МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
комп'ютерних систем та мереж

_____ Ігор ЖУКОВ
"_____"_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВР

Тема: «Система управління мережевою базою даних компанії»

Виконавець: _____ Євгеній ХАРЧЕНКО
(підпис)

Керівник: _____ Василь МАЛЯРЧУК
(підпис)

Нормоконтролер: _____ Сергій ЖУРАВЕЛЬ
(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет комп’ютерних інформаційних технологій
Кафедра комп’ютерних систем та мереж
Спеціальність 123 «Комп’ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп’ютерних систем та мереж

Ігор ЖУКОВ
“_____”_____2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Харченка Євгенія Олександровича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема кваліфікаційної роботи: «Система управління мережевою базою даних компанії»
затверджена наказом ректора від «26» квітня 2023 р. № 591/ст
2. Термін виконання роботи: з 22.05.2023 р. по 25.06.2023 р.
3. Вихідні дані до роботи: аналіз ефективності та продуктивності системи за допомогою відповідних метрик та інструментів; вивчення та аналіз існуючих систем управління базами даних та їх функціональності; використання програмних інструментів для розробки та реалізації системи; тестування та валідація розробленої систем
4. Зміст пояснювальної записки: аналіз функціональності та ефективності сучасних мультимедійних систем у контексті мережевого середовища; забезпечення захисту інформації в системах та мережах; архітектура системи безпеки SQL Server
5. Перелік обов'язкового графічного (ілюстративного) матеріалу: слайди презентації в програмному пакеті Microsoft PowerPoint

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Розробити деталізований зміст розділів кваліфікаційної роботи	22.05.2023-24.05.2023	Виконано
2	Вступ	25.05.2023	Виконано
3	АНАЛІЗ ФУНКЦІОНАЛЬНОСТІ ТА ЕФЕКТИВНОСТІ СУЧАСНИХ МУЛЬТИМЕДІЙНИХ СИСТЕМ У КОНТЕКСТІ МЕРЕЖЕВОГО СЕРЕДОВИЩА	26.05.2023-29.05.2023	Виконано
4	ЗАБЕЗПЕЧЕННЯ ЗАХИСТУ ІНФОРМАЦІЇ В СИСТЕМАХ ТА МЕРЕЖАХ	30.05.2023-07.06.2023	Виконано
5	АРХІТЕКТУРА СИСТЕМИ БЕЗПЕКИ SQL SERVER	08.06.2023-14.06.2023	Виконано
6	Усунення недоліків та захист кваліфікаційної роботи	15.06.2023-16.06.2023	Виконано

7. Дата видачі завдання: “22" травня 2023 р.

Керівник кваліфікаційної роботи _____ Василь МАЛЯРЧУК
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Євгеній ХАРЧЕНКО
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Кваліфікаційна робота «Система управління мережевою базою даних компанії» містить 69 сторінок, 13 рисунків, 0 таблиць, 18 використаних джерел.

БАЗА ДАНИХ, ФУНКЦІОНАЛЬНІСТЬ, ЗАХИСТ, АТАКА, ЕФЕКТИВНІСТЬ, АРХІТЕКТУРА, МОНІТОРИНГ, АУТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, ШИФРУВАННЯ, CLOUD-ТЕХНОЛОГІЇ, СИСТЕМА УПРАВЛІННЯ.

Об’єкт дослідження – система управління мережевою базою даних компанії.

Предмет дослідження – аналіз функціональності та ефективності системи управління базою даних, що працює у мережевому середовищі компанії.

Мета кваліфікаційної роботи – дослідження та аналіз системи управління мережевою базою даних компанії.

Метод дослідження – збір та аналіз статистичних даних про продуктивність, завантаження та швидкодію системи управління мережевою базою даних.

Матеріали кваліфікаційної роботи рекомендується використовувати при розробці та налаштуванні ефективної мережевої бази даних з урахуванням сучасних технологій та принципів безпеки. Це допоможе компанії ефективно управляти своїми даними, забезпечити швидкий доступ до інформації та зменшити можливість виникнення помилок.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

SQL	-	Structured Query Language
IaaS	-	Infrastructure as a Service
PaaS	-	Platform as a Service
SaaS	-	Software as a Service
AWS	-	Amazon Web Services
GCP	-	Google Cloud Platform
VM	-	Virtual Machine
API	-	Application Programming Interface
TDE	-	Transparent Data Encryption
TCP	-	Transmission Control Protocol
UDP	-	User Datagram Protocol
TDS	-	Tabular Data Stream
MAC	-	Media Access Control
RBAC	-	Role Based Access Control
ПК	-	Портативний Комп’ютер
БД	-	База Даних
НСД	-	Несанкціонований доступ
СУБД	-	Системи Управління Базами Даних
СКБД	-	Системи Керування Базами Даних

ВСТУП

Актуальність теми. У сучасному світі, де обмін і обробка інформації є необхідністю для бізнесу, ефективне управління даними є критично важливим завданням для будь-якої компанії. Збільшення обсягу даних, швидкість їхнього збору та обробки, а також розподілена природа роботи з даними ставлять перед компаніями складні завдання щодо управління мережевими базами даних.

Система управління мережевою базою даних є інструментом, який допомагає компаніям зберігати, організовувати, доступати та аналізувати великі обсяги даних, які зберігаються на різних вузлах мережі. Це особливо важливо в контексті компаній з глобальними мережами або деякими розподіленими системами, де даними обмінюються між різними вузлами або підрозділами компанії.

Актуальність теми полягає в тому, що компанії стикаються з рядом викликів щодо ефективного управління мережевими базами даних. Важливо мати надійну систему, яка забезпечує швидкий доступ до даних, захист інформації, масштабованість і можливість розширення, а також забезпечує цілісність та консистентність даних в розподілених середовищах.

Крім того, з поширенням хмарних технологій і збільшенням кількості даних, які зберігаються в хмарних середовищах, управління мережевими базами даних стає ще більш актуальним. Компаніям потрібні ефективні та безпечні рішення для управління даними, що знаходяться в хмарі.

Таким чином, вивчення систем управління мережевою базою даних є важливим напрямом досліджень, оскільки вона допомагає компаніям ефективно управляти своїми даними, забезпечувати доступ до них, захищати інформацію та забезпечувати надійність роботи мережевих систем.

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

Мета і завдання дослідження. Метою даної дипломної роботи є дослідження та аналіз системи управління мережевою базою даних компанії. Завданнями даного дослідження є:

1. Проаналізувати сучасні тенденції управління мережевими базами даних і визначити основні вимоги до системи управління мережевою базою даних компанії.

2. Вивчити функціональні можливості системи управління мережевою базою даних компанії, включаючи можливості зберігання, організації, доступу та обробки даних.

3. Вивчити аспекти безпеки і захисту даних у системі управління мережевою базою даних компанії, включаючи заходи з аутентифікації, авторизації, шифрування та аудиту доступу до даних.

4. Розробити рекомендації щодо оптимального використання та покращення системи управління мережевою базою даних компанії, враховуючи виявлені переваги, недоліки та вимоги компанії.

Дослідження цих завдань дозволить отримати глибоке розуміння системи управління мережевою базою даних компанії, її впливу на бізнес-процеси та розробити рекомендації для оптимального використання цієї системи.

Для досягнення поставленої мети вирішуються такі наукові завдання.

1.

Об'єктом дослідження – є система управління мережевою базою даних компанії.

Предметом дослідження – є аналіз функціональності та ефективності системи управління базою даних, що працює у мережевому середовищі компанії.

Метод дослідження – збір та аналіз статистичних даних про продуктивність, завантаження та швидкодію системи управління мережевою базою даних.

Практичне значення отриманих результатів.

Результати дослідження дозволяють компанії розробити та налаштувати ефективну мережеву базу даних з урахуванням сучасних технологій та принципів безпеки. Це допоможе компанії ефективно управляти своїми даними, забезпечити швидкий доступ до інформації та зменшити можливість виникнення помилок.

РОЗДІЛ 1
АНАЛІЗ ФУНКЦІОНАЛЬНОСТІ ТА ЕФЕКТИВНОСТІ СУЧАСНИХ МУЛЬТИМЕДІЙНИХ СИСТЕМ У КОНТЕКСТІ МЕРЕЖЕВОГО СЕРЕДОВИЩА

1.1. Аналіз Cloud-технологій та засобів моніторингу в хмарних середовищах

Cloud-технології представляють собою інноваційний підхід до розгортання, керування та використання різноманітних обчислювальних ресурсів, які надаються через мережу Інтернет. Вони дозволяють організаціям зберігати дані, розгортати програмне забезпечення та надавати послуги без необхідності володіння та управління фізичною інфраструктурою. У цьому розділі ми розглянемо основні концепції Cloud-технологій та їх моніторингу. У Cloud-технологіях існує кілька моделей хмарних сервісів, що визначають рівень відповідальності та контролю, який користувачі мають над обчислювальними ресурсами. Основні моделі хмарних сервісів включають:

- *Інфраструктуру як сервіс (Infrastructure as a Service, IaaS)* - ця модель надає користувачам доступ до віртуальних обчислювальних ресурсів, таких як віртуальні машини, сховища даних та мережі. Користувачі мають контроль над операційними системами, програмним забезпеченням та застосунками, які вони розгортають у хмарному середовищі.
- *Платформу як сервіс (Platform as a Service, PaaS)* - ця модель надає користувачам середовище для розробки, тестування та розгортання власних додатків. Користувачі отримують доступ до платформи, яка включає операційну систему, середовище виконання та інструментарій для розробки. Інфраструктурні деталі приховані від користувачів.

- *Програмне забезпечення як сервіс (Software as a Service, SaaS)* - ця модель надає користувачам готові застосунки, які доступні через Інтернет. Користувачі не мають контролю над інфраструктурою, але можуть використовувати програмне забезпечення відповідно до своїх потреб.

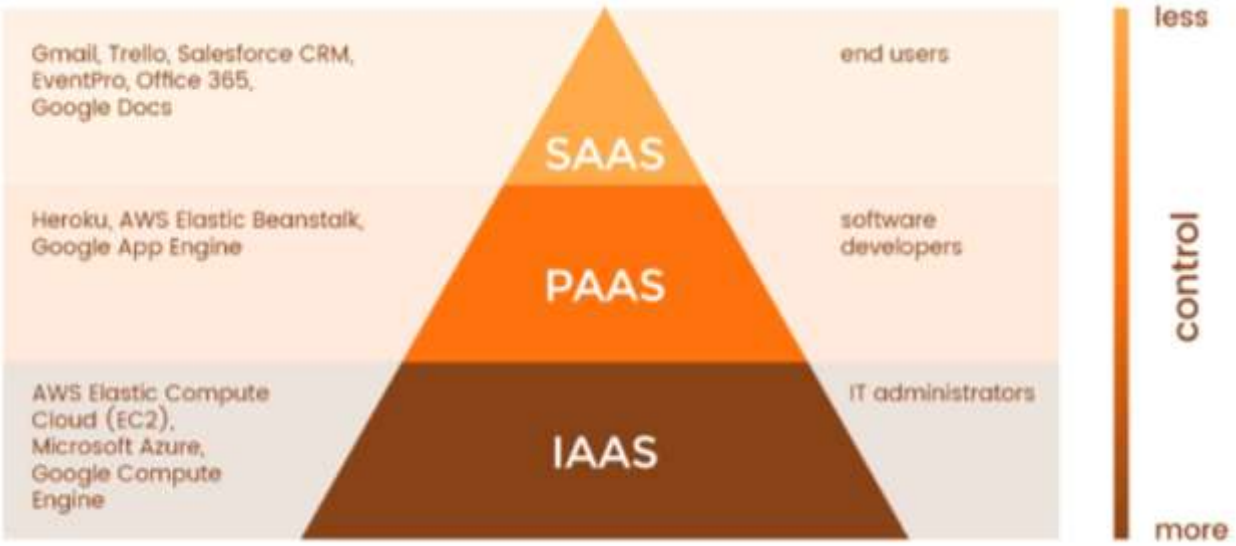


Рис. 1.1. Модель хмарних сервісів

Крім цього, у Cloud-технологіях використовуються різні типи хмарних середовищ, такі як публічні, приватні та гібридні хмари. Публічні хмари надають доступ до обчислювальних ресурсів широкому колу користувачів через мережу. Приватні хмари використовуються в межах конкретної організації та забезпечують більшу контрольованість та безпеку. Гібридні хмари поєднують публічні та приватні хмари, що дозволяє організаціям комбінувати переваги обох типів середовищ. У Cloud-технологіях існує специфічна архітектура та компоненти, які забезпечують роботу хмарного середовища. Деякі з них включають:

- *Облікові записи та автентифікація*. В цьому компоненті здійснюється управління обліковими записами користувачів та їх автентифікація при доступі до хмарних сервісів.
- *Інфраструктура для зберігання даних*. Цей компонент забезпечує зберігання даних у хмарних сервісах. Він може включати розподілені сховища даних, бази даних та інші механізми для забезпечення надійного зберігання та доступу до даних.

- *Мережеві ресурси*. Цей компонент включає мережеві ресурси, такі як мережеві сегменти, комутатори та маршрутизатори, які забезпечують комунікацію між різними складовими хмарного середовища.

Моніторинг хмарного середовища є важливою складовою для забезпечення його ефективності, безпеки та доступності.

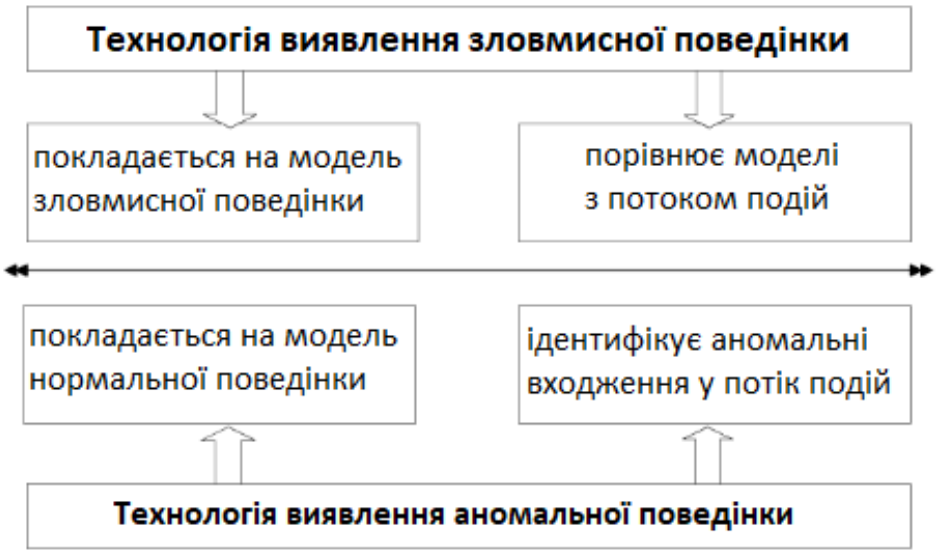


Рис. 1.2. Технологія виявлення зловмисної поведінки

При моніторингу хмарних сервісів можуть використовуватись різноманітні інструменти та підходи, такі як:

- *Моніторинг доступності та продуктивності.* це включає моніторинг ресурсів хмарного середовища, таких як обчислювальна потужність, мережевий трафік та сховища даних, щоб забезпечити їх належну доступність та продуктивність для користувачів.
- *Моніторинг безпеки.* цей аспект моніторингу спрямований на виявлення та реагування на можливі загрози та вразливості в хмарному середовищі. Він може включати виявлення аномальної активності, контроль за доступом до ресурсів та моніторинг логів подій.
- *Моніторинг витрат.* цей аспект моніторингу спрямований на відстеження та оптимізацію витрат на використання хмарних ресурсів.

Він дозволяє оцінювати ефективність витрачання ресурсів та розробляти стратегії для економії коштів.

Для моніторингу Cloud-технологій існує широкий спектр інструментів, які можна використовувати. Деякі з популярних інструментів включають:

1. *Amazon CloudWatch* є однією з провідних послуг моніторингу в хмарних обчисленнях. Це повнофункціональна система моніторингу та спостереження, яка дозволяє відстежувати та збирати дані про різні ресурси та послуги, що працюють на Amazon Web Services (AWS).

Рис. 1.3. Інструмент Amazon CloudWatch

Основна мета Amazon CloudWatch полягає в наданні користувачам цілісного уявлення про стан їх інфраструктури в хмарному середовищі. Вона забезпечує моніторинг обчислювальних ресурсів, мережевих взаємозв'язків, зберігання даних та інших послуг AWS, а також дозволяє збирати власні метрики, створювати сповіщення та налаштовувати автоматичні реакції на події. Amazon CloudWatch пропонує розширені можливості моніторингу, такі як метрики продуктивності, журнали подій, трасування запитів та аналіз реакції системи. Користувачі можуть відстежувати метрики ресурсів, налаштовувати сповіщення про перевищення порогових значень та створювати власні налаштування для моніторингу та аналізу. Amazon CloudWatch дозволяє організаціям ефективно керувати та оптимізувати використання ресурсів, виявляти проблеми та приймати вчасні заходи для їх вирішення. Вона є невід'ємною

частиною процесу моніторингу та управління хмарною інфраструктурою на платформі AWS.

2. *Google Cloud Monitoring* є рішенням для моніторингу та управління інфраструктурою в середовищі Google Cloud Platform (GCP). Ця послуга дозволяє користувачам відстежувати, аналізувати та спостерігати за різними аспектами їх інфраструктури, додатків та послуг, які працюють на платформі GCP.

Рис. 1.4. Інструмент Google Cloud Monitoring

Основна мета Google Cloud Monitoring полягає в забезпеченні оперативного контролю та виявленні проблем в реальному часі. Вона надає розширені можливості збору метрик та журналів, а також дозволяє створювати сповіщення та налаштовувати автоматичні реакції на випадки виникнення проблем або перевищення порогових значень. Google Cloud Monitoring дозволяє відстежувати метрики продуктивності ресурсів, мережеві взаємозв'язки, роботу додатків, стан здоров'я системи та багато іншого. Користувачі можуть аналізувати дані за допомогою гнучких запитів та візуалізувати їх у зручних графіках та звітах. Google Cloud Monitoring також пропонує розширені функції аналізу, такі як деталізація подій, відстежування трендів та прогнозування майбутньої продуктивності. Це допомагає користувачам розуміти залежності та зрозуміти, як впливають різні фактори на їхню інфраструктуру.

Google Cloud Monitoring дозволяє організаціям забезпечувати надійну та ефективну роботу їх інфраструктури, виявляти проблеми та реагувати на них швидко

та ефективно. Вона є незамінною частиною процесу моніторингу та управління інфраструктурою в середовищі Google Cloud Platform.

3. *Microsoft Azure Monitor* є послугою моніторингу та аналізу від Microsoft для хмарної платформи Azure. Вона надає засоби для відстежування, аналізу та візуалізації метрик, журналів та діагностичних даних в реальному часі.

Рис. 1.5. Microsoft Azure Monitor

Основна мета Microsoft Azure Monitor полягає в наданні інсайтів щодо продуктивності, доступності та безпеки інфраструктури та додатків, що працюють на платформі Azure. Вона забезпечує оперативний контроль за різними складовими Azure, такими як віртуальні машини, служби, контейнери, бази даних та інші ресурси. Microsoft Azure Monitor збирає та агрегує метрики з різних джерел, що дозволяє користувачам відстежувати продуктивність та виявляти аномалії. Вона також забезпечує можливості розширеного аналізу даних, такі як кореляція подій, прогнозування майбутньої продуктивності та виявлення аномалій на основі штучного інтелекту. Крім того, Microsoft Azure Monitor дозволяє створювати сповіщення та налаштовувати автоматичні реакції на події, що стосуються продуктивності та доступності інфраструктури. Користувачі можуть налаштовувати правила сповіщень, які будуть викликатися при перевищенні певних порогових значень метрик або виникненні певних подій.

Microsoft Azure Monitor є потужним інструментом для моніторингу та управління інфраструктурою в середовищі Azure. Вона допомагає забезпечити

належну продуктивність, доступність та безпеку системи, а також реагувати на проблеми та вдосконалювати її ефективність.

Отже, вище перелічені інструменти надають функціональність для моніторингу різних аспектів хмарного середовища, таких як метрики продуктивності, логи подій та безпека.

1.2. Microsoft Windows Azure

Microsoft (Windows) Azure - це назва хмарної платформи від компанії Microsoft. Платформа Windows Azure надає можливість розробляти та виконувати додатки і зберігати дані на серверах, що розташовані в розподілених центрах обробки даних. У 2014 році платформа була перейменована в Microsoft Azure [1].

Microsoft Azure повністю реалізує дві моделі хмари - платформи як сервісу (Platform as a Service, PaaS) та інфраструктури як сервісу (Infrastructure as a Service, IaaS) [2]. Роботу платформи Windows Azure забезпечує глобальна мережа дата-центрів Microsoft.

Основні особливості цієї моделі включають:

а) оплата лише за використані ресурси;

б) загальна, багатопотокова структура обчислень;

в) абстракція від інфраструктури.

Основою роботи Microsoft Azure є запуск віртуальної машини для кожного екземпляра додатку. Розробник визначає потрібний обсяг для зберігання даних та необхідні обчислювальні потужності (кількість віртуальних машин), після чого платформа надає відповідні ресурси. Коли початкові потреби в ресурсах змінюються згідно з новим запитом замовника, платформа надає додаткові ресурси для додатку або звільняє не використовувані ресурси дата-центра.

Microsoft Azure, як PaaS, забезпечує не тільки всі базові функції операційної системи, але й додаткові можливості, такі як виділення ресурсів за запитом для необмеженого масштабування, автоматична синхронна реплікація даних для

підвищення стійкості до відмов, обробка відмов інфраструктури для забезпечення постійної доступності та багато іншого [3].

Microsoft Azure також реалізує інший тип сервісу - інфраструктуру як сервіс. Модель надання інфраструктури (апаратних ресурсів) надає можливість орендувати такі ресурси, як сервери, пристрої зберігання даних та мережеве обладнання. Управління всією інфраструктурою здійснюється постачальником, а споживач керує лише операційною системою та встановленими додатками. Такі сервіси оплачуються залежно від фактичного використання та дозволяють збільшувати або зменшувати обсяг інфраструктури через спеціальний портал, наданий постачальниками. Ці сервісні моделі дозволяють запускати практично будь-які додатки, встановлені на стандартні образи ОС.

В доступній галереї образів доступні образи наступних операційних систем: Windows Server, OpenSUSE, CentOS, Ubuntu, SUSE.

У 2013 році було представлено нове сховище зразків віртуальних машин - VM Depot. VM Depot - це проект для спільноти Windows Azure, запущений командою Microsoft OpenTechnologies, Inc, відповідальною за відкриті технології всередині Microsoft [4]. Вміст порталу, а також налаштовані віртуальні машини для різних завдань, будуть створюватись і публікуватись зусиллями спільноти.

Microsoft Azure складається з:

a) Compute - компонент, що реалізує обчислення на платформі Windows Azure.

b) Storage - компонент сховища, яке надає масштабоване сховище. Сховище не має можливості використовувати реляційну модель і є альтернативною, "хмарною" версією SQL Server.

c) Fabric - Windows Azure Fabric, у свою чергу, є "контролером" і ядром платформи, забезпечуючи функції моніторингу в реальному часі, стійкості до відмов, виділення ресурсів, розгортання серверів, віртуальних машин та додатків, балансування навантаження та керування обладнанням.

Практично всі сервіси Microsoft Azure мають API, побудоване на основі REST, що дозволяє розробникам використовувати "хмарні" сервіси з будь-якої операційної системи, пристрою і платформи [5].

Microsoft Azure була визнана Compuware найшвидшою "хмарною" платформою. Також бенчмарк Microsoft Azure продемонстрував високу продуктивність для масштабних обчислень з результатами 151,3 ТФлопс на 8064 ядрах з 90,2-відсотковою ефективністю. Аналітична компанія Nasuni представила нове дослідження провайдерів "хмарних" сервісів зберігання даних. Згідно з цим звітом, платформа Microsoft Azure є лідером у тестах продуктивності при записі та читанні даних з хмари, доступності даних і мінімальної кількості помилок (0%). У 2014 році Nasuni випустила новий звіт з результатами тестування "хмарних" сховищ Amazon, Google, HP, Microsoft і Rackspace.

1.3. Криптографічні сервіси та захист даних у Windows Azure

Windows Azure SDK розширює базові бібліотеки .NET, щоб розробники могли інтегрувати та використовувати сервіси, які надає Windows Azure. Доступ до служб захисту конфіденційності в проектах та сервісах Windows Azure не обмежений. З існуючими збірками, з якими розробники вже знайомі, більша частина роботи, пов'язаної з шифруванням і розшифруванням даних, залишається незмінною. Однак, в нижчезазначеній архітектурі відбулися зміни, пов'язані з тим, коли дані шифруються та зберігаються ключі [6].

1.3.1 Сховище ключів

При будь-якій стратегії шифрування на рівні додатка або підприємства, інфраструктура шифрування і дешифрування становить лише частину рішення. Основна проблема полягає в зберіганні ключів та тривалості їх збереження. Надійність захисту зашифрованих даних залежить від використовуваних ключів, і ця проблема є значно складнішою, ніж можна подумати на перший погляд [7]. При зберіганні ключів у хмарному середовищі виникає важливе питання: де слід зберігати

ці ключі? Деякі люди висловлюють занепокоєння тим, що зберігання ключів у хмарі ставить під загрозу безпеку, яка походить саме з хмарного середовища. Іншими словами, хтось може отримати фізичний доступ до ваших даних, які за замовчуванням зберігаються на диску у незашифрованому вигляді (наприклад, у випадку з Windows Azure). Перше, на що варто звернути увагу, - неможливо використовувати будь-які ключі, надані Windows Azure, в якості шифрувальних ключів у будь-якому додатку. Наприклад, ключі, які надає Windows Azure для сервісу зберігання, налаштовані так, щоб їх можна було легко змінювати для більшої безпеки або у разі компрометації. Іншими словами, ніхто не гарантує, що вони будуть існувати у майбутньому [8]. Власна бібліотека ключів у Windows Azure Storage є хорошим способом збереження конфіденційної інформації, оскільки можна покладатися на ці дані як на надійні в багатокористувацькому середовищі і захищати їх за допомогою власних ключів зберігання. Це відрізняється від використання ключів зберігання як шифрувальних ключів. Замість цього можна використовувати ключі сервісу зберігання для доступу до бібліотеки ключів, як це робили б з будь-якого іншого збереженого файлу. Реалізація такого варіанту є досить простою. Процес створення ключів показано на рисунку 1.6.

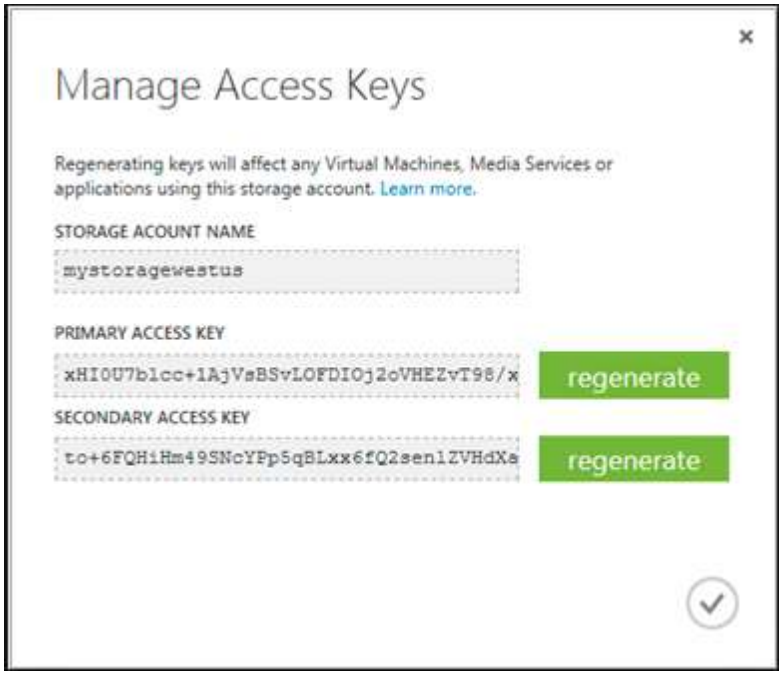


Рис. 1.6 – Створення ключів у Microsoft Azure

1.3.2 Шифрування в SQL Azure

У SQL Server 2008 було введено новий інструмент - прозоре шифрування даних (Transparent Data Encryption, TDE). Вперше в SQL Server стало можливим повністю шифрувати дані з мінімальними зусиллями з вашого боку порівняно з обмеженим шифруванням, яке підтримувалося в SQL Server 2005. Однак початкова версія хранилища SQL Azure наразі не підтримує шифрування на рівні бази даних, хоча така підтримка ймовірно буде включена у наступну версію. Слід зауважити, що в даний час SQL Azure доступний лише через TCP-з'єднання і лише через порт 1433 [9].

Хоча цей інструмент наразі не інтегрований у Windows Azure, існують кілька інших функцій захисту SQL Azure, про які повинні знати розробники або проектувальники. По-перше, SQL Azure підтримує потік табличних даних (tabular data stream, TDS). Це означає, що в основному ви можете підключатись до бази даних і взаємодіяти з нею так само, як завжди. Варто розглянути використання переваг шифрування ADO.NET і сертифікатів довірених серверів, особливо якщо ви звертаєтесь до бази даних SQL Azure з-за хмари. Властивості підключення Encrypt = True і Trust Server Certificate = False в правильній комбінації забезпечать захист передаваних даних і допоможуть запобігти атакам з посередниками (man-in-the-middle attacks). Крім того, це обов'язково для підключення до SQL Azure - неможливо підключитись до SQL Azure, поки ви не увімкнете шифрування на рівні підключення.

Другий інструмент захисту SQL Azure, з яким варто ознайомитись, - брандмауер SQL Azure. Він добре відомий тим, хто користувався локальними брандмауерами або набором інструментів захисту SQL Server. Він дозволяє дозволяє або забороняє з'єднання з різних джерел, включаючи конкретні IP-адреси або діапазони. Брандмауером SQL Azure можна керувати через портал SQL Azure або безпосередньо в головній базі даних за допомогою збережених процедур, таких як sp_set_firewall_rule і sp_delete_firewall_rule [10]. Додавання IP-адреси для брандмауера показано на зображенні 1.7.

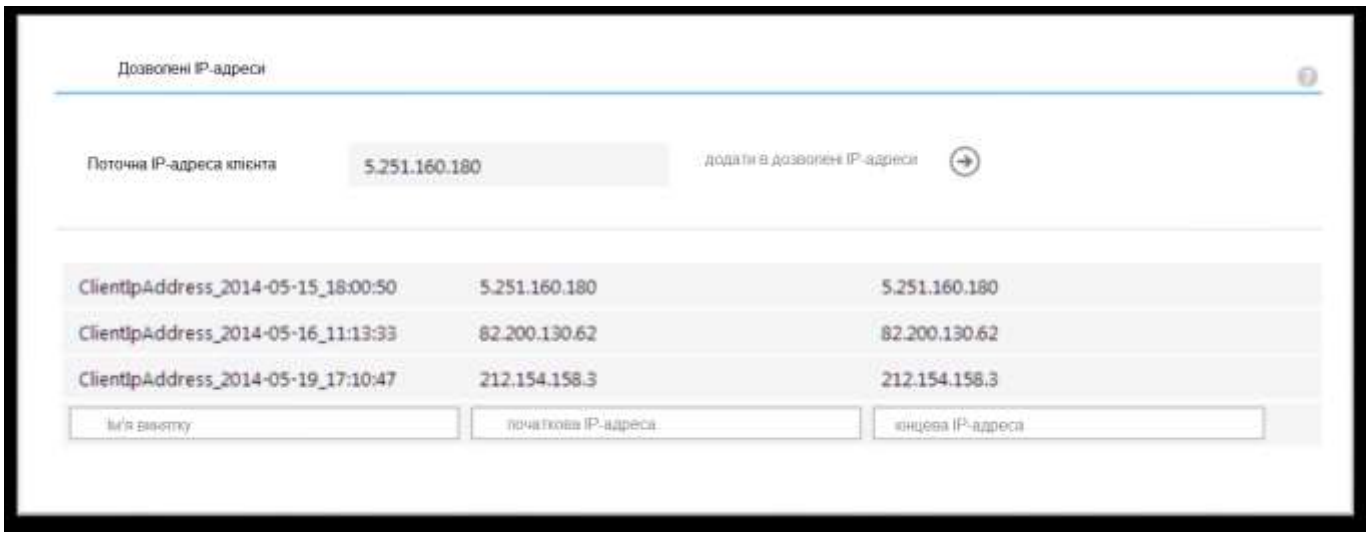


Рис. 1.7 – Додавання IP-адреси

Як і в будь-якій реалізації SQL Server, керування користувальницькими обліковими записами - ще один аспект, який потрібно жорстко контролювати. Брандмауер в SQL Azure - дійсно чудовий інструмент, але не варто покладатись тільки на нього. При використанні, потрібно застосовувати стійкі паролі до облікових записів і налаштовувати для них чітко визначені права доступу, а також ретельно продумати свою модель захисту даних. Завдяки цим новим засобам, SQL Azure є високозахищеною керованою платформою для хмарних додатків. Якщо пробувати вперше працювати з цим сервісом, не треба забувати, що перед спробами підключення, потрібно спочатку налаштувати брандмауер SQL Azure. Перший раз це робиться лише через веб-портал SQL Azure, але в подальшому ним можна керувати за допомогою головної бази даних, як вже було зазначено.

1.4. Опис Azure SQL

Windows Azure SQL Databases (початково відома як SQL Server Data Services, потім як SQL Services, а пізніше як Windows Azure SQL Databases) - це хмарний сервіс від компанії Microsoft, що надає можливість зберігання та обробки реляційних даних, а також генерації звітності. Він надає функціональність для різних сценаріїв синхронізації даних (локальна інфраструктура <=> хмара, хмара <=> хмара). Він є частиною Windows Azure. Windows Azure SQL Databases базується на Microsoft SQL

Server, але надає лише підмножину типів даних. Підтримуються основні типи, такі як точні і наближені числа, символьні рядки (включаючи Юнікод), дата і час, просторові, двійкові та інші типи даних. Використовується формат на основі XML для передачі даних. Так само, як і Microsoft SQL Server, Windows Azure SQL Databases використовує T-SQL як мову запитів. Протокол передачі даних Tabular Data Stream (TDS) використовується для доступу до сервісу через Інтернет. Доступ через протокол HTTP REST не надається. Microsoft рекомендує використовувати ADO.NET Data Services для передачі даних та створення сервісів [11].

Користувач може відправляти запити Transact SQL до сервісу Windows Azure SQL Databases за допомогою протоколу TDS, що дозволяє додаткам використовувати Windows Azure SQL Databases так само, як вони використовують локальний SQL Server. Однак, оскільки Windows Azure SQL Databases є сервісом, його адміністрування має свої особливості. У відмінність від адміністрування локального SQL Server, Windows Azure SQL Databases розділяє логічний та фізичний аспекти адміністрування. Клієнт продовжує адмініструвати базу даних, керувати логінами, користувачами та ролями, але про обладнання дбає Microsoft. У результаті, Windows Azure SQL Databases надає масштабований багатокористувацький сервіс баз даних з найвищим рівнем доступності, масштабовості, безпеки та самовідновлення.

1.4.1. Огляд платформи

Платформа Windows Azure є хмарною платформою для додатків, яка дозволяє зберігати дані та виконувати додатки у дата-центрах Microsoft. Windows Azure надає хмарну операційну систему, на основі якої працюють всі сервіси Azure та розроблені додатки [12]. Ця платформа надає доступ до можливостей публічного хмарного середовища. Використовуючи публічний хмарний сервіс, клієнт сплачує лише за ресурси та потужність, які використовуються в додатку, і тільки за фактичний час використання цих ресурсів.

Основні особливості цієї моделі:

a) плата лише за спожиті ресурси;

b) загальна, багатопотокова структура обчислень;

с) абстракція від інфраструктури.

Роботоздатність платформи Windows Azure забезпечується 8 глобальними Data-центрами Microsoft які знаходяться на різних континентах.

ВИСНОВКИ ДО РОЗДІЛУ 1

У даному розділі було проведено аналіз Cloud-технологій та засобів моніторингу в хмарних середовищах. Виявлено, що Cloud-технології набули широкого розповсюдження в сучасному інформаційному середовищі, дозволяючи ефективно використовувати ресурси та послуги, доступні у хмарі. Це дає можливість підприємствам збільшити масштаб своїх операцій, прискорити розробку та розгортання додатків, а також знизити витрати на обладнання та обслуговування.

Особлива увага була приділена Microsoft Windows Azure - одному з провідних хмарних рішень на ринку. Виявлено, що Azure надає широкі можливості для створення, розгортання та управління хмарними додатками та послугами. Крім того, було розглянуто криптографічні сервіси та захист даних у Windows Azure, зокрема, сховище ключів та шифрування в SQL Azure. Ці засоби дозволяють забезпечити конфіденційність та цілісність даних, зменшуючи ризики їхнього несанкціонованого доступу та зловживання.

Детальний опис Azure SQL показав, що ця платформа має ряд переваг для роботи з базами даних у хмарному середовищі. Зокрема, надійність, масштабованість та висока продуктивність роботи з даними. Огляд платформи Azure SQL дозволяє розуміти, як ця технологія може бути використана для реалізації різноманітних додатків та послуг, що вимагають надійного та швидкого доступу до даних.

Отже, аналіз Cloud-технологій та засобів моніторингу в хмарних середовищах, а також розгляд Microsoft Windows Azure та його криптографічних сервісів та бази даних Azure SQL, надають знання та розуміння необхідних аспектів для подальшого дослідження та розробки мультимедійних систем у хмарному середовищі з врахуванням їхньої функціональності та ефективності.

РОЗДІЛ 2
ЗАБЕЗПЕЧЕННЯ ЗАХИСТУ ІНФОРМАЦІЇ В МЕРЕЖЕВИХ БАЗАХ ДАНИХ

2.1. Актуальність застосування захисту інформації в мережевих БД

Актуальність застосування безпеки інформації в мережевих базах даних полягає в необхідності забезпечення захисту конфіденційності, цілісності і доступності даних, що зберігаються і обробляються в мережевих середовищах. З огляду на зростаючу кількість загроз інформаційній безпеці, а також значущість даних, які зберігаються в мережевих базах даних, важливо використовувати ефективні заходи захисту, щоб запобігти несанкціонованому доступу, втраті даних або їх спотворенню. Забезпечення безпеки інформації в мережевих базах даних є критично важливим для забезпечення довіри, конфіденційності та успішної функціонування бізнес-систем та організацій [13].

Тому, можна виділити наступні визначення:

Інформація - це дані про особи, об'єкти, події, явища і процеси (незалежно від форми їх подання), які використовуються для отримання знань та практичних рішень. Найважливішими в практичному плані властивостями інформації є цінність, достовірність і своєчасність. Цінність інформації визначається можливістю досягнення мети, поставленої перед отримувачем, тобто цінність інформації визначається її корисністю для власника (надає певні переваги).

Істинною або *достовірною інформацією* - є інформація, яка з достатньою точністю відображає об'єкти і процеси навколишнього світу. Своєчасність відображає відповідність цінності і достовірності інформації певному часовому періоду.

Під *інформаційними ресурсами* - розуміється накопичена інформація про навколишню дійсність, зафіксована на матеріальних носіях і в будь-якій іншій формі, що забезпечує її передачу в часі і просторі між різними користувачами для вирішення наукових, виробничих, управлінських та інших завдань. Особливо слід зазначити, що ресурсом є вся накопичена інформація, включаючи недостовірну інформацію,

представлену сумнівними фактами, хибними положеннями, неефективними підходами, а також застарілу і навмисну "дезінформацію", що надійшла в інформаційні потоки. Це дійсно важливо, оскільки без виявлення недостовірної та застарілої інформації, яка накопичується в інформаційних ресурсах, створюються передумови для прийняття неефективних, а в деяких випадках і помилкових рішень, що завдають значної шкоди [14].

Рис. 2.1. Аспекти інформаційної безпеки

Для баз даних і систем, що базуються на зберіганні даних, важливі три основні аспекти інформаційної безпеки:

- конфіденційність (потрібний захист від несанкціонованого доступу);
- цілісність (потрібний захист від втрати згоди та достовірності даних);
- доступність (потрібний захист від несанкціонованого утримання інформації та ресурсів, підтримка працездатності та захист від руйнування).

Захист даних - включає систему заходів, які захищають дані від несанкціонованого використання, руйнування або спотворення. У англійській мові використовуються такі терміни: Secrecy, Integrity і Availability of Data. Загальна ідея

захисту бази даних полягає в дотриманні рекомендацій, сформульованих для класу безпеки С2 в "Критеріях оцінки надійних комп'ютерних систем".

Оскільки організація системи захисту даних в базах даних та експертних системах не має принципових відмінностей, для зручності подальшого викладу ми будемо посилатись лише на бази даних та системи їх управління.

Слід враховувати, що практично неможливо забезпечити абсолютний захист даних, тому зазвичай обмежуються відносним захистом інформації - забезпечують її гарантовану безпеку на той період часу, поки несанкціонований доступ до неї, втрата її цілісності або доступності не мають небажані наслідки.

Можна перерахувати певні типові ситуації, в яких зберігані дані можуть бути вкрадені, спотворені або втрачені.

1. *Несанкціонований доступ (НСД).* Якщо будь-яка особа може отримати доступ до будь-яких збережених даних та внести в них будь-які зміни, то дані можуть бути пошкоджені некомпетентним або зловмисним користувачем, або використані з порушенням прав власника або в його шкоду.

2. *Неконтрольований паралелізм.* Зазвичай у багатокористувацькій системі з базою даних одночасно працює декілька користувачів. Деякі з них можуть намагатися одночасно змінювати стан БД. Якщо вони робитимуть це незалежно, то результируючий стан БД може бути неконсистентним.

3. *Локальна помилка.* У процесі виконання прикладної програми може виникнути аварійна ситуація (наприклад, ділення на нуль), в результаті якої виконання програми буде припинено. Якщо прикладна програма виконувала оновлення даних, то БД може опинитися в неконсистентному стані.

4. *Втрата оперативної пам'яті (м'який збій).* Незважаючи на високу надійність персональних комп'ютерів (ПК), у будь-який момент може статися системний збій (наприклад, відключення живлення), в результаті якого втрачатиметься вміст системних буферів і буферів програм, розташованих в оперативній пам'яті. Подібні ситуації називаються м'якими збоями системи. Оскільки стан БД в момент м'якого збою непередбачуваний, він може опинитися в неконсистентному стані після

перезавантаження системи. В відміну від локального збою, при м'якому збої можуть постраждати дані всіх користувачів.

5. *Фізичне руйнування даних (жорсткий збій)*. Під час експлуатації зовнішнього пристрою пам'яті може бути пошкоджена поверхня диска, блок головок дисководу і т.д. Подібні ситуації називаються жорсткими збоями системи. Ймовірність жорсткого збою дуже мала, проте не можна її ігнорувати.

Вимоги до захисту даних від руйнування у загальному вигляді зводяться до наступного: жодні випадкові або навмисні руйнування даних не можуть бути необоротними.

Зазначені вище ситуації можуть бути наслідком:

1. Некомпетентних, безвідповідальних або злочинних дій користувача;

2. Несогласованих оновлень, що виконуються одночасно та незалежно декількома користувачами;

3. Виникнення неправильно обробленої помилки в прикладній програмі;

4. Введення шкідливого коду в програмне забезпечення;

5. Аварійного відключення енергії, збою або відмови різних пристроїв системи, порушення зв'язку;

6. Фізичного знищення комп'ютерної системи або її частин.

Отже, для захисту даних від втрати цілісності, руйнування та несанкціонованого використання система повинна забезпечувати:

1. Підтримку обмежень прав та повноважень доступу користувачів до даних, визначених політикою інформаційної безпеки підприємства.

2. Захист від шкідливих дій користувачів, спрямованих на пошкодження даних або отримання конфіденційної інформації на основі аналізу доступної інформації.

3. Дисципліну паралельної роботи багатьох користувачів, що виключає можливість внесення несогласованих змін до стану бази даних.

4. Відновлення цілісного стану бази даних після локальних та м'яких збоїв системи.

5. Відновлення бази даних після жорстких збоїв.

Для забезпечення безпеки даних застосовуються комп'ютерні та некомп'ютерні засоби захисту.

Рис. 2.2 Засоби захисту інформації

Спочатку розглянемо комп'ютерні засоби захисту. Традиційними є наступні засоби контролю та захисту:

1. *Авторизація користувачів та обмеження доступу.* Авторизація користувачів означає надання певних прав користувачеві, що дозволяють йому законно отримувати доступ до всієї системи або окремих об'єктів. Це може бути здійснено як засобами операційної системи, так і засобами самої СУБД.

2. *Доступ до даних через представлення.* Представлення - це результат однієї або кількох реляційних операцій з базовими відношеннями з метою створення іншого відношення. Представлення є віртуальним відношенням, якого фактично не існує в БД, але створюється на запит користувача. Механізм представлення є потужним і гнучким інструментом організації захисту даних, який дозволяє приховати від певних користувачів деякі частини БД. В результаті користувачі не будуть мати жодної інформації про існування атрибутів або рядків даних, які недоступні через представлення, які є у їхньому розпорядженні. Представлення може бути визначено на основі кількох відношень, після чого користувачу будуть надані необхідні привілеї доступу до цього представлення, але не до базових відношень. В даному випадку

використання представлення є більш жорстким механізмом контролю доступу, ніж звичайне надання користувачу певних прав доступу до базових відношень.

3. *Оновлення даних та реалізація правил "бізнес-логіки" за допомогою збережених процедур, функцій та тригерів.* Збережені процедури, функції та тригери є фрагментами коду на високорівневій мові програмування, які розширюють мову SQL, зберігаються та виконуються на сервері. Реалізація всіх правил "бізнес-логіки" за допомогою збережених процедур, функцій та тригерів надає значно вищий рівень захисту даних, ніж їх реалізація в клієнтських програмах.

4. *Резервне копіювання та відновлення.* Резервне копіювання - періодична процедура отримання копії БД та її журнального файлу (а також, можливо, програм СУБД) на зовнішній постійний носій, зазвичай на окремому носії. У разі відмови, що призводить до непридатності БД для подальшої експлуатації, резервна копія та фіксована в журнальному файлі оперативна інформація використовуються для відновлення БД до останнього згодованого стану. Ведення журналу - це процедура створення та обслуговування файлу журналу, який містить інформацію про всі зміни, внесені в БД з моменту створення останньої резервної копії, і призначений для ефективного відновлення системи у разі відмови. Якщо в системі, що відмовила, функція ведення системного журналу не використовувалася, 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 втрачені.

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

6. *Управління паралелізмом виконання транзакцій.* Управління транзакціями є одним з найважливіших завдань СУБД. Розвинуті багатокористувацькі СУБД мають досить сучасні засоби управління паралелізмом виконання транзакцій, що дозволяють багатьом користувачам мати одночасний доступ до даних.

7. *Аудит (реєстрація всіх звернень до захищуваної інформації)*. Для визначення активності використання БД аналізуються її файли журналу. Ці ж джерела можуть бути використані для виявлення будь-яких незвичайних дій у системі. Регулярне проведення аудиторських перевірок, доповнене постійним контролем вмісту файлів журналу з метою виявлення аномальної активності в системі, дуже часто дозволяє вчасно виявити та припинити будь-які спроби порушення захисту.

8. *Шифрування.* Надійний (на певний час) захист конфіденційних даних можливий лише за допомогою шифрування даних. Шифрування - це кодування даних за допомогою спеціального алгоритму, в результаті чого дані стають недоступними для читання. Для успішного розшифрування даних потрібне знання ключа розшифрування. Деякі СУБД мають вбудовані засоби шифрування для захисту конфіденційної інформації, але також можна використовувати додаткові засоби шифрування на рівні операційної системи.

Це лише деякі загальні засоби захисту даних у базах даних. Реалізація конкретних заходів залежить від типу і розмірів бази даних, а також від специфіки використання та вимог до безпеки.

Некомп'ютерні засоби захисту включають прийняття адміністративних заходів, розробку обмежень та угод. До них належать:

- Заходи забезпечення безпеки та планування захисту від непередбачуваних обставин.
- Контроль над персоналом та загальний контроль за фізичним доступом.
- Захист приміщень та сховищ за допомогою різноманітних охоронних систем.
- Укладання гарантійних угод та договорів щодо супроводження апаратного забезпечення та програмного забезпечення систем безпеки, які виробляються третьою стороною, що дозволяє на певний термін продовжити впевненість у тому, що ці засоби не стануть причиною реалізації загроз безпеки.

У документах, що стосуються заходів забезпечення безпеки, слід визначити наступне:

- Сферу ділових процесів організації, для яких вони встановлюються.
- Відповідальність та обов'язки окремих працівників.
- Дисциплінарні заходи, що приймаються у разі виявлення порушення встановлених обмежень.
- Перелік обов'язково виконуваних процедур.

2.2. Основні принципи забезпечення інформаційної безпеки

Розглянуті підходи до забезпечення безпеки можуть бути реалізовані за дотримання таких основних принципів:

1. *Принцип системності.* Системний підхід до захисту комп'ютерних систем передбачає необхідність врахування всіх взаємозв'язаних, взаємодіючих та змінюваних з часом елементів, умов і факторів на всіх етапах життєвого циклу системи з урахуванням взаємодії об'єкта захисту з зовнішнім середовищем. Система захисту повинна будуватися з урахуванням всіх відомих шляхів проникнення, а також з можливістю появи нових шляхів реалізації загроз безпеки.

2. *Принцип комплексності.* Спеціалісти з комп'ютерної безпеки мають в своєму розпорядженні широкий спектр заходів, методів і засобів захисту комп'ютерних систем. Зокрема, сучасні обчислювальні пристрої, операційні системи та прикладне програмне забезпечення мають вбудовані елементи захисту. Комплексне їх використання передбачає взаємодію різнорідних засобів для побудови єдиної системи захисту, яка покриває всі суттєві шляхи реалізації загроз і не містить слабких місць у з'єднаннях окремих її компонентів.

3. *Принцип неперервності захисту.* Захист інформації - це неодноразова подія або конкретний набір заходів щодо розгортання та налаштування засобів захисту, а неперервний цілеспрямований процес, який передбачає вжиття відповідних заходів на всіх етапах життєвого циклу КС (починаючи з ранніх стадій проектування, а не лише на етапі експлуатації). Розробка системи захисту повинна вестися паралельно з розробкою самої захищуваної системи. Це дозволить врахувати вимоги безпеки при проектуванні архітектури і, у кінцевому рахунку, створить більш ефективні захищені

системи (як з точки зору витрат ресурсів, так і стійкості). Більшість фізичних та технічних засобів захисту потребують постійної організаційної (адміністративної) підтримки (своєчасна зміна та забезпечення правильного зберігання і використання ідентифікаторів, паролів, шифрувальних ключів, переоцінка повноважень тощо). Перерви в роботі засобів захисту можуть бути використані зловмисниками для аналізу використовуваних методів і засобів захисту, впровадження спеціального програмного та апаратного забезпечення для обходу системи захисту після відновлення її функціонування.

4. *Принцип розумної достатності.* Створити абсолютно неперебороную систему захисту принципово неможливо: за відповідних часу і ресурсів можна обійти будь-який захист. Наприклад, засоби криптографічного захисту в більшості випадків не гарантують абсолютної стійкості, а забезпечують конфіденційність інформації при використанні сучасних обчислювальних засобів протягом прийнятного для захищеної сторони часу. Тому розумно говорити лише про прийнятний рівень безпеки. Високоефективна система захисту є дорогою, використовує значну частину потужності і ресурсів комп'ютерної системи і може створювати помітні додаткові незручності користувачам. Важливо правильно вибрати достатній рівень захисту, при якому витрати, ризик і розмір можливих збитків були б прийнятними (задача аналізу ризику).

5. *Гнучкість системи захисту.* Часто доводиться створювати систему захисту в умовах великої невизначеності. Тому прийняті заходи та встановлені засоби захисту, особливо в початковий період їх експлуатації, можуть забезпечувати як надмірний, так і недостатній рівень захисту. Зрозуміло, що для забезпечення можливості змінювати рівень захищеності, засоби захисту повинні мати певну гнучкість. Особливо це важливо у випадках, коли засоби захисту необхідно встановлювати на працюючу систему, не порушуючи процес її нормального функціонування. Крім того, зовнішні умови і вимоги з часом змінюються. У таких ситуаціях властивість гнучкості врятує власників КС від необхідності прийняття кардинальних заходів щодо повної заміни засобів захисту.

6. *Принцип відкритості алгоритмів та механізмів захисту.* Полягає в тому, що захист не повинен забезпечуватись лише за рахунок засекреченості структурної організації та алгоритмів функціонування його підсистем. Знання алгоритмів роботи системи захисту не повинно давати можливості її подолання (навіть автору). Однак це зовсім не означає, що інформація про конкретну систему захисту повинна бути загальнодоступною - необхідно забезпечувати захист від загрози розкриття параметрів системи.

7. *Принцип простоти застосування засобів захисту.* Полягає в тому, що механізми захисту повинні бути інтуїтивно зрозумілими й легкими у використанні. Використання засобів захисту не повинно бути пов'язане зі знанням спеціальних мов або виконанням дій, які вимагають значних додаткових зусиль при звичайній роботі законних користувачів, а також не повинно вимагати від користувача виконання рутинних незрозумілих йому операцій (введення декількох паролів і імен тощо). Це неминуче призведе до висвітлення списку паролів на найбільш видимому місці, наприклад, на моніторі.

2.3. Управління доступом до даних

В будь-якій організації існують певні правила зберігання та використання інформації, які обмежують доступ до інформаційних ресурсів. Конкретні правила визначаються інформаційною політикою керівництва підприємства, але, в будь-якому випадку, розумні обмеження доступу базуються на наступних принципах:

- Підприємство є власником всієї службової інформації, одержаної його підрозділами або співробітниками.
- Підрозділ або співробітник є власником одержаної ним інформації і може використовувати її в інтересах підприємства без обмежень.
- Співробітник має право доступу лише до тієї інформації, яка є необхідною для виконання його службових обов'язків.
- Співробітник має право виконувати лише ті операції з доступною інформацією, які обумовлені його службовими обов'язками.

Ці принципи реалізуються у вигляді системи правил, які обмежують права доступу співробітників до інформаційних ресурсів підприємства. Сучасні системи управління базами даних (СУБД) мають розвинуті засоби підтримки подібних правил - підсистеми адміністрування даних [15]. Метою підсистеми адміністрування є забезпечення санкціонованого доступу співробітників підприємства до збережених даних. Концептуально, вона повинна надавати привілеї користувачам СУБД щодо даних та контролювати надання конкретному користувачу лише певних привілеїв.

За забезпеченням доступу користувачів до комп'ютерної системи зазвичай відповідає системний адміністратор, чиї обов'язки включають створення облікових записів користувачів. Кожному користувачеві присвоюється унікальний ідентифікатор, який операційна система використовує для визначення особи. Кожен ідентифікатор пов'язується з паролем, який обирає користувач і відомий операційній системі. При реєстрації користувач повинен надати свій пароль системі для аутентифікації. Така процедура дозволяє забезпечити контрольований доступ до комп'ютерної системи, але не обов'язково надає право доступу до СУБД або іншої прикладної програми. Для надання користувачу прав доступу до СУБД може використовуватись окрема подібна процедура. Відповідальність за надання прав доступу до СУБД, зазвичай, несе адміністратор бази даних, чиї обов'язки включають створення окремих ідентифікаторів користувачів, вже в межах самої СУБД. Кожен з ідентифікаторів користувачів СУБД також пов'язується з паролем, який повинен бути відомий тільки цьому користувачеві.

Деякі СУБД підтримують списки дозволених ідентифікаторів користувачів та паролів, які відрізняються від аналогічного списку, підтримуваного операційною системою. Інші типи СУБД підтримують списки, елементи яких відповідають існуючим спискам користувачів операційної системи і здійснюють реєстрацію на основі поточного ідентифікатора користувача, вказаного ним при реєстрації в системі. Це запобігає спробам користувачів зареєструватися в СУБД під ідентифікатором, відмінним від того, яким вони користувалися при реєстрації в системі.

Виходячи з вищезазначеного, сформулюємо термін "управління доступом" більш детально. Управління доступом - це метод захисту інформації шляхом регулювання використання ресурсів системи (елементів БД, програмних та технічних засобів). Воно включає такі функції захисту:

- ідентифікація користувачів та ресурсів системи;
- підтвердження справжності об'єкта або суб'єкта за наданим ним ідентифікатором (аутентифікація);
- розмежування та перевірка повноважень (авторизація). Створення умов роботи в межах встановленого регламенту;
- реєстрація звернень до захищених ресурсів (протоколювання та аудит);
- реагування на спроби несанкціонованого доступу.

2.4. Ідентифікація та автентифікація користувачів

Для отримання доступу до бази даних користувач повинен вказати свій ідентифікатор (ID) та підтвердити право його використання. Після ідентифікації користувача, система готова виконувати операції з даними від його імені. Залежно від рівня безпеки системи можуть використовуватися різні процедури для підтвердження особистості користувача.

Рис. 2.3 Автентифікація користувачів

Нижче наведено прості й найчастіше використовувані програмні процедури аутентифікації, які не пов'язані з аналізом "по-черка" користувача (наприклад,

швидкістю натискання клавіш клавіатури та інтервалами між окремими натисканнями):

1. *Парольний захист.* Простий варіант аутентифікації полягає у тому, що з ідентифікатором пов'язується пароль - набір символів, відомий тільки власнику ідентифікатора та системі. Власник ідентифікатора має мінімальні повноваження, такі як доступ до системи та можливість змінювати свій пароль. Жоден користувач (включаючи системного адміністратора) не має права переглядати паролі. Паролі зберігаються у вигляді хеш-коду або зашифровані.

2. *Захист з використанням запитання-відповіді.* Складніший спосіб входу в систему може передбачати введення власником ідентифікатора під час реєстрації серії запитань та відповідей на них. Запитання мають особистий характер, тому неможливо вгадати правильні відповіді на них. Наприклад: "Як на кличку відгукується ваш племінник?" або "Де народилася ваша бабуся?" і т.д. При спробі входу в систему система випадковим чином висуває користувачу одне або кілька запитань з цієї серії для перевірки його особистості.

3. *Обумовлений алгоритм.* Якщо потенційний зловмисник має доступ до комунікаційної лінії, що з'єднує клієнтський комп'ютер і сервер, для захисту входу можна використовувати обумовлений алгоритм, відомий власнику ідентифікатора та системі. При спробі входу система висуває випадкове число користувачеві. Користувач повинен виконати необхідні перетворення та ввести результат. Спостерігач на лінії може побачити лише початкове і кінцеве числа. Такий вид захисту широко використовується в системах доступу до банківських рахунків в Інтернеті. Власнику рахунку видається спеціальний портативний пристрій з чіпом банківської смарт-карти. Під час встановлення з'єднання власник спершу вставляє карту в зчитувач, вводить ПІН-код, а потім випадковим чином згенероване на сторінці доступу число. Відповідь цього пристрою надсилається назад на сервер.

Під час мережевого з'єднання обидва об'єкти, що з'єднуються, повинні пройти процедуру аутентифікації. Після встановлення з'єднання необхідно виконати вимоги щодо захисту при обміні повідомленнями:

- отримувач повинен бути впевнений у автентичності джерела даних;

- отримувач повинен бути впевнений у автентичності переданих даних;
- відправник повинен бути впевнений у доставці даних отримувачу;
- відправник повинен бути впевнений у автентичності доставлених даних.

Цю вимогу до захисту можна виконати за допомогою так званого цифрового підпису. Якщо всі ці чотири вимоги реалізовані в Системі Керування Базою Даних (СКБД), то забезпечується функція підтвердження передачі, коли відправник не може заперечувати надсилання повідомлення або його змісту, а отримувач не може заперечувати факт отримання повідомлення або його автентичність.

Інформація про зареєстрованих користувачів бази даних зберігається в її системному каталозі. Сучасні Системи Керування Базами Даних (СКБД) не мають спільного синтаксису SQL-запиту для підключення до бази даних, оскільки їх власний синтаксис сформувався раніше, ніж стандарт ISO. Найчастіше використовується запит CONNECT. Наприклад, для IBM DB2:

CONNECT TO <БД> USER <користувач> USING <пароль>

З'єднання з системою непідтверджених користувачів та користувачів, у яких перевірка автентичності представленого ідентифікатора не пройшла, виключається. Протягом сеансу роботи користувача (від успішного проходження ідентифікації та аутентифікації до відключення від системи) всі його дії безпосередньо пов'язані з результатом ідентифікації. Відключення користувача може бути як нормальним, так і примусовим (з боку користувача-адміністратора, наприклад, у випадку видалення користувача або аварійного відключення зв'язку між клієнтом та сервером). У другому випадку користувач повинен бути проінформований про це, і всі його дії скасовуються до остаточного фіксування змін, внесених ним в таблиці бази даних. У будь-якому випадку протягом сеансу роботи ідентифікований користувач буде суб'єктом доступу до засобів захисту інформації від несанкціонованого доступу до СКБД.

2.5. 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.

Рис. 2.4 Авторизація користувачів

Усі суб'єкти доступу можуть бути розподілені системою на декілька категорій, наприклад: CONNECT, RESOURCE і DBA. Набір таких категорій визначається виробником Системи Керування Базою Даних (СКБД). Збільшення можливостей (повноважень) для кожного окремого типу підключення відбувається в зазначеному порядку:

- *CONNECT* - кінцеві користувачі. За замовчуванням їм дозволено лише підключатися до бази даних та виконувати запити до даних, всі їхні дії регламентуються виданими їм привілегіями;
- *RESOURCE* - привілеговані користувачі, які мають право створювати власні об'єкти в базі даних (таблиці, представлення, збережені процедури та тригери).
- *DBA* - категорія адміністраторів бази даних. Включає можливості обох попередніх категорій, а також можливість реєстрації суб'єктів захисту або зміни їх категорії.

Варто особливо відмітити, що в деяких Системах Керування Базою Даних (СКБД) адміністративні дії також розподілені, що призводить до наявності додаткових категорій. Наприклад, в Oracle користувач з ім'ям DBA є адміністратором сервера баз даних, а не лише однієї окремої бази даних. У IBM DB2 існує кілька категорій адміністраторів: SYSADM (найвищий рівень; системний адміністратор, який має всі привілеї); DBADM (адміністратор бази даних, який має повний набір привілеїв у межах конкретної бази даних) [16]. Привілеї управління сервером баз даних належать користувачам з іменами SYSCTRL (найвищий рівень повноважень управління системою, які застосовуються лише до операцій, що впливають на системні ресурси; безпосередній доступ до даних заборонений, дозволені операції створення, зміни та видалення бази даних, створення та видалення просторів таблиць). Для кожної адміністративної операції в IBM DB2 визначено необхідний набір адміністративних категорій, до яких повинен належати користувач, що виконує певний запит адміністрування. Так, назначення привілеїв користувачам може здійснювати SYSADM або DBADM, а для створення об'єкта даних користувач повинен мати привілегію CREATETAB.

Адміністратор кожної Бази Даних займається створенням списку можливих користувачів Бази Даних та розподілом повноважень цих користувачів. Інформація про розгортання обмежень знаходиться в системному каталозі бази даних. Очевидно, що ця інформація може бути використана для несанкціонованого доступу і, отже, також потребує захисту. Захист цих даних здійснюється засобами самої системи керування базою даних (СКБД).

На сьогоднішній день найбільш поширені три основні підходи до розгортання доступу: вибірковий, обов'язковий і рольовий. Вибірковий підхід описується моделлю дискреційного контролю доступу (DAC), обов'язковий або повноважний - моделлю мандатного контролю доступу (MAC). Обидва забезпечують створення системи безпеки як для бази даних в цілому, так і для окремих її об'єктів - таблиць, представлень, кортежів і т.д., включаючи конкретні значення певного атрибута в певному кортежі визначеного відношення. Рольова модель розгортання доступу (RBAC) є, в певному сенсі, комбінацією згаданих вище моделей. Слід згадати також

про модель Китайської стіни, яка полягає в побудові фізичного бар'єра між базою даних і групою авторизованих осіб та рештою світу.

ВИСНОВКИ ДО РОЗДІЛУ 2

У цьому розділі була розглянута актуальність застосування безпеки інформації в мережевих базах даних. Виявлено, що з огляду на зростаючу кількість кібератак та загроз безпеці даних, захист інформації є критично важливою задачею для підприємств та організацій. Враховуючи те, що бази даних зберігають значну кількість чутливої інформації, необхідно вживати заходів для їхнього захисту від несанкціонованого доступу та зловживання.

У розділі також було розглянуто основні принципи забезпечення інформаційної безпеки. Встановлено, що безпека інформації базується на таких важливих аспектах, як конфіденційність, цілісність та доступність даних. Для досягнення цих принципів необхідно впроваджувати різноманітні технології та методи захисту, такі як шифрування, контроль доступу та аудит безпеки.

Управління доступом до даних є ще одним важливим аспектом захисту інформації. Встановлено, що ефективне управління доступом до бази даних полягає в правильній ідентифікації та автентифікації користувачів. Ідентифікація дозволяє встановити особу, яка намагається отримати доступ до системи, тоді як автентифікація перевіряє правомірність цього доступу. Застосування надійних методів ідентифікації та автентифікації є ключовим для запобігання несанкціонованому доступу до системи та даних.

Також, було розглянуто питання авторизації користувачів, вона дозволяє встановити права доступу користувача до певних ресурсів системи чи бази даних. Ефективна авторизація забезпечує гнучкість та гранулярність контролю доступу, дозволяючи адміністраторам системи встановлювати рівні привілеїв для кожного користувача відповідно до їхніх ролей та обов'язків.

Загалом, у цьому розділі було проведено детальний аналіз основних аспектів забезпечення інформаційної безпеки в системах та мережах. Розглянуті принципи

конфіденційності, цілісності та доступності даних, управління доступом, ідентифікація, автентифікація та авторизація користувачів. Застосування цих принципів і методів забезпечення безпеки є критичним для забезпечення надійного та захищеного середовища для мультимедійних систем у мережевому контексті.

……

РОЗДІЛ 3
АРХІТЕКТУРА СИСТЕМИ БЕЗПЕКИ SQL SERVER

З огляду на постійно зростаючі загрози кібербезпеці і необхідність забезпечення безпеки даних, роль систем безпеки в структурі баз даних стає надзвичайно важливою.

Microsoft SQL Server, одна з провідних систем управління базами даних, надає розмаїті функції та інструменти для забезпечення безпеки. Ось кілька ключових аспектів, які варто враховувати при впровадженні системи безпеки SQL Server:

Автентифікація та авторизація: SQL Server дозволяє встановлювати рівні доступу для користувачів та ролей, обмежуючи права доступу до бази даних. Використання надійних методів автентифікації, таких як логіни та паролі або інтеграція зі службою доменного контролера, дозволяє підтверджувати ідентичність користувачів перед наданням доступу до даних.

Шифрування даних: SQL Server надає можливості шифрування для захисту конфіденційної інформації. Transparent Data Encryption (TDE) дозволяє шифрувати дані на рівні файлів бази даних, забезпечуючи захист від несанкціонованого доступу до фізичних файлів [17]. Крім того, SQL Server дозволяє шифрувати резервні копії баз даних та з'єднання по SSL для забезпечення безпеки під час передачі даних.

Аудит безпеки: SQL Server дозволяє встановлювати аудиторські політики для відстеження подій, які відбуваються в базі даних. Це дозволяє контролювати доступ до даних, виявляти можливі порушення безпеки та забезпечувати відповідність з правилами безпеки. Аудит безпеки є важливим інструментом для виявлення та реагування на потенційні загрози.

Моніторинг та захист від вторгнень: SQL Server надає можливості моніторингу активності, які допомагають виявляти небажану або зловмисну діяльність. Інструменти виявлення та запобігання вторгненням допомагають уникнути атак та забезпечити безпеку бази даних.

Загально кажучи, система безпеки SQL Server постійно оновлюється та вдосконалюється, щоб протидіяти новим загрозам і забезпечити надійний рівень захисту даних. Впровадження цих функцій та інструментів допоможе зменшити ризики безпеки, збільшити конфіденційність даних та забезпечити відповідність регуляторним вимогам.

3.1. Система безпеки рівня сервера

Система безпеки MS SQL Server впроваджує дискреційний та рольовий підходи до керування доступом та має два рівні: рівень сервера та рівень бази даних (див. Рис. 3.1). 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 керувати доступом користувачів до баз даних.

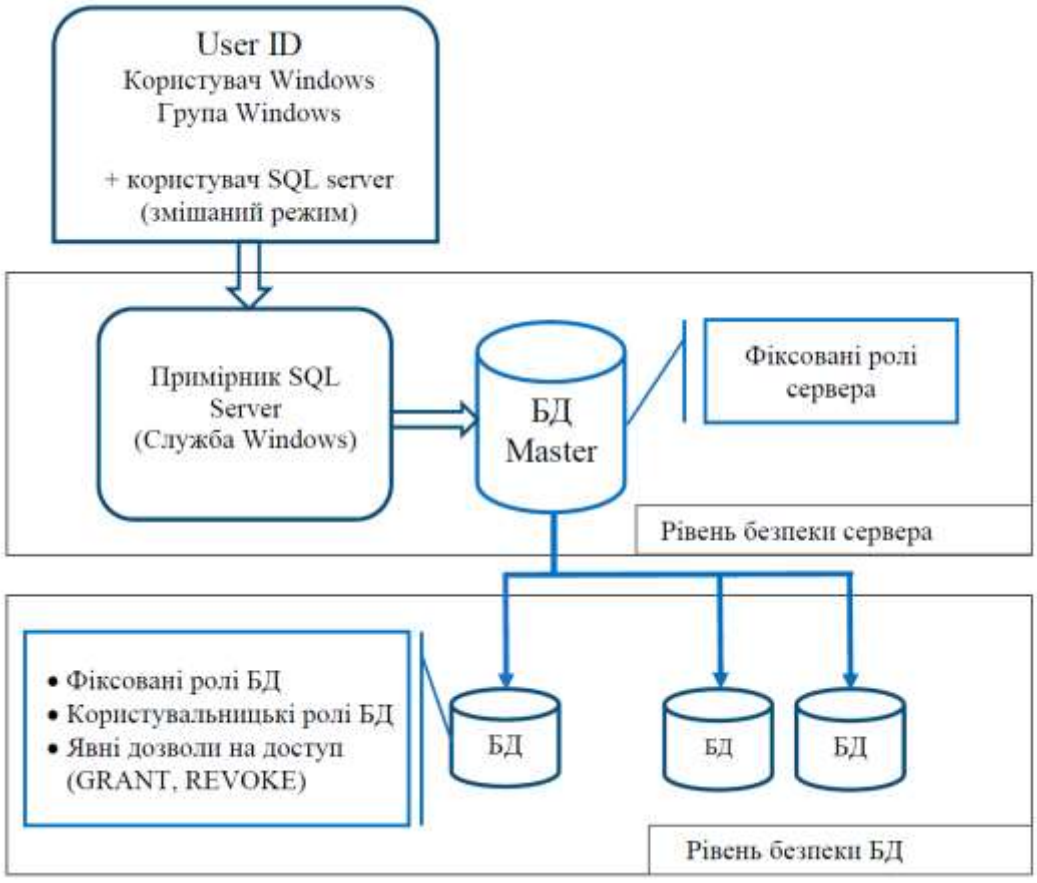


Рис. 3.1. Узагальнена схема безпеки СУБД MS SQL Server

На рівні сервера система безпеки оперує такими поняттями:

- аутентифікація (authentication);
- обліковий запис (login);
- вбудовані ролі сервера (fixed server roles).

На рівні бази даних використовуються поняття:

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

Отже, можна виділити дві групи ролей:

- ролі сервера (server role);
- ролі бази даних (database role).

SQL Server використовує двоетапну схему аутентифікації, де користувач спочатку проходить аутентифікацію на сервері. Тільки після успішної аутентифікації користувачеві надається доступ до однієї або декількох баз даних. Вся інформація про реєстраційні записи зберігається в базі даних master.

SQL Server пропонує два режими аутентифікації користувачів:

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

Змішаний режим аутентифікації (Windows Authentication and SQL Server Authentication): у цьому режимі системи аутентифікації Windows і SQL Server функціонують незалежно одна від одної.

Під час встановлення SQL Server одним із перших рішень, які потрібно прийняти, є вибір методу аутентифікації. Режим аутентифікації, встановлений під час інсталяції, можна змінити на сторінці Security у властивостях SQL Server за допомогою утиліти Management Studio. У програмному коді поточний режим аутентифікації можна перевірити за допомогою системної збереженої процедури xp_loginconfig:

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

Результат виконання цієї процедури має такий вигляд:

```
name config_value
login mode Mixed
```

Відповідно, сервер ідентифікує користувача за одним з таких методів:

- За допомогою облікового запису користувача Windows, який, як правило, належить до домену або робочої групи.
- На основі членства в одній з груп користувачів Windows.
- За допомогою окремого реєстраційного запису SQL Server, якщо використовується змішана модель безпеки на сервері.

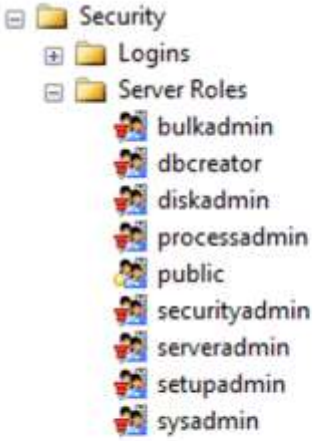
Важливо не плутати реєстраційні записи SQL Server з обліковими записами Windows на цьому сервері баз даних, оскільки ці два типи реєстрації є зовсім різними. Користувачам SQL Server не потрібен прямий доступ до каталогів, де зберігаються файли бази даних або інші файли, оскільки доступ до них здійснюється процесом SQL Server, а не користувачем. Проте, процесу SQL Server потрібні права доступу до цих файлів, і для цього використовується обліковий запис Windows. Існують два варіанти:

Обліковий запис локального адміністратора: SQL Server може використовувати цей обліковий запис для отримання доступу до комп'ютера, на якому він встановлений. Цей варіант подібний до встановлення на віддаленому сервері, оскільки в такому випадку немає можливості підтримувати мережеву систему безпеки Windows, яка необхідна для розподіленої обробки.

Обліковий запис користувача домену (рекомендований): SQL Server може використовувати обліковий запис користувача домену Windows, створений спеціально для нього. Цьому обліковому запису користувача SQL Server можуть бути призначені адміністративні привілеї для локального сервера, і він може отримати доступ до мережі для взаємодії з іншими серверами [18].

Набір ролей сервера є строго обмеженим. Ніхто, включаючи адміністратора сервера, не може створити нову роль сервера або видалити наявну. Ці ролі

називаються фіксованими ролями сервера (fixed server roles). Нижче наведено список деяких фіксованих ролей сервера з коротким описом кожної з них (рис. 3.2).



- **Sysadmin (System Adminis-trators).** Члени цієї ролі мають абсолютні права в SQL Server. Ніхто не має більших прав доступу, ніж члени цієї ролі.
- **Setupadmin (Setup Adminis-trators).** Цій ролі надано права управління пов'язаними серверами, конфігурації збережених процедур, що запускаються автоматично під час старту SQL Server, а також право додавати облікові записи в роль **setupadmin**.
- **Serveradmin (Server Adminis-trators).** Зазвичай у цю роль включаються користувачі, які повинні виконувати адміністрування сервера. Мають право на зупинку сервера (SHUTDOWN), змінювати параметри роботи служб (sp_configure), застосовувати зміни (RECONFIGURE), керувати повнотекстовим пошуком (sp_fulltext_service).
- **Securityadmin (Security Admin-istrators).** Члени цієї ролі мають можливість створювати нові облікові записи, яким вони можуть надавати права на створення баз даних та їхніх об'єктів, а також керувати пов'язаними серверами, включати облікові записи в роль securityadmin і читати журнал помилок SQL Server.
- **Dbcreator (Database Crea-tors).** Члени цієї ролі можуть створювати нові бази даних, видаляти і перейменовувати наявні, відновлювати резервні копії бази даних і журналу транзакцій.

Рис. 3.2. Фіксовані ролі сервера (fixed server roles)

Щоб надати обліковому запису користувача певну фіксовану роль сервера, можна використати збережену процедуру sp_addsrvrolemember. Ця процедура має такий синтаксис:

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

Давайте розглянемо приклад, як додати обліковий запис Windows з ім'ям Admin на комп'ютері STORAGE до фіксованої ролі сервера sysadmin:

```
EXEC sp_addsrvrolemember 'STORAGE\Admin', 'sysadmin'
```

Хоча, як правило, потрібно мати обліковий запис на сервері, який буде включений в одну з фіксованих ролей, є одна виняткова ситуація. Користувач Windows може отримати доступ до SQL Server як член певної групи Windows, якій вже надано доступ до сервера. В таких випадках такі облікові записи можуть бути автоматично включені в ролі сервера, без необхідності окремо надавати доступ обліковому запису. Для отримання інформації про те, який обліковий запис знаходиться в якій фіксованій ролі сервера, використовується така системна збережена процедура:

```
sp_helpsrvrolemember [ [@srvrplename =] 'role']
```

Якщо процедура викликається без параметрів, вона повертає повний список облікових записів, які включені в будь-яку роль сервера. Але якщо потрібно отримати список облікових записів, які включені в конкретну роль сервера, треба вказати назву цієї ролі. В прикладі нижче показана інформація про членів ролі sysadmin:

```
EXEC sp_helpsrvrolemember 'sysadmin'
```

Цей запит наддасть вам інформацію про облікові записи, які є членами ролі sysadmin.

Для видалення облікового запису з фіксованої ролі сервера використовується системна збережена процедура sp_dropsrvrolemember, яка має такий синтаксис:

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

З переліку фіксованих ролей рівня сервера видно, що додавання користувачів до цих ролей може здійснювати лише адміністратори сервера, а не звичайними користувачами баз даних. Процес додавання звичайних користувачів баз даних залежить від обраного режиму аутентифікації користувачів.

Якщо використовується режим аутентифікації засобами OS Windows, то звичайні користувачі баз даних можуть бути додані шляхом надання їм відповідних привілеїв в самій базі даних, використовуючи інструменти керування базою даних.

У випадку змішаного режиму аутентифікації (Windows Authentication and SQL Server Authentication), додавання звичайних користувачів баз даних може бути здійснено за допомогою облікових записів SQL Server. Після створення облікового запису користувача в SQL Server він може бути наданий необхідні привілеї в базі даних.

Загальним висновком є те, що процес додавання звичайних користувачів баз даних залежить від обраного режиму аутентифікації та використовує відповідні механізми для надання їм доступу до баз даних.

3.2. Аутентифікація засобами ОС Windows

Аутентифікація за допомогою Windows дозволяє користувачам отримувати доступ до SQL Server, використовуючи їх існуючі облікові записи Windows. Під час процесу аутентифікації, ідентифікатор безпеки Windows передається з операційної системи на сервер баз даних. SQL Server припускає, що реєстрація користувачів у мережі здійснюється безпечним способом і не вимагає додаткових перевірок. Відразу після реєстрації в домені, користувач автоматично отримує відповідні права доступу до даних SQL Server. Цей метод надання доступу відомий як встановлення довірчого з'єднання.

Операційна система працює з обліковими записами (логінами), які зберігають усю необхідну інформацію про користувача, включаючи ім'я, пароль, членство у групах та каталог за замовчуванням. Кожен обліковий запис має унікальний ідентифікатор (ID логіна), також відомий як ідентифікатор безпеки (SID), який генерується випадковим чином при створенні облікового запису. Цей підхід дозволяє уникнути підробки облікових записів користувачів.

Якщо користувача видалять з домену, навіть якщо створити новий обліковий запис з тими самими характеристиками (ім'я облікового запису, пароль, членство в

групі), він не матиме доступу до об'єктів, до яких мав доступ оригінальний користувач. Це забезпечує високий рівень безпеки. У випадку SQL Server, якщо користувач домену мав певні права доступу, але був видалений, ніхто не може надати новому користувачу ті самі права доступу.

Аутентифікація Windows зберігає лише ідентифікаційний номер облікового запису користувача в системній базі даних Master. Інформація про ім'я користувача, пароль та інші дані зберігаються в базі даних домену. Зміна імені або пароля користувача не має впливу на права доступу до SQL Server. Під час підключення користувача, SQL Server зчитує інформацію про його обліковий запис та членство в групах Windows з бази даних системи безпеки домену. Якщо адміністратор вносить зміни до облікового запису, наприклад, виключає користувача з певної групи, ці зміни враховуються під час наступної реєстрації користувача в домені або SQL Server, залежно від характеру змін.

Аутентифікація Windows має кілька переваг. Всі правила політики безпеки, встановлені в домені, автоматично застосовуються до користувачів. Користувачам не потрібно запам'ятовувати ще один пароль, що підвищує загальний рівень безпеки даних. Наприклад, мінімальна довжина пароля та строк його дії автоматично контролюються операційною системою. Користувачам може бути заборонено використовувати паролі, які вже використовувалися раніше. Крім того, Windows має вбудовані заходи захисту від перебору паролів. Аутентифікація Windows також працює з групами користувачів. Коли ім'я групи Windows передається в SQL Server як реєстраційний запис, будь-який член цієї групи може бути аутентифікований сервером баз даних. SQL Server також знає справжнє ім'я користувача Windows, який входить до групи, що дозволяє здійснювати аудит на рівні користувачів та груп користувачів.

Для створення облікового запису користувача або групи Windows в SQL Server можна використовувати системну збережену процедуру sp_grantlogin. При виклику цієї процедури в якості аргументу передається повне ім'я користувача, включаючи ім'я домену, наприклад: "DOMAIN\Username".

```
EXEC sp_grantlogin 'RFE\Kharchenko'
```

Для видалення облікового запису або групи Windows з SQL Server програмним шляхом можна використовувати системну збережену процедуру sp_revokelogin. Ця процедура використовується для відкликання (видалення) раніше наданого доступу користувачеві до SQL Server. При виклику процедури sp_revokelogin передається аргументом ім'я облікового запису або групи Windows, яке необхідно видалити з SQL Server. Наприклад:

```
EXEC sp_revokelogin 'RFE\Kharchenko'
```

Цей код видалить обліковий запис користувача з іменем 'Username' у домені 'DOMAIN' з SQL Server.

Зрозуміло, ця операція не призводить до видалення цього облікового запису з операційної системи. Замість цього вона просто виключає користувача зі списку користувачів сервера баз даних.

Для заборони облікового запису Windows можна скористатися системною збереженою процедурою sp_denylogin. Таким чином, доступ будь-якого користувача до SQL Server може бути примусово закритий. Це повністю обмежує доступ користувача до SQL Server, навіть якщо він намагається отримати доступ за допомогою іншого методу. Наприклад:

```
EXEC sp_denylogin 'RFE\Kharchenko'
```

Якщо користувач Windows був доданий до SQL Server, а потім видалений з домену Windows, він продовжує існувати як користувач у базі даних, але вважається «осиротілим». Це означає, що, незважаючи на те, що формально у користувача є доступ до сервера, він не має доступу до ресурсів мережі і, отже, до всього набору засобів SQL Server. За допомогою системної збереженої процедури sp_validatelogins можна знайти «осиротілих» користувачів і отримати їх ідентифікатори системи

безпеки Windows NT та імена облікових записів. Наступний приклад ілюструє ситуацію, коли користувачу RFE\Joe було надано доступ до SQL Server, а потім його обліковий запис було видалено з Windows.

```
EXEC sp_validatelogins
SID NT Login
Ox0105000000000005150000000FCE31531A931 RFE\ Joe
```

Цей факт не можна розцінювати як уразливість системи безпеки. Без облікового запису Windows з відповідним ідентифікатором (SID), користувач не може підключитися до SQL Server (повторне створення облікового запису з заданим SID неможливе). Щоб вирішити проблему «осиротілих» користувачів, можна використовувати такий протокол:

- Відкличте права доступу користувача до всіх баз даних за допомогою збереженої процедури sp_revokedbaccess.
- Відкличте право доступу цього користувача до сервера за допомогою sp_revokelogin.
- Створіть новий обліковий запис для користувача.
- Зареєструйте обліковий запис на сервері.

Ці кроки допоможуть відновити належний доступ користувача із новим обліковим записом до SQL Server, після того як він був видалений з домену Windows.

3.3. Аутентифікація засобами SQL Server

Цей тип аутентифікації реалізується безпосередньо на сервері SQL Server. У цьому випадку повна інформація про користувачів, включаючи їх імена та паролі, зберігається в системній базі даних Master. Кожному користувачеві присвоюється ім'я облікового запису, унікальний ідентифікатор SQL Server, пароль та інші дані. Коли користувач намагається підключитися до сервера, система безпеки запитує ім'я облікового запису та пароль. Потім система порівнює ці дані з інформацією, що зберігається в системних таблицях. Якщо дані збігаються, користувач одержує

доступ. У разі невідповідності введених даних користувач отримує повідомлення про помилку, і підключення не встановлюється.

Використання додаткових реєстраційних записів SQL Server є доцільним, коли аутентифікація Windows недоступна або не відповідає вимогам системи. Ці реєстраційні записи забезпечують зворотну сумісність з попередніми версіями сервера, а також з додатками, де реєстраційний запис вбудований у програмний код. Аутентифікація SQL Server в основному використовується клієнтами, які не можуть бути зареєстровані в домені Windows. Наприклад, це можуть бути користувачі Novell NetWare, Unix та інших систем. Крім того, під час підключення до SQL Server через Інтернет реєстрація в домені не виконується, тому для таких випадків також необхідно використовувати аутентифікацію SQL Server.

Під час інсталяції SQL Server і вибору змішаного режиму автентифікації, майстер установки створює спеціальний обліковий запис SA (системний адміністратор) з порожнім паролем. Цей обліковий запис є членом фіксованої серверної ролі sysadmin і має повний доступ до сервера. Нерідко забувають присвоїти пароль цьому обліковому запису, що створює потенційну вразливість в системі безпеки і відкриває можливість для несанкціонованого доступу до сервера. Хакери перевіряють наявність цієї вразливості у першу чергу. З метою забезпечення безпеки слід спочатку задати пароль для облікового запису SA або відключити його, а потім надати адміністративні привілеї іншому користувачу (або створити додаткові ролі з адміністративними привілеями). Обліковий запис SA залишений з метою забезпечення зворотної сумісності з попередніми версіями SQL Server. У минулому цей обліковий запис був обов'язковим, мав абсолютні права управління сервером і не міг бути видалений. Обліковому запису SA, який створюється під час встановлення, завжди присвоюється однаковий ідентифікатор безпеки 0x01 на всіх комп'ютерах.

Ідентифікатор безпеки зберігається в стовпці SID таблиці sysxlogins у системній базі даних Master. Ця таблиця містить інформацію про облікові записи як SQL Server, так і Windows. Максимальний розмір ідентифікатора безпеки для облікових записів SQL Server становить 16 байт, а для облікових записів Windows - 28 байт. Кожен рядок таблиці sysxlogins відповідає одному обліковому запису, тому ця таблиця може

містити багато рядків з інформацією про облікові записи. Для зручності роботи з локальними обліковими записами можна використовувати представлення syslogins, яке містить лише ті рядки таблиці sysxlogins, де стовпець SRVID (ідентифікатор сервера) має значення NULL.

Для реєстрації користувача використовується системна збережена процедура sp_addlogin.

```
sp_addlogin 'ім'я', 'пароль', 'база_за_замовчуванням ',
'мова_за_замовчуванням', 'ідентифікатор_користувача_сервера',
'параметр_шифрування'
```

Серед переданих аргументів процедурі обов'язковим є лише ім'я реєстраційного запису. Оскільки в цьому випадку потрібні налаштування користувача, а не просто вибір зі списку, виконання цієї процедури складніше, ніж процедура sp_grantlogin. Наприклад, наступний фрагмент програмного коду створює користувача SQL Server з ім'ям **Yevhenii_Kharchenko** і призначає йому навчальну базу даних test в якості бази даних за замовчуванням:

```
EXEC sp_addlogin 'Yevhenii_Kharchenko', 'myoldpassword',
'test'
```

Параметр skip_encryption вказує серверу зберігати пароль у системній таблиці sysxlogins без шифрування. Однак SQL Server очікує, що пароль буде зашифрований, тому він не впізнає пароль, що був створений з використанням цього параметра. Тому краще уникати використання даного параметра.

Якщо потрібно створити одного і того ж користувача на двох серверах, необхідно явно вказати другому серверу ідентифікатор SID, який був присвоєний першим сервером. Для отримання ідентифікатора SID використовується подання sysserver_principals:

```
SELECT Name, SID FROM sysserver_principals WHERE Name =
'Yevhenii_Kharchenko'
Name SID
Yevhenii_Kharchenko 0X1EFDC478FXEB52 045B52D241HG33B2CD7E
```

Пароль може бути змінений за допомогою системної збереженої процедури sp_password, наприклад:

```
EXEC sp_password 'myoldpassword', 'mynewpassword',
'Yevhenii_Kharchenko'
```

Якщо пароль порожній, то в збережену процедуру замість порожнього рядка (' ') передається NULL. Якщо параметр @sid опущений або для нього вказано значення NULL (що також є значенням за замовчуванням для параметра @sid), то збережена процедура sp_addlogin самостійно згенерує ідентифікатор безпеки. Для прикладу створимо обліковий запис із конкретним ідентифікатором безпеки та паролем:

```
USE pubs
EXEC sp_addlogin 'Yevhen','student', @sid =
0x0123456789ABCFEF0123456789ABCDEF
```

У принципі, після створення облікового запису можна змінити ідентифікатор безпеки за допомогою команди UPDATE, безпосередньо звернувшись до системної таблиці. Під час вибору значення для параметра @sid слід дотримуватися вимоги унікальності ідентифікаторів безпеки. Тобто на поточному сервері до моменту реєстрації не повинно бути облікових записів з ідентифікатором безпеки, що дорівнює обраному значенню. Список використовуваних ідентифікаторів безпеки локального сервера можна переглянути за допомогою такого запиту:

```
SELECT sid FROM syslogins
```

Для видалення реєстраційного запису SQL Server використовується системна збережена процедура sp_droplogin, наприклад:

```
EXEC sp_droplogin 'Yevhenii_Kharchenko'
```

Видалення облікового запису призводить до видалення і всіх його налаштувань безпеки.

3.4. Система безпеки рівня бази даних

Для того, щоб користувач мав можливість виконувати певні дії з об'єктами бази даних, йому необхідно мати відповідні права доступу. Власник бази даних або конкретного об'єкта повинен дозволити користувачам звертатися до цих об'єктів. Крім явних прав доступу, що надаються користувачу безпосередньо, існують неявні права доступу. Ці неявні права доступу надаються користувачу через його членство у фіксованій ролі сервера або бази даних. Наприклад, є фіксована роль сервера dbcreator, що дозволяє користувачеві створювати бази даних. Інший приклад неявного права доступу - коли власник об'єкта отримує абсолютні права на управління цим об'єктом. Якщо користувач створює об'єкт у базі даних, йому автоматично надаються повні права на його управління.

Отже, на рівні бази даних можуть бути надані привілеї користувачу шляхом приєднання до фіксованої ролі бази даних або явного призначення привілеїв за допомогою оператора SQL GRANT. Однак, користувачі, облікові записи яких включені до фіксованої ролі сервера sysadmin, є винятком. Члени цієї ролі мають необмежені права в межах сервера, що означає повний доступ до всіх баз даних, що є на сервері.

Перелічимо всі фіксовані ролі бази даних:

db_securityadmin. Члени ролі можуть керувати правами доступу до об'єктів бази даних інших користувачів і членством їх у ролях.

db_owner. Члени ролі мають права власника, тобто можуть виконувати будь-які дії.

db_denydatawriter. Членам цієї ролі заборонено зміну даних незалежно від виданих їм дозволів.

db_denydatareader. Членам цієї ролі заборонено перегляд даних незалежно від виданих їм дозволів.

db_ddladmin. Члени ролі можуть створювати, змінювати і видаляти об'єкти бази даних.

db_datawriter. Користувачі, включені в цю роль, можуть змінювати дані в будь-якій таблиці або поданні бази даних.

db_datareader. Користувачі, включені в цю роль, можуть читати дані з будь-яких таблиць і подань бази даних.

db_backupoperator. Члени ролі виконують резервне копіювання бази даних.

db accessadmin. Члени ролі мають право керувати користувачами бази даних: створювати, видаляти і змінювати.

Користувацькі ролі виступають як додаткові групи ролей. Ролям можна надати доступ до об'єктів бази даних, а користувачеві можна призначити користувацьку роль бази даних. Усі користувачі автоматично стають членами стандартної ролі бази даних «public».

Включення користувачів до ролі реалізується шляхом неявного призначення привілеїв. Явні дозволи на об'єкти надаються за допомогою SQL-інструкцій GRANT, REVOKE і DENY, і вони можуть бути досить деталізованими. Для кожної можливої дії (SELECT, INSERT, UPDATE, RUN тощо) над будь-яким об'єктом існують окремі дозволи. Заборона привілею переважає його надання, а надання привілею переважає його відкликання. Користувачеві може бути надано багато дозволів на об'єкт (індивідуальних, успадкованих від ролі або забезпечених членством у ролі «public»). Деякі фіксовані ролі бази даних також впливають на доступ до об'єкта, контролюючи можливість читання та записування інформації в базу даних.

Існує можливість, що користувач був розпізнаний в SQL Server, але у нього немає доступу до жодної з баз даних. Також можлива зворотна ситуація, коли

користувачу відкрито доступ до баз даних, але сервер не розпізнає його. Переміщення бази даних та її дозволів на інший сервер без одночасного переміщення реєстраційних записів сервера може призвести до виникнення таких "осиротілих" користувачів.

У кожній базі даних автоматично створюються два користувачі:

- **dbo (власник бази даних):** це спеціальний користувач, який володіє базою даних і має повні права на її управління. Користувача dbo не можна видалити. За замовчуванням, користувач dbo відображається як обліковий запис SA, якому надаються максимальні права в базі даних. Крім того, всі члени ролі db_owner також вважаються власниками бази даних. Користувач dbo включений у роль db_owner і не може бути вилучений з неї.

- **guest:** якщо обліковому запису не надано явний доступ до бази даних, то сервер автоматично відображає його як користувача guest. Користувач guest може бути використаний для надання дозволів на доступ до об'єктів бази даних, що є необхідним для будь-якого користувача. Дозволяючи доступ користувачеві guest, ви, фактично, надаєте аналогічні права доступу всім обліковим записам, які налаштовані на SQL Server. З метою забезпечення безпеки збереження інформації рекомендується видаляти користувача guest з бази даних.

Інформація про користувачів, які були створені 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
```

Створення нового користувача і зв'язування його з обліковим записом виконується за допомогою однієї з двох збережених процедур:

- **sp_adduser.** Ця процедура залишена для забезпечення сумісності з попередніми версіями SQL Server і оперує застарілими поняттями.

- **sp_grantdbaccess.** Ця збережена процедура повністю відповідає поняттям системи безпеки SQL Server і прийшла на зміну попередній процедурі, починаючи з версії SQL Server 7.0.

Процедура sp_grantdbaccess має такий синтаксис:

```
sp_grantdbaccess [@loginame =] 'login' [,[@name_in_db =]
'name_in_db' [OUTPUT]]
```

Право виклику зазначеної збереженої процедури мають члени фіксованої 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 має існувати на сервері. За допомогою параметра @name_in_db вказується ім'я, яке буде присвоєно створюваному користувачеві. Це ім'я має бути унікальним у межах бази даних. Наведений далі приклад ілюструє використання збереженої процедури sp_grantdbbaccess. 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'
```

Видалення користувача виконується за допомогою системної збереженої процедури sp_revokedbaccess, що має синтаксис:

```
sp_revokedbaccess [@name_in_db =] 'name'
```

Право виклику зазначеної збереженої процедури мають члени фіксованої ролі сервера sysadmin і члени фіксованих ролей бази даних db_owner і db_accessadmin. Єдиний параметр процедури визначає ім'я користувача, якого необхідно видалити. Однак, перш ніж зважитися на подібний крок, слід переконатися, що користувачеві не належить жоден 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 достатньо буде виконати команду:

```
EXEC sp_revokedbaccess 'Admin'
```

Щоб включити нового члена у 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
```

Виключення з фіксованої ролі виконується за допомогою процедури sp_droprolemember. Наприклад, для виключення з фіксованої ролі db_accessadmin користувача User1 необхідно виконати таку команду:

```
USE pubs
EXEC sp_droprolemember 'db_accessadmin', 'User1'
```

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

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

або просто `CREATE ROLE <rolename>`

Право виконання зазначеної збереженої процедури мають 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 повного контролю над роллю. Якщо необхідно присвоїти 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 користувацької ролі може також бути і роль програми. Як приклад розглянемо створення користувацької ролі AccessDBUser, власником якої буде користувач guest:

```
USE pubs
EXEC sp_addrole 'AccessDBUser', 'guest'
```

Видалення користувацької ролі здійснюється за допомогою збереженої процедури sp_droprole, що має такий синтаксис:

```
sp_droprole [rolename =] 'role'
```

За допомогою єдиного параметра процедури вказується ім'я користувацької ролі, яку необхідно видалити з поточної бази даних. Однак перед подібною операцією

потрібно видалити з неї всіх членів, для чого можна використовувати збережену процедуру sp_droprolemember. Якщо роль володіє одним або більше об'єктами бази даних, то знищити таку роль буде неможливо. Перш ніж приступити до видалення, необхідно або передати права володіння відповідними об'єктами за допомогою збереженої процедури sp_changeobjectowner, або видалити ці об'єкти. За допомогою наведеного нижче прикладу можна видалити призначену для користувача роль:

```
USE pubs
EXEC sp_droprole 'AccessDBUser'
```

Для перегляду списку явних прав доступу можливий запуск системної збереженої процедури sp_helprotect, що має синтаксис:

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

Наприклад:

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

Ця процедура ефективна лише для перегляду явних привілеїв, призначених за допомогою оператора GRANT. Для перегляду всіх привілеїв, включно з неявними, слід використовувати такий 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, яка має один обов'язковий параметр - ім'я бази даних, що від'єднується:

```
USE [master]
GO
sp_detach_db 'test'
go
```

Після переміщення файлів БД у місце призначення проводиться зворотна процедура. Якщо місце, куди скопійовано файли, відрізняється від каталогу за замовчуванням для сервера на комп'ютері призначення, то потрібно призначити даному каталогу повні права доступу групі SQL Server MS SQL User$ MyPC$SQLEXPRESS (примітка: між символами долара знаходиться ім'я сервера). Якщо сервер запущено від імені деякого користувача, то права призначаються йому, а не вищевказаній групі. Під'єднання БД здійснюється за допомогою збереженої

процедури sp_attach_db. Першим її аргументом йде ім'я БД, потім шлях до файлу БД із розширенням mdf, потім шлях до файлу БД із розширенням ldf, наприклад:

```
use [master]
sp_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 осиротілих користувачів. Покажемо процедуру відновлення на прикладі користувача operator. Спочатку проводиться додавання облікових записів БД.

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

Після етапу додавання облікових записів проводиться відновлення (генерація) їхніх системних ідентифікаторів. Для цього потрібні спеціальні повноваження, які дозволяють реконфігурацію налаштувань сервера. У сервері їх можна отримати, викликавши процедуру sp_configure 'allow update', 1 і потім викликати команду RECONFIGURE WITH OVERRIDE. Системний ідентифікатор отримують за допомогою функції SUSER_SID ('логін'). Наприклад:

```
USE test
GO
sp_configure 'allow update', 1
RECONFIGURE WITH OVERRIDE
GO
USE Test
```

```
UPDATE sysusers SET sid=SUSER_SID('operator') WHERE name
= 'operator'
GO
```

На закінчення відновлюються вихідні налаштування сервера:

```
sp_configure 'allow update', 0
RECONFIGURE
GO
```

Для серверів необхідно запустити сервер у монопольному режимі (ключ -m) і потім провести процедуру оновлення SID осиротілих користувачів.

На закінчення розглянемо приклад створення БД у SQL Server і виконання дій з адміністрування БД. Ці дії виконаємо в консолі, використовуючи клієнтську програму sqlcmd.

```
cmd
sqlcmd -S lab48-sk\SQLEXPRESS
```

Після з'єднання всі команди виконуються в клієнті. Для початку увійдемо в систему з правами адміністратора. Отримаємо список користувачів, включених у 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 непривілейованого користувача RFE\User (база за замовчуванням test):

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

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

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

Додамо користувача RFE\User до списку користувачів бази даних test (з ім'ям user):

```
test USE
go
sp_grantdbaccess 'RFE\User', 'user'
Включимо RFE\User у фіксовану роль БД
sp_addrolemember 'db_datareader', 'user'
```

Додамо ще одного користувача Operator

```
sp_grantlogin 'RFE\Operator'
sp_grantdbbaccess 'RFE\Operator', 'operator'
```

Наддамо йому явні права доступу:

```
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
```

Переглянемо список призначених явних прав доступу:

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

ВИСНОВОК ДО РОЗДІЛУ 3

Архітектура системи безпеки SQL Server грає важливу роль у забезпеченні безпеки даних і захисту баз даних від несанкціонованого доступу та зловживань. Вона пропонує різноманітні механізми та компоненти для забезпечення конфіденційності, цілісності та доступності даних.

Одним з ключових аспектів архітектури є механізм аутентифікації та авторизації користувачів, який дозволяє ідентифікувати користувачів і керувати їхніми правами доступу до баз даних. SQL Server пропонує різні рівні аутентифікації, включаючи Windows-аутентифікацію та аутентифікацію на рівні SQL Server.

Іншим важливим компонентом є ролі та дозволи, які використовуються для управління доступом користувачів до об'єктів бази даних. Різні рівні ролей, такі як серверні ролі і ролі бази даних, надають можливість деталізовано керувати правами доступу та обмежувати функціональні можливості користувачів.

Для забезпечення цілісності даних SQL Server використовує механізми перевірки цілісності, такі як обмеження, тригери та транзакції. Вони дозволяють встановлювати правила, які контролюють введення та зміну даних у базі даних, а також забезпечують консистентність даних.

Особлива увага приділяється також захисту даних у спокійному стані та під час передачі через мережу. SQL Server пропонує механізми шифрування, які дозволяють захистити дані у базі даних та під час їх передачі по мережі.

Загалом, архітектура системи безпеки SQL Server надає широкий набір інструментів і можливостей для забезпечення безпеки даних. Вона дозволяє адміністраторам баз даних налаштовувати рівень безпеки відповідно до конкретних потреб і вимог організації, забезпечуючи захист від потенційних загроз і зловживань.

ВИСНОВКИ

Дипломна робота присвячена дослідженню та аналізу функціональності, ефективності та захисту інформації в сучасних мультимедійних системах у контексті мережевого середовища. Робота включає три розділи, кожен з яких детально розглядає важливі аспекти цієї проблематики.

У першому розділі проведений аналіз Cloud-технологій та засобів моніторингу в хмарних середовищах. Виявлено, що Cloud-технології стають все більш популярними і дозволяють організаціям ефективно використовувати ресурси, масштабувати системи та забезпечувати безпеку даних. Засоби моніторингу, такі як Amazon CloudWatch, Google Cloud Monitoring та Microsoft Azure Monitor, надають можливість контролювати працездатність та ефективність хмарних середовищ.

Другий розділ присвячений захисту інформації в системах та мережах. Була розглянута актуальність застосування безпеки інформації в мережевих базах даних та основні принципи забезпечення інформаційної безпеки. Управління доступом до даних, ідентифікація, автентифікація та авторизація користувачів є ключовими аспектами забезпечення безпеки в системах і мережах.

Третій розділ присвячений архітектурі системи безпеки SQL Server. Були розглянуті рівні безпеки на рівні сервера та бази даних, а також методи аутентифікації засобами ОС Windows та SQL Server. Система безпеки рівня бази даних дозволяє керувати доступом до об'єктів бази даних, що забезпечує додатковий рівень контролю та захисту даних.

У процесі виконання дипломної роботи були виявлені і проаналізовані ключові аспекти функціональності, ефективності та захисту інформації в сучасних мультимедійних системах у контексті мережевого середовища. Дослідження показали, що Cloud-технології надають багато переваг, таких як гнучкість, масштабованість та безпеку, що дозволяє ефективно використовувати ресурси та забезпечувати якісну роботу систем. Засоби моніторингу допомагають забезпечувати контроль та нагляд за роботою систем в хмарних середовищах.

Питання захисту інформації є надзвичайно важливими у мережевих системах. Використання ефективних методів управління доступом, ідентифікації, автентифікації та авторизації дозволяє забезпечити високий рівень безпеки та конфіденційності даних. Архітектура системи безпеки SQL Server забезпечує гнучкість та можливості налаштування, що відповідають потребам організацій у захисті інформації.

Загальний висновок дипломної роботи полягає у тому, що функціональність, ефективність та захист інформації є важливими аспектами сучасних мультимедійних систем у контексті мережевого середовища. Використання Cloud-технологій, а також застосування ефективних методів моніторингу та захисту інформації, забезпечує стабільну та безпечну роботу систем.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Клименко, О. О. Основи цифрової обробки сигналів: навчальний посібник / О. О. Клименко, О. Л. Букрєєва, А. А. Клименко. – Київ: Видавничий дім "Києво-Могилянська академія", 2018. – 336 с.
2. Іванов, В. П., Петрова, О. М. Управління базами даних: підручник. Київ: Видавничий центр "Академія", 2019. 456 с.
3. Smith, J. R., Jones, A. B. Network Database Management Systems: A Practical Guide. New York: Wiley, 2017. 280 p.
4. Brown, M. D., Williams, S. R. Introduction to Database Systems. Boston: Pearson, 2020. 624 p.
5. Панасенко, І. В., Кушнір, О. М. Бази даних: практикум. Київ: Центр учбової літератури, 2017. 296 с.
6. Кузьменко, О. М. Бази даних та засоби їх проектування: підручник. Київ: Видавництво "Наукова думка", 2019. 496 с.
7. Ozsu, T. M., Valduriez, P. Principles of Distributed Database Systems. Springer, 2011. 912 p.
8. Abiteboul, S., Hull, R., Vianu, V. Foundations of Databases. Addison-Wesley, 1994. 678 p.
9. Курочкін, О. М., Ширяєв, В. А. Бази даних: навчальний посібник. Київ: Центр учбової літератури, 2020. 416 с.
10. Мартиненко В. І. Захист мереж інформаційного зв'язку в умовах кібербезпеки: монографія. – К.: Аграр Медіа Груп, 2020.
11. "DDoS Attacks and Defense Mechanisms: Practical Insights and Solutions" - Автор: Shon Harris, Michael Lester, Joshua New (2021).
12. "Security Operation Center (SOC) for Dummies" - Автор: Amy E. McDougall, Brian Kelley (2021).

13. Балабанов О. Г., Лук'янова Н. В., Пархоменко А. А. Інформаційна безпека: теорія та практика: монографія. – К.: ВПЦ «Київський університет», 2020.

14. Самойленко А. А., Єрмолаєв О. І. Комп'ютерна безпека: підручник для студентів вищих навчальних закладів. – К.: НТУУ «КПІ», 2021.

15. "Practical Internet of Things Security: Designing and Building Secure IoT Networks and Cloud Architectures" - Автор: Brian Russell, Drew Van Duren, John Matlock (2021).

16. "Network Security: Private Communication in a Public World" - Автор: Charlie Kaufman, Radia Perlman, Mike Speciner (2021).

17. Львівський С. В., Білоус О. О., Виноградов О. В. Захист комп'ютерних мереж: підручник. – К.: КНЕУ, 2020.

18. "Securing the Cloud: Cloud Computer Security Techniques and Tactics" - Автори: Vic (J.R.) Winkler, Prashant Haldankar (2022).