в роль sysadmin:

```
sp_addsrvrolemember 'RFE\Joe', 'sysadmin'
```

У разі змішаної аутентифікації сервера додамо користувача RFE \ User (база за замовчуванням test):

```
sp_addlogin 'RFE\User', 'mypassword', 'test'
```

У разі аутентифікації Windows просто надасть доступ користувачеві RFE \ User до сервера

```
sp_grantlogin 'RFE\User'
```

Додамо користувача RFE\User в список користувачів бази даних test (с именем user)

```
USE test

go

sp_grantdbaccess 'RFE\User', 'user'
```

Включим RFE\User у фіксовану роль БД

```
sp_addrolemember 'db_datareader', 'user'
```

Додамо ще одного користувача Operator

```
sp_grantlogin 'RFE\Operator'

sp_grantdbaccess 'RFE\Operator', 'operator'
```

Надамо йомуправадоступу

GRANT SELECT, INSERT, UPDATE, DELETE ON Clients TO operator

GRANT SELECT, INSERT, UPDATE, DELETE ON Orders TO operator GRANT SELECT, INSERT, UPDATE, DELETE ON Goods TO operator DENY UPDATE(Goods.price) ON Goods TO operator

Переглянемосписокпризначенныхправдоступу

```
exec sp_helprotect @username='operator'
```

Висновки до розділу

1. Найпростіший спосіб захисту – це шифрування бази даних. При використанні службових програм або текстових редакторів файл бази даних стискається і стає недоступним для читання. Найбільш гнучкий і поширений спосіб захисту бази даних називають захистом на рівні користувача, що дозволяє встановити різні рівні доступу до важливих даних і об'єктів в базі даних для різних користувачів.
2. Захист реалізується шляхом модифікації файлу БД і його розширення. Даний спосіб захисту грунтується на модифікації перших байт файлу. Отже, перед відкриттям БД за допомогою додаткової програми в її файл записується правильний заголовок, що зберігається в програмі доступу, а після закриття перетворюється в неправильний.
3. Система безпеки MS SQL Sever реалізує дискреційний і рольовий підходи до управління доступом і має два рівні: рівень сервера і рівень бази даних.
4. SQL Server використовує двоетапну схему аутентифікації. Спочатку користувач, який аутентифікується сервером, після успішної аутентифікації на сервері може отримати доступ до однієї або декількох БД. SQL Server зберігає всю інформацію про реєстраційні записи в базі даних master.
5. Ідентифікатор безпеки Windows передається з операційної системи на сервер баз даних. SQL Server передбачає, що процес реєстрації користувачів в мережі достатньо захищений, і тому не виконує ніяких додаткових перевірок.
6. Неявні права доступу надаються користувачеві через членство у фіксованій ролі сервера або бази даних.

5 СТАРТАП-ПРОЕКТ

5.1 Основні відомості

Сутність стартап-проекту. Досліджуючи ринок мережевих технологій, було виявлено можливість розгортки БД за допомогою використання нових рішень. Зміст ідеї стартапу та визначення її характеристик наведено в табл. 5.1 та табл. 5.2.

Таблиця 5.1 – Зміст ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Запропонувати дієве і ефективне рішення для створення бази даних на базі існуючих технологій.	1. Бізнес	Оптимізація локальних процесів
	2. Виробництво	Оптимізація способу виробництва
	3. Фінансовий сектор	Покращення можливостей моніторингу та автоматизації транзакцій

Таблиця 5.2 – Визначення характеристик ідеї стартап-проекту

п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів		W (слабка сторона)	N (нейтральна торона)	S (сильнас торона)
		Запропонований метод	Загальновживаний метод			
.	Продаж професійного програмного забезпечення для БД	Дає змогу	Дає змогу	Може бути економічно затратним	Цінова політика можене задовільнти кінцевого споживача	Можливість зайняти відносно вільну нішу, продукт цікавийдля бізнесута виробництва
.	Створення нових мереж БД	Дає змогу	Не дає змогу	Потребує детального опрацювання аспектів захисту	Підтримка користувача, оновлення ПО	Можливість комерційної та дотаційної реалізації

5.3 Технологічний аудит ідеї стартап-проекту

У таблиці 5.3 оцінено можливість технологічної реалізації ідеї стартапу та показано технології, які можна застосувати для реалізації проекту.

Таблиця 5.3. Технологічна здійсненність ідеї проекту

п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
	Створення новихБД для надання можливостей покращення роботи в фінансовому секторі	Спеціалізоване обладнання для організації потокового мовлення	Присутня	Доступна
		Використання існуючих апаратних систем стільникових мереж	Присутня	Доступна
		Розробка власних апаратно-програмних рішень	Присутня	Доступна у випадку достатнього бюджету

Обрана технологія реалізації ідеї проекту: розробка та продаж програмного забезпечення для створення БД в фінансовому секторі з використанням існуючих методів та обладнання.

5.4 Аналіз можливостей ринку для запуску проекту

У таблиці 5.4 показано попередню характеристику потенційного ринку стартап-проекту.

Таблиця 5.4. Попередня характеристика потенційного ринку стартапу

п/п	Показники ринку (найменування)	Характеристика
	Кількість основних гравців, од	3

	Обсяг продажів, грн/ум.од	300000
	Тенденції ринку (якісна оцінка)	Зростає
	Обмежень для входу (вказати характер обмежень)	Залучення потенційних клієнтів
	Специфічні вимоги стандартизування та сертифікування	Ліцензія
	Середня норма рентабельності в даній галузі, %	300000/210000 = 143%

У таблиці 5.5 показано характеристику потенційних клієнтів стартап-проекту

Таблиця 5.5. Характеристика потенційних клієнтів стартап-проекту

п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності поведінки потенційних цільових груп клієнтів	Вимоги поживачів до товару
	Стабільний зв’язок під час проведення онлайн операцій	Бізнес, виробництво,фінансовий сектор	Необхідний рівень якості передавання данних	Результат повинен відповідати найвищим стандартам якості
	Якісна передача інформації	Бізнес, виробництво,фінансовий сектор	Кожна група має власні вимоги до стандартів передачі даних	Забезпечення передавання даних залежності від потреб споживача

У табл. 5.6 наведено основні загрози реалізації стартап-проекту.

Таблиця 5.6. Фактори загроз

п/п	Фактор	Опис загрози	Планове реагування компанії

	Недостатній інтерес клієнтів	В випадку невдалого маркетингу клієнта можуть не зацікавити запропоновані послуги	Забезпечення додаткових сервісних послуг
	Втрата конкурентних позицій	Втрата статусу надійного постачальника	Якісний та кількісний приріст інтенсивності та виважена цінова політика

У табл.5.7 наведено основні можливості під час реалізації стартап-проекту.

Таблиця 5.7. Основні можливості

п/п	Фактор	Опис можливості	Планове реагування компанії
	Лідерські позиції на ринку телекомунікаційних послуг	Стрімке зростання попиту	Якісне та кількісне збільшення продукту, якісна підтримка користувача, постійні оновлення безпеки
	Впровадження запропонованих технологій в уже існуючі системи стільникових мереж	Збільшення об’ємів закупівель	Якісне та кількісне збільшення обсягів продукту

У таблиці 5.8 наведено особливості та вплив конкурентного середовища на впровадження проекту [19].

Таблиця 5.8. Аналіз конкуренції

Особливості конкурентного середовища	Прояв даної характеристика	Вплив на діяльність підприємства (планові дії компанії для забезпечення конкурентоспроможності)
1.Конкуренція	Застосування вже існуючих технологій	Проведення стандартизації на високому рівні
2.Локальний	Відсутність єдиного постачальника послуг	Індивідуальний підхід до кожного клієнта та його апаратної частини
3.Міжгалузева	Відсутня	Відсутня
4.Товарно-видова	Використання стандартизованих технологій	Застосування загальновживаних апаратних засобів, за необхідності
5.Цінова	Використання високовартісних спеціалізованих комплексів	Можливість заощадити шляхом застосування загальновживаних апаратних засобів
6.Марочна	Кожна діагностика повинна бути стандартизованою	Здобуття переваги на ринку телекомунікаційних послуг

У таблиці 5.9 проаналізовано конкуренцію проекта в галузі за ринком Іспанії

Таблиця 5.9. Аналіз конкуренції ринку

Складові аналізу	Прямі конкуренти	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	Програмні постачальники	Потреба пошуку постачальників	Залучення відомих та перевірених постачальників	Самостійність у прийнятті клієнтських рішень	Надання переваги авторитетнішим апаратним рішенням

Висновки:	Низька	Є можливість виходу на ринок	Постачальники встановлюють цінову політику на обладнання	Клієнти встановлюють вимоги до якості	Обмеження є тільки в випадку відмови від діагностики

У табл. 5.10 наведено та обґрунтувано фактори конкурентноспроможності.

Таблиця 5.10. Обґрунтування факторів конкурентоспроможності

п/п	Фактор конкурентоспроможності	Обґрунтування (чинники, що роблять фактор порівняння конкурентних проектів значущим)
	Раціональніша цінова політика	Можливість раціональнішого використання ресурсів
	Забезпечення сервісних послуг	Сервісне обслуговування апаратної та програмної частини

У табл. 5.11 перелічено сильні та слабкі сторони проекту.

Таблиця 5.11. Порівняльний аналіз сильних та слабких сторін проекту

п/п	Фактор конкурентоспроможності	Бали 1-20	Порівняння рейтингу товарів-конкурентів						
			3	2	1		1	2	3
	Раціональніша цінова політика	15							
	Послуги сервісного обслуговування	12							
	Періодична діагностика	7							
	Потреба в залученні висококваліфікованих кадрів	7							

У табл.5.12 представлений SWOT-аналіз стартап-проекту.

Таблиця 5.12. SWOT- аналіз стартап-проекту

Сильні сторони: раціональна цінова політика, послуги сервісного обслуговування	Слабкі сторони: періодична діагностика, потреба в залученні висококваліфікованих кадрів
Можливості: Ексклюзивне використання нового методу, впровадження методу в існуючі мережеві логічні комплекси	Загрози:низька зацікавленість клієнтів, втрата монополії

Альтернативи ринкового впровадження стартапу показані в табл.5.13.

Таблиця 5.13. Альтернативи ринкового впровадження проекту

п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність залучення ресурсів	Терміни реалізації
	Складання договорів з телекомунікаційними компаніями та оперативне захоплення ринку при використанні нового рішення	висока	короткі
	Застосування приладів загального вжитку для підвищення конкурентноспроможності	середня	короткі

5.4 Розроблення ринкової стратегії проекту

Обгрунтування вибору цільових груп потенціійних споживачів показано в табл. 5.14.

Таблиця 5.14. Вибір цільових груп потенційних споживачів

п/п	Загальний профіль цільової групи потенційних клієнтів	Готовність сприйняття продукту споживачами	Орієнтовний попит цільової групи (сегменту)	Напруженість конкуренції в сегменті	Складність входу у сегмент
	Оператори стільникового покриття.	Середня	Високий	Низька	Середня
	Приватні мережі	Середня	Середній	Середня	Низька

Визначення базової стратегії розвитку наведено у табл. 5.15.

Таблиця 5.15. Визначення базової стратегії розвитку

п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Основні конкурентоспроможні позиції згідно з обраною альтернативою	Базова стратегія розвитку*
	Застосування альтернативних технологій та пристроїв	Впровадження нового стандарту якості	Залучення ключових гравців у галузі БД	Стратегія диференціації
	Бюджетність проекту	Оптимізованіші затрати на обладнання, та послуги	Використання загальновживаних апаратних рішень замість спеціалізованих комплексів та нового обладнання	Стратегія лідерства по витратах

Визначення основної стратегії конкурентної поведінки показано в табл. 5.16.

Таблиця 5.16. Визначення базової стратегії конкурентної поведінки

п/п	Чи є проект унікальним на ринку?	Чи необхідно буде компанії шукати нових споживачів, чи	Чинеобхідно компанії копіювати основні	Стратегія конкурентної поведінки*

		опрацьовувати існуючих у конкурентів?	характеристики товару конкурента?	
	Так	Опрацьовувати існуючих та шукати нових	Немає необхідності	Стратегія виклику лідера

Визначення стратегії позиціонування показано в табл. 5.17.

Таблиця 5.17. Визначення стратегії позиціонування

п/п	Вимоги цільової аудиторії до товару	Основна стратегія розвитку	Основні конкурентоспроможні позиції стартап-проекту	Визначення асоціацій, які сформують комплексну позицію стартап-проекту (три основних)
	Належна висока якість послуг	Стратегія диференціації	Новизна, гарант якості, точність дослідження	Якість,підтримка, надійність
	Невисокі витрати	Стратегія лідерства по витратах	Універсальність запропонованого рішення	Універсальність, економічна доцільність

5.5 Розроблення маркетингової програми стартап-проекту

Основні переваги концепції потенційного товару показано в табл. 5.18.

Таблиця 5.18. Визначення основних переваг концепції потенційного товару

п/п	Потреба	Вигода, яку пропонує товар	Основні переваги перед конкурентами (існуючі або потенційні)
	Якість	Належна висока якість, надійність	Постійна підтримка користувача, оффлайн підтримка.
	Невисока вартість	Оптимальне використання коштів, не потрібно купувати нове обладнання	Невисока вартість

Виявлено три рівні моделі товару. Зміст та складові рівнів товару показано в табл. 5.19.

Таблиця 5.19. Опис трьох рівнів моделі товару

Рівні товару	Зміст та складові		
I. Товар за задумом	Якісний товар та послуги, стандартизована якість послуг та обладнання		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1)Вартість обслуговування, 2)Кількість комплектів програми 3)Строкбезвідмовної експлуатації 4)Технологічна собівартість товару	1) М 2) М 3) М 4) М	1)Е 2) Пр 3)Нд 4)Тх
	Якість: міжнародні стандарти, постійне обслуговування та підтримка обладнання		
	Доставка, встановлення і налаштування		
	Марка: БД		
III. Товар із підкріпленням	До продажу – програмне забезпечення		
	Після продажу – обслуговування та сервісна підтримка		

Визначення цінової політики на послугу показано в табл. 5.20.

Таблиця 5.20. Визначення меж встановлення ціни

п/п	Цінова політика товарів-замінників	Цінова політика на товари-аналоги	Рівень купівельної спроможності цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
	5000 у.о./од. (стандартна методика)	-	Високий	Н.4000у.о.– В.4500у.о. (Товар)

				Н.100 у.о. – В.500у.о. (Послуга)

Створення системи збуту послуги вказано у табл. 5.21.

Таблиця 5.21. Створення системи збуту

п/п	Закупівельна поведінка цільових клієнтів	Функції збуту, що повинен забезпечувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
	Орієнтованана максимальний дохід від існуючого обладнання та вкладених коштів	Поставки якісного, точного та надійного товару	Значна	Договірна система збуту

Концепції маркетингових комунікацій показано в табл. 5.22.

Таблиця 5.22. Концепція маркетингових комунікацій

п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій цільових клієнтів	Основні методи позиціонування	Завдання рекламного звернення	Концепція рекламного звернення
	Зацікавленість в точному та якісному продукті з раціональним використанням ресурсів	Мережні ресурси	Гарантія якості та стандартизація, сервісна політика	Привернути увагу до покращень, пов’язаних із зростаючою популярністю послуг	Позиціонування центру синхронізації відправною точкою на шляху до над якісного контенту

	Зацікавленість у великих об’ємах продукції із дотриманням умов якості	Мережні ресурси	Глибина каналу постачальників, гарант якості	Привернути увагу до переваг первісності та в глибині каналу постачання	Позиціонування послуг центру синхронізації єдиним раціональним шляхом забезпечення стабільного трафіку

Висновки до розділу

1. Встановлено, що комерціалізацію стартап-проекту щодо застосування та розвитку запропонованого програмного рішення для розгортки БД на основі існуючих стільникових мереж можна вважати доцільною. На ринку телекомунікаційних послуг у світі існує суттєвий попит на дану пропозицію, який зараз задовільняють товари замінники та більш дорогі рішення. Рентабельність на ринку послуг забезпечить в першу чергу можливість впровадження нових мереж на основі існуючої апаратної частини, й як наслідок економічну доцільність та універсальність.

2. Можливість виходу на ринок є дуже високою, адже існуючим операторам вигідно отримувати максимум доходу від вже придбаного обладнання, водночас з тим сфера БД є дуже перспективною та високодохідною в майбутньому. Конкурентні спроможності проекту реалізовано внаслідок можливості зайняти порожню нішу в Україні та надати гарний рівень підтримки продукту. Це є перевагою і основним критерієм входження на ринок запропонованого рішення.

3. Обрана альтернатива впровадження – пошук альтернативних технологій та пристроїв для побудови систем БД. Позитивні умови для

просування проекту зумовлені вимогами ринку, а саме прагнення багатьох великих міст до втілення концепту «розумного міста».

ВИСНОВКИ

1. В сучасних БД і ЕС досить успішно вирішуються завдання захисту конфіденційних даних від несанкціонованого доступу, забезпечення цілісності та доступності даних. Забезпечення доступності даних на фізичному рівні досягається шляхом використання стійких до відмов пристроїв зберігання даних, наприклад, кількох жорстких дисків, об’єднаних в масив RAID.

2. Цілісність БД не гарантує достовірності, що міститься в ній інформації, але забезпечує принаймні правдоподібність цієї інформації, відкидаючи свідомо неймовірні, неможливі значення.

3. Посилальна цілісність забезпечує підтримку несуперечливого стану БД (узгодженого стану зовнішніх ключів) в процесі модифікації даних, а також при виконанні операцій додавання або видалення записів.

4. Віртуальні таблиці - це іменований запит, що зберігається в базі даних. Віртуальні таблиці не вимагають для свого зберігання дискової пам'яті, за винятком пам'яті, необхідної для зберігання визначення самих таблиць. Віртуальні таблиці не можуть існувати самі по собі, а визначаються тільки в термінах однієї або декількох таблиць.

5. Тригери використовуються для перевірки цілісності даних, а також для відкату транзакцій. Таким чином, тригер – це відкомпільована SQL-процедура, виконання якої обумовлено настанням певних подій всередині реляційної бази даних.

6. Шифрування бази даних - це найпростіший спосіб захисту. При шифруванні бази даних її файл стискається і стає недоступним для читання за допомогою службових програм або текстових редакторів.

7. Система безпеки MS SQL Sever реалізує дискреційний і рольовий підходи до управління доступом і має два рівні: рівень сервера і рівень бази даних.

8. Ідентифікатор безпеки Windows передається з операційної системи на сервер баз даних. SQL Server передбачає, що процес реєстрації

користувачів в мережі достатньо захищений, і тому не виконує ніяких додаткових перевірок.

В результаті роботи було отримано такі результати:

- виконано огляд сучасних технологій захисту корпоративнх мереж та методів впливу на них зловмисників;
- досліджено доцільність використання кожного із розглянутих методів за конкретних умов;
- представлено результати дослідження у вигляді, зручному для кінцевого користувача.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Смирнов, С.Н. Безпека систем баз даних / С.Н. Смирнов. – М.: Гелиос-АРВ, 2007. – 352с.

2. Цирлов, В. Л. Основи інформаційної безпеки. короткий курс / В. Л. Цирлов. – Феникс, 2008. - 256 с.

3. Герасименко В.А., Малюк А.А. Основи захисту інформації. М .: МІФІ, 2001 г.

4. Neerja Bhatnagar. Security in Relational databases / N. Bhatnagar // - In: Handbook of Information and Communication Security. ed. by P. Stavroulakis, M. Stamp.- Springer. – 2010. - pp. 257 - 272.

5. Pernul, G. Database Security / G. Pernul // In: Advances in Computers, Vol. 38. ed. by M. C. Yovits. - Academic Press. – 1994. - pp. 1 - 74.

6. Хорев П.Б. Програмно-апаратний захист інформації: навчальний посібник/ П.Б. Хорев. - М.: Форум, 2013. - 352 с.

7. Бєлов О.Б. [та ін]. Основи інформаційної безпеки: навч-ве посібник для студентів вузів, що навч. по спец. в обл. інформ. безпеки / Є.Б. Бєлов - М: Гаряча лінія - Телеком, 2006. - 544 с.

8. Барсуков В.С. Сучасні технології безпеки: Інтегральний підхід. / В.С. Барсуков, Водолазький В.В. - М: Нолидж, 2000. - 496 с.

9. Романець Ю.В. Захист інформації в комп'ютерних системах і мережах / Ю.В. Романець, П.А. Тимофєєв, В.Ф. Шаньгина. - М: Радіо і зв'язок, 1999. - 328 с.

10. Голіков А.М. Основи інформаційної безпеки: навчальнийпосібник / А.М. Голіков. - Томськ: Томськ. держ. ун-т систем упр. і радіоелектроніки, 2007. - 288 с.

11. Захист інформації. Основні терміни та визначення. ГОСТ Р 50922 - 2006. - М: Стандартинформ, 2008.

12. Захист інформації. Забезпечення інформаційної безпеки в організації. Основні терміни та визначення. ГОСТ Р 53114-2008. - М: Стандартинформ 2009.

13. Міхєєва, В. Microsoft Access 2003 / В. Міхєєва і І. Харитонова. - СПб: БХВ, 2004. - тисяча сімдесят два с.

14. Нільсен, Пол. SQL Server 2005. Біблія користувача. Пер з англ. / Пол Нільсен. - Москва, СПб, Київ: Вільямс (Діалектика), 2008.- 1232 с.

15. Мамаєв, І. Microsoft SQL Server 2000. Найбільш повне керівництво / Є. Мамаєв - СПб: БХВ, 2005. - 1280 с.

16. НД ТЗІ 2.5-004-99 «Критерії оцінки захищеності інформації в комп’ютерних системах від несанкціонованого доступу».

Додаток А

РЕФЕРАТ

АНГЛІЙСЬКОЮ МОВОЮ ЗА ТЕМОЮ ДИПЛОМНОЇ РОБОТИ

In modern conditions, the activities of any organization are associated with the operation of a large amount of information, access to which has a wide range of people. In such circumstances, malicious or simply incompetent actions of only one of the employees can cause irreparable damage to the organization as a whole. It may not even be about stealing valuable information, it is enough to simply block access to an important information resource for a long time.

Database management systems (DBMS), especially relational databases and expert systems (ES), have become the dominant tool in the field of data storage, processing and presentation. Any failure of the DBMS (ES), accompanied by loss of access to data, immediately affects the competitiveness of the enterprise. Therefore, the protection of data from unauthorized access, from unauthorized modification or simply from their destruction is one of the priorities in the design of any information system. This data protection problem covers both the physical protection of system program data and the protection against unauthorized access to data transmitted over communication lines and stored on drives, which is the result of the activities of both third parties and special virus programs. If we take into account that the core of the information system is a database, the provision of information security of the latter is crucial in choosing specific means of ensuring the required level of security of the organization as a whole.