Міністерство освіти і науки України

«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій

Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
завідувач кафедри кібербезпеки
та захисту інформації
_____ Н.В. Лукова-Чуйко
« » червня 2021р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

дипломної роботи
бакалавра
(назва освітнього рівня)

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)

спеціальність 125 Кібербезпека
(код і назва спеціальності)

освітня програма Кібербезпека
(назва освітньої програми)

на тему: «Створення програмного забезпечення маскування даних»

Виконавець: студент IV курсу, групи КБ-41

_____ **Золотарьов Кирило Михайлович**
(підпис) (прізвище ім'я по-батькові)

	Прізвище, ініціали	**Підпис**
Керівник	Зюбіна Р. В.	

Нормоконтроль	Даков С. Ю.	

Київ 2021

Міністерство освіти і науки України

«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій

Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
завідувач кафедри кібербезпеки
та захисту інформації
_____ Н.В. Лукова-Чуйко
«10» жовтня 2020 р.

ЗАВДАННЯ
на виконання дипломної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

Студенту **КБ-41** (група) **Золотарьову Кирилу Михайловичу** (прізвище ім'я по-батькові)

Тема дипломної роботи Створення програмного забезпечення маскування даних

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема дипломної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №2 від 08.10.2020 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Технологія маскування даних, модифікація інформації

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Необхідно провести аналітичний огляд технології маскування даних, обґрунтувати можливі методи та засоби реалізації, розробити схему порівняння з технологією шифрування даних. Запропонувати модель розробки програмного забезпечення для маскування, а також практичну реалізацію данної технології.

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Розроблено унікальне програмне забезпечення для реалізації технології маскування даних

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання:12 жовтня 2020 року

Завдання видала		Р. В. Зюбіна
	(підпис)	(ініціали, прізвище)
Завдання прийняв до виконання		К. М. Золотарьов
	(підпис)	(ініціали, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	25.01.2021 – 01.02.2021	*виконано*
2	Аналіз літератури	02.02.2021 – 17.02.2021	*виконано*
3	Обґрунтування вибору рішення	18.02.2021 – 20.02.2021	*виконано*
4	Дослідження процесу маскування даних	21.02.2021 – 04.03.2021	*виконано*
5	Аналіз методів маскування	05.03.2021 – 15.03.2021	*виконано*
6	Аналіз проблем при використанні технології маскування	16.03.2021 – 23.03.2021	*виконано*
7	Створення моделі розробки програмного рішення для маскування даних	24.03.2021 – 12.04.2021	*виконано*
	Практична реалізація технології маскування	13.04.2021 – 10.05.2021	
8	Оформлення пояснювальної записки	11.05.2021 – 18.05.2021	*виконано*
9	Підготовка до захисту дипломної роботи	19.05.2021 – 08.06.2021	*виконано*

Завдання видав		Р. В. Зюбіна
	(підпис)	(ініціали, прізвище)
Завдання прийняв до виконання		К. М. Золотарьов
	(підпис)	(ініціали, прізвище)

Термін подання дипломної роботи до ЕК 08 червня 2021 року

РЕФЕРАТ

Дипломна робота «Створення програмного забезпечення маскування даних» складається зі івступу, трьох розділів, загальних висновків, списку використаних джерел, додатків, має 46 сторінок основного тексту та 4 таблиці. Список використаних джерел містить 33 найменування і займає 3 сторінки.

Метою даної роботи є створення програмного рішення для маскування даних.

У роботі проведено аналітичний огляд технології маскування даних, обґрунтовано можливі методи та засоби реалізації цієї технології, розроблено схему порівняння процедур маскування та шифрування, запропоновано модель розробки програмного рішення для маскування, практичну реалізацію процедури маскування даних.

Розроблений лістинг додатку, що реалізує технологію маскування даних, може використовуватися як шаблон або підґрунтя для реалізації власного програмного рішення.

Розроблені модель створення програмного рішення призначена для надання рекомендацій по реалізації індивідуального додатку.

Ключові слова: Конфіденційні дані, захист інформації, безпека в сховищі даних, модифікація інформація, обфускація.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

AES	–	Advanced Encryption Standard
DES	–	Data Encryption Standart
NRDM	–	Non Reversible Data Masking
RDM	–	Reversible Data Masking
MOBAT	–	MOdule Based Technique
DBA	–	Data Base Administrator
FAST	–	Find,Assess,Secure,Test
IT	–	Information Technology
SSL	–	Secure Sockets Layer
SQL	–	Structured Query Language
ПЗ	–	Програмне забезпечення
FPE	–	Format-preserving Encryption

ЗМІСТ

РЕФЕРАТ 4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ 5
ЗМІСТ 6
ВСТУП 8
РОЗДІЛ 1 ПРОЦЕС МАСКУВАННЯ ДАНИХ ТА ЙОГО ОСОБЛИВОСТІ 9
1.1 Структура сховища даних 9
1.2 Безпека та захист даних 10
1.3 Права користувачів, порядок їх доступу до конфіденційних даних 11
1.4 Технологія маскування даних 14
1.4.1 Причини використання технології маскування даних 17
Висновки за розділом 1 17
РОЗДІЛ 2 МЕТОДИ МАСКУВАННЯ ДАНИХ 19
2.1 Традиційний способ маскування даних 19
2.1.1 Техніка обнулення значень 21
2.1.2 Метод перетасовки 22
2.1.3 Техніка старіння числа та дати та числове чергування 23
2.1.4 Метод заміни даних 23
2.2 Реверсивний спосіб маскування даних 26
2.3 Статичний та динамічний способи маскування даних 29
2.4 Цілісність даних при використанні технології маскування 31
2.5 Проблеми при використанні технології маскування даних 32
Висновки за розділом 2 34
РОЗДІЛ 3 СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МАСКУВАННЯ ДАНИХ 36
3.1 Існуючі програмні рішення для маскування даних 36
3.1.1 Рішення на основі надбудови Excel 37
3.1.2 Веб-додатки 38

3.1.3 Стороннє рішення - додаток 39
3.1.4 Недоліки використання готових додатків 40
3.2 Створення програмного рішення для маскування даних..... 41
3.2.1 Оранізаційні питання процессу розробки..... 41
3.2.2 План розробки додатку та технічне завдання..... 42
3.2.3 Структура додатку 45
3.2.4 Демонстрація роботи программного рішення 51
Висновки за розділом 3 53
ВИСНОВКИ 54
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 55
ДОДАТОК А 58
ДОДАТОК Б..... 60

ВСТУП

В сучасних реаліях, дані, що використовуються та обробляються у системі – це найцінніший актив будь-якої організації. Згідно цього, безпека даних є одним з найважливіших завдань на підприємстві. Внаслідок розвитку інформаційних технологій надзвичайно збільшується обсяг інформації та даних, що потребують обробки.

Інформація зберігається в базах даних, що можуть бути зв’язані між собою, або існувати окремо. Робота з базами передбачає взаємодію з інформацією, а це, в свою чергу, означає, що доведеться мати справу як з відкритими даними, так і з конфіденційними. Завдяки достатньо великій кількості даних та потребі їх обробки у режимі реального часу, виникає проблема забезпечення захисту цієї інформації. Як правило, коли розробляється база даних, дані використовуються безпосередньо з виробничої бази, яка містить реальні або правильні дані, а не довільно сгенеровані, або модифіковані. Дані, що містяться у відкритому вигляді, створюють потенційні проблеми для підприємства, тому що, при витоку або несанкціонованому доступі до них, зловмиснику не потрібно розшифровувати або демодифікувати дані. Також проблема полягає у тому, що при користуванні загальною базою даних, користувачі, яким не надано доступ до конфіденційної інформації, матимуть можливість побачити її.

Найкращим рішенням для досягнення належного рівня безпеки інформації на підприємстві буде використання технології маскування даних, тобто, мінімізації випадкового та навмисного розкриття інформації, що обробляється в системі. Головною метою цієї технології є забезпечення недоступності доступу до даних, шляхом модифікації цих же даних, або їх зміною.

РОЗДІЛ 1
ПРОЦЕС МАСКУВАННЯ ДАНИХ ТА ЙОГО ОСОБЛИВОСТІ

1.1 Структура сховища даних

Сховище даних – це система, яка впроваджується та використовується для аналізу інформаційних активів організації та створення звітів. Сховище даних на підприємстві може включати в собі декілька окремих або зв’язаних баз даних. На рисунку 1 наведено загальну модель різних баз, що об’єднані в одну

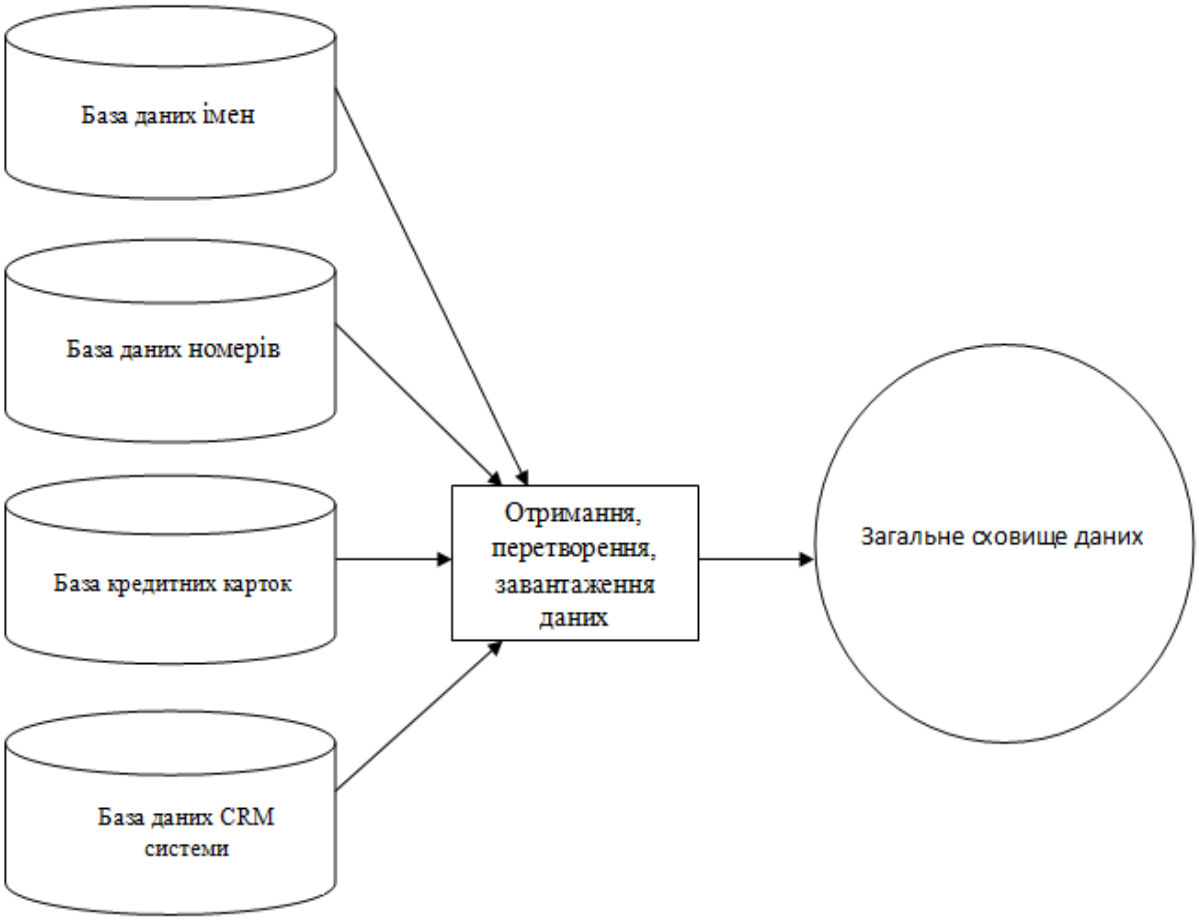

Рисунок 1.1 - Структура загального сховища даних

Загальна база даних потрібна, щоб всі потрібні для роботи дані зберігалися в одному місці. Це дуже спрощує процес створення звітів та аналітичну роботу.

Така архітектура сховища даних використовує багатошарові схеми, це означає, що існує основний рівень , де зберігаються всі дані. Далі йде презентаційний рівень, в якому містяться зв’язки таблиць та їх відношення один до одного. І на найнижчому рівня йдуть окремі таблиці, що містять безпосередньо використовуванідані кожною з окремих таблиць, без зазначених зв’язків між ними. Нижче на схемі наведено приклад структури сховища даних, основаної на трьох рівнях.

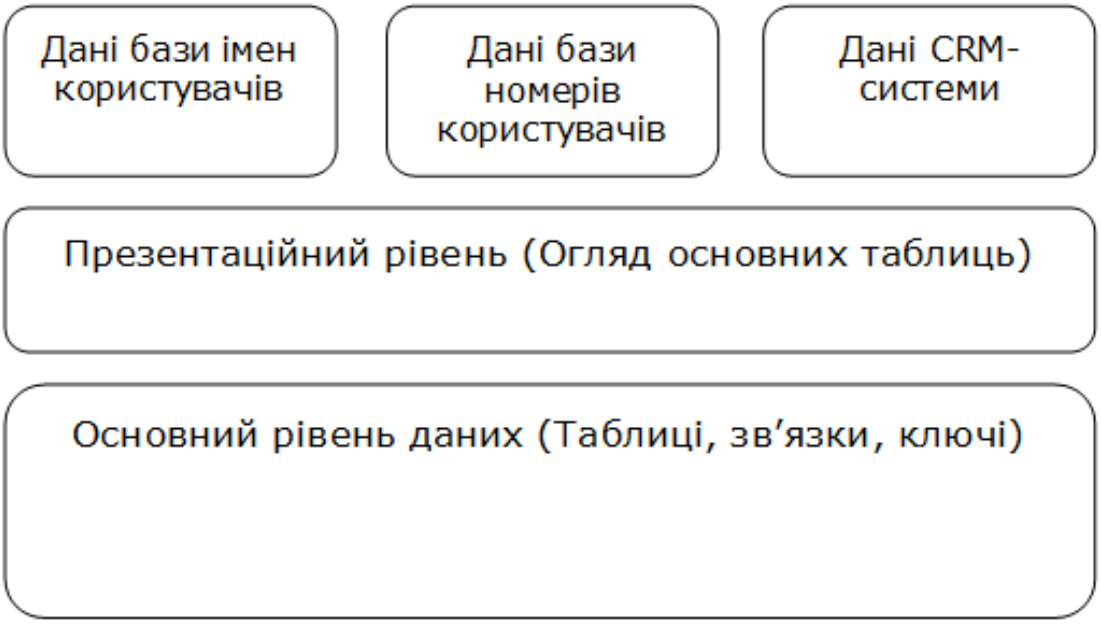

Рисунок 1.2 - Трьохрівнева структура представлення сховища даних

1.2 Безпека та захист даних.

Безпека даних є важливим завданням для будь-якої організації. Особисті файли, дані банківського рахунку або будь-яка інша інформація про клієнта – це такий тип даних, які досить важко відновити та замінити без зробленого зазадлегідь резервного копіювання. Буває , що дані можуть бути втрачені внаслідок стихійних лих, таких як повінь чи пожежа, це ситуації, де від людського фактору нічого не залежить, але якщо дані викрадені та потрапили до рук хакерів, то наслідки такої ситуації можуть бути набагато гірші.

Захист бази даних означає беспосереднє знання ризиків, які можуть бути як віртуального характеру, так і фізичного. Також слід зазначити, що дуже важливо мати компетентних та підкованих робітників, які працюють з даними, Для будь-яких випадків можливих небезпек потрібно мати план, у якому зазначені основні загрози, їх опис, та способи реагування, тобто модель загроз.

Захист даних на підприємстві ускладнений через комплексну систему, в якій налічується достатньо велика кількість користувачів. Загальна база містить дані в основному про всі частини та всі ланки організації, це означає, що там може міститися конфіденційна інформацію, яка є, наприклад, особистою інформацією клієнтів та працівників. Згідно законів України, конфіденційна інформація не повинна бути розголошена, тому її захист є пріоритетним.

Кожен співробітник, що має доступ до сховища даних, повинен отримувати доступ лише до тих даних, які необхідні йому для роботи. Згідно цього, повинен бути затверджений спосіб керування доступом до бази. Це реалізується шляхом створення ролей для різних групп, рівнів та об'єктів для доступу до потрібної працівникові інформації.

Якщо мова йде про розробників бази даних, які працюють у середовищі розробки, і про тестерів, які працюють у тестовому середовищі, то вони бачать всі дані бази, тому що їм потрібно переконатися, що система працює адекватно, а дані, що містяться в базі, є правильними.

1.3 Права користувачів, порядок їх доступу до конфіденційних даних

База даних є центром організації як збереження конфіденційних даних, а також інформаціїї про співробітників та клієнтів компанії. Існує можливість потрапляння таких даних під атаку, тому коли відбувається спроба порушення конфіденційності даних, потрібно мати модель загроз та реагування на них, в якій буде прораховано можливі ризики та їх наслідки від спроби порушення даних

Для кожної організації, одним з найважливіших завдань безпеки є забезпечення цілісності важливих даних, а також важливим фактором у цьому є

створення технічних та фізичних завад. Захист бази даних означає повне розуміння того, хто використовує базу даних, до чого користувачі можуть отримати доступ в базі даних. Для невеликої організації, що налічує у своєму складі лише 20 працівників, що мають доступ до бази, захист, як правило, не є проблемою, і ним легко управляти, не приділяючи особливої уваги до управління правами користувачів, але коли у організаці, де до бази даних мають доступ сотні користувачів, які можуть мати дуже різні потреби в використанні певної інформаціїї, тоді необхідно мати загальний підхід або спосіб управління їх правами.

Впорядкування прав користувачів є досить складним процесом і займає багато часу. Все починається з заповнення форми, де і які права знадобляться певному користувачеві, після цього запит повинен бути прийнятий безпосереднім керівником особи, яка замовила права. Наступним етапом є передача цього запиту IT-службою до адміністратора бази даних, який вручну надає права користувачеві на певні об'єкти. Адміністратор бази даних також повинен підтримувати права користувача в подальшому, навіть якщо користувач вийшов з організації або більше не потребує прав. Процес упорядкування прав користувачів показаний на рисунку 1.3.

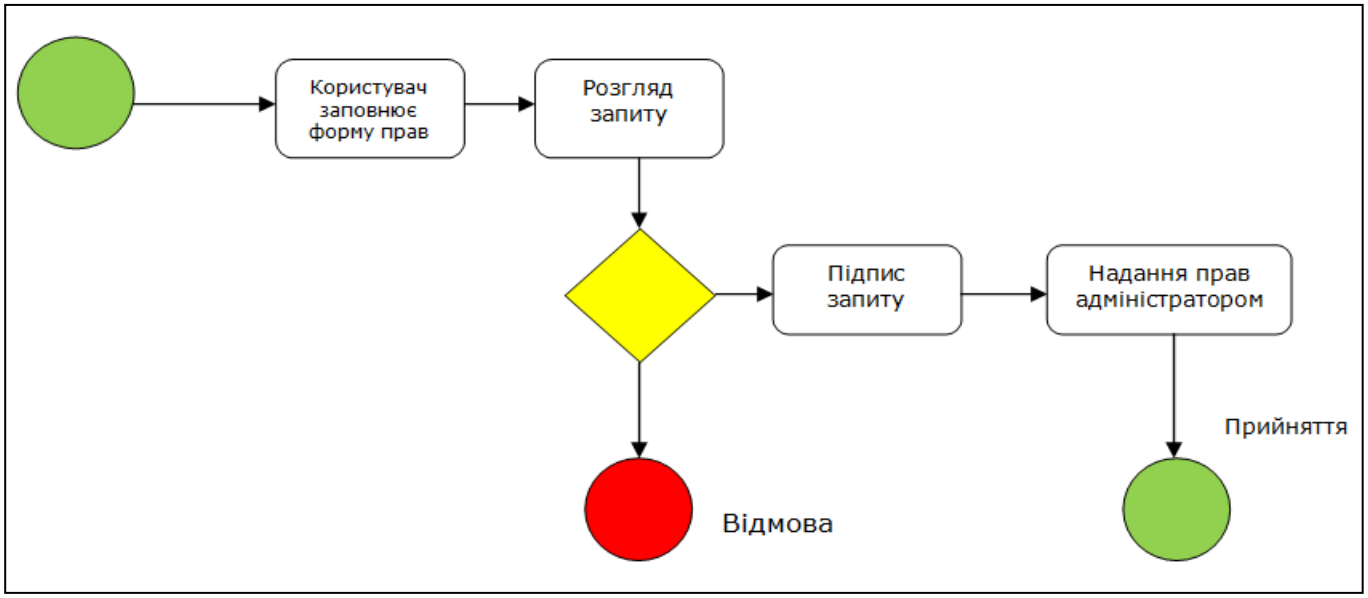

Рисунок 1.3 - Процес упорядкування прав користувачів

Існує ряд методів, які створенні для спрощення вищеописаного процесу. Перший метод - використовувати права користувачів на основі проекту. Користувачам, як правило, потрібні права доступу, пов’язані з якимось новим або існуючим проектом. Зазвичай, коли починається новий проект, багатьом співробітникам потрібно замовляти права. Наприклад, у нас є 10 тестувальників, яким потрібно замовити права користувачів на доступ до нових даних, пов'язаних з проектом. Кожному користувачеві потрібно заповнити форму, отримання та схвалення якої може зайняти день або більше часу, кожен запит повинен бути розглянутий, а адміністратор повинен виконувати кожне завдання окремо.

Пропозиція полягає в тому, що права замовлення можуть здійснювати менеджери проекту для всіх користувачів, яким це необхідно. У цьому випадку ми можемо деталізувати процесс сбору інформаціїї для запиту, додавши деталь, як довго потрібні права, менеджер проекту замовить права на певний період, і якщо виконання проекту займе більше часу ніж очікувалося то , права користувача можуть бути продовжені.

Користувач, що замовлює:	Ім'я менеджера проекту
Користувачі, які потребують прав:	User1, User2, User3
База даних	Цільова база
Назва ролі користувача:	Project_XX11
Дата закінчення:	2022-09-30

Рисунок 1.4 - Форма замовлення прав для групи користувачів

Теоретично можливо пропустити частину затвердження, оскільки керівник проекту відповідає за нього, а значить, регулює права та доступ співробітників.. Процес замовлення прав менеджером проекту показаний далі на рисунку 1.5.

Рисунок 1.5 - Спрощенний процес отримання прав користувачами

Якщо не створювати запит на отримання прав для кожного співробітника окремо то, виграш у часі і простоті може бути в 100 разів більшим. Фактично, за один прохід процедури, ви надаєте доступ до конфіденційних даних усім, потребучим цього користувачам. Це також зменшує об'єм роботи служби технічної підтримки, яка призначає запити та створює завдання, а також роботу адміністраторів, які можуть виконати 1 завдання замість 10.

1.4 Технологія маскування даних

Ознайомившись з описаною у попередніх пунктах інформацією, потрібно сформувати процес забезпечення конфіденційності чутливої інформації, таким чином, щоб дані не були відкритими, не могли ідентифікувати особу, але при цьому ж база даних використовувала реальні дані. Для вирішення цієї проблеми можна використовувати різні види методів маскування даних та управління правами користувача, що є темою цієї дипломної роботи.

Маскування даних - це процес заміни чутливих даних, таких як номери кредитних карток, на підроблені, тобто замінені за певним методом, але реалістичні дані. Як правило, основною метою маскування даних є щось схоже на криптографію, в тому сенсі, що маскування даних робить конфіденційну інформацію недоступною для неавторизованих користувачів, але на відміну від

криптографії, немає необхідності у важких та довгих обчисленнях та великої кількості повторюваних операцій. Маскування даних також можуть називати як де-ідентифікацію, затуманення, анонімізація даних

Для кращого розуміння поняття маскування даних, необхідно спочатку коротко розглянути процесс криптографічного захисту інформації. Потім, проаналізувати його та знайти основні відмінності між криптографічним захистом інформаціїї та процессом маскування даних.

Криптографія - це наука «секретних кодів», дані шифруються за допомогою криптосистем або шифрів. Шифр використовується для шифрування або “кодування” повідомлення чи інформації, що називається “відкритим текстом”, результат цього процесу шифрування називається зашифрованим текстом. Ціль шифрування полягає у захисті повідомлення, для того, щоб лише уповноважені сторони, які мають секретний ключ, могли отримати до нього доступ. Існує два основних типи криптосистем:

- криптосистема симетричного ключа, яка використовує той самий ключ для шифрування та розшифруванн
- криптосистема з відкритим ключем, що використовує відкритий ключ для шифрування повідомлення та закритий ключ для розшифровки.

Найбільш поширені та безпечні симетричні алгоритми ключів, наприклад, AES та DES, залежать від поєднання лінійних та нелінійних операцій заміщення, перестановки та зсуву, що виконуються на всіх раундах заміни, залежно від розміру ключа.

Загалом, слід додати, що криптографія потребує складних обчислювальних можливостей на додаток до додаткового сховища та, порівняно з маскуванням даних, довгий час обробки інформаціїї.

Існує два типи маскування даних:

- традиційне маскування даних, яке також називається незворотним маскуванням даних (NRDM)
- зворотнє маскуванням даних. (RDM)

На рисунку 1.6 наведено порівняння процесів шифрування та маскування даних.

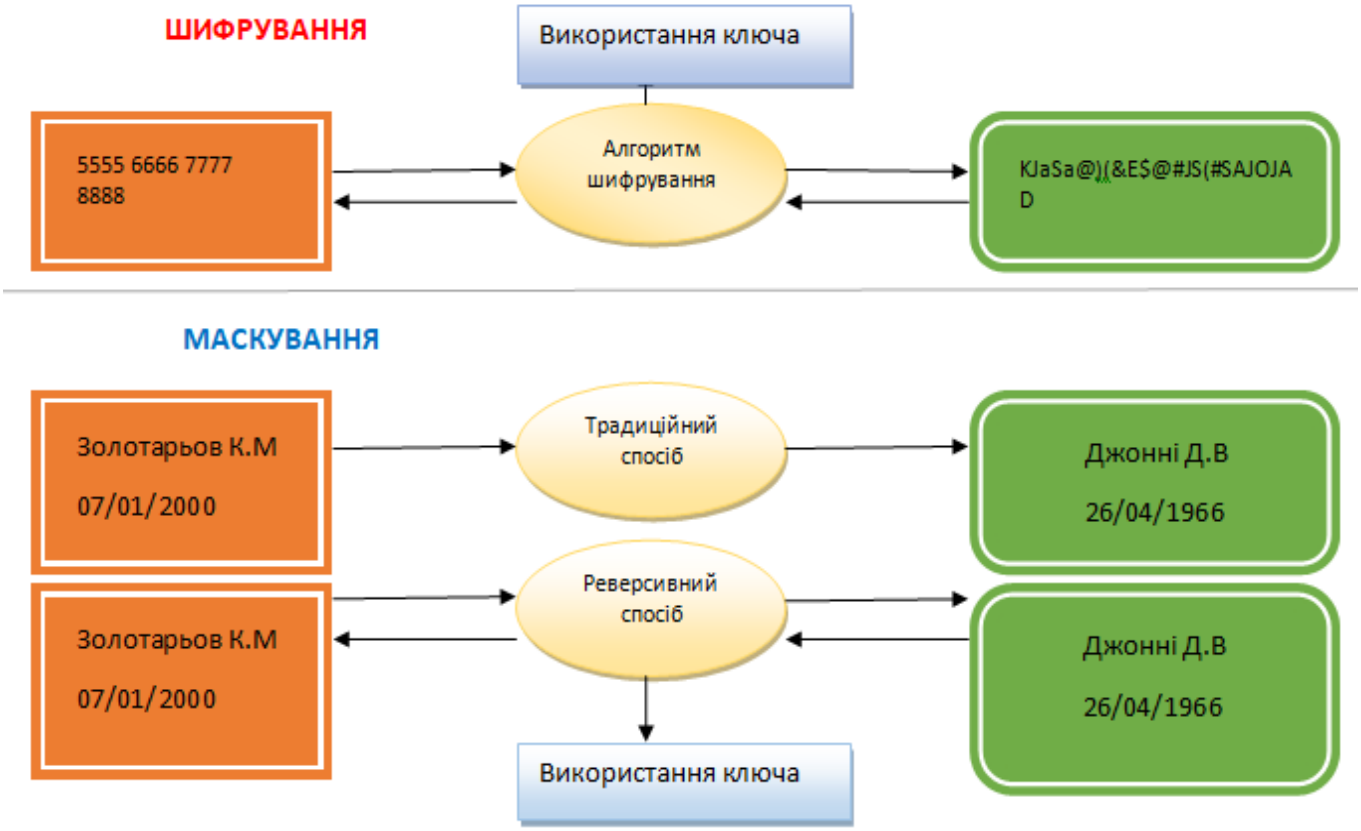

Рисунок 1.6 - Порівняння процесів шифрування та маскування даних

Визначивши основні особливості процесів шифрування та маскування даних, можна стверджувати, що маскування як процес не передбачає можливості відновлення даних, тобто інформація завжди буде збережена у модифікованому стані. Якщо детальніше розглянути процес реверсивного маскування даних, то дійдемо до висновку, що при такому способі, відновлення даних можливе, так само, як і в процесі шифрування даних.

Традиційне маскування використовується без можливості відновлення даних для того, щоб повністю відкинути можливість розкриття даних при їх незаконному заволодіванні

1.4.1 Причини використання технології маскування даних

Коли в організації або на підприємстві містяться дані, які є конфіденційними або ідентифікують особу, то вони є пріоритетними та потребують особливої уваги, оскільки є вразливими. Порушення безпеки даних може статися через системну помилку або несправність, або коли хтось викрадає дані. Це може бути, як працівник, що має доступ до них, так і злодій, який атакує ззовні організації. Щоб зменшити ризик порушення безпеки даних, можна використовувати методи маскування даних.

Дані, для яких було використанно технологію маскування також дають кращий огляд інформації про те, хто їх використовує. Огляд конфіденційної інформації та можливість підтримувати використання конфіденційних даних повинні бути в кожній організації, оскільки вона повинна забезпечувати збереження та захист інформації відповідно до законів про конфіденційність та нормативних актів. Також відомості про те, хто використовує конфіденційні дані, допомагає запобігти порушенню з боку внутрішніх користувачів, які можуть використовувати дані для своїх інтересів.

Порушення безпеки конфіденційної інформації несе за собою достатньо істотні наслідки для будь-якої організації, тому процес вирішення проблем після порушення безпеки є досить складним. З'являється потреба в технічному огляді системи безпеки, проведення організаційних та технічних робіт з пошуку вразливості, та причини інциденту. Також можливо знадобиться розробка нового алгоритму захисту та системи безпеки, щоб усунути причину порушення безпеки даних. Тому, для більш надійного захисту та мінімізації ризику втрати конфіденційних даних використовують технологі маскування даних.

Висновки за розділом 1

Необхідність захисту інформаційних активів та конфіденційних даних на будь-якому підприємстві пояснюється можливими негативними наслідками для

загального сховища даних та, в цілому, платформи, що використовується. Підходи до захисту інформації можуть бути абсолютно різними, але слід виділити технології шифрування та маскування даних, як найефективніші та широко розповсюджені методи.

Маскування даних – спосіб, що робить викриття реальної інформації майже неможливим. На відміну від шифрування, маскування займає набагато менше часу, простіше у своїй реалізації, та не потребує серйозних обчислювальних можливостей.

Для забезпечення належного рівня безпеки за допомогою технології маскування інформації, необхідно розуміти структуру та вміти працювати з загальним сховищем даних на підприємстві. Також важливим фактором є визначення прав груп користувачів, та окремих осіб, порядку їх доступу до чутливої інформації.

Маскування даних, як технологія, корисна не тільки для захисту даних, а ще й для створення аналітичних звітів, тому що, при використанні данного підходу, існує можливість моніторингу користувачів, які працюють з модифікованою інформацією.

РОЗДІЛ 2
МЕТОДИ МАСКУВАННЯ ДАНИХ

Існують різні способи, як замаскувати дані, оскільки маскування - це процес зміни вхідного значення на певну модифіковану інформацію на виході, то тип та метод маскування буде напряму залежати від типу інформації та потреби відновлення даних. Наприклад, вхідне значення можна замінити, змінити, відредагувати, просто показавши, наприклад, символ «*». Також вхідне значення може бути перемішане з безліччю інших значень у стовпці. Вхідні значення можуть бути розмиті, за рахунок однойменної техніки. Можливе також і перетворення значення в атрибут, що належить певному діапазон значень. Також існують техніки, що усереднюють значення.

2.1 Традиційний способ маскування даних

Традиційний спосіб маскування даних - це алгоритм, при якому дані маскуються та не можуть бути віднайдені та відновлені. Після процесу маскування даних, шанси на те , що вони можуть бути викрадені та відновлені у первинному вигляді, практично не має. Якщо розробнику бази або будь-якому користувачеві, якому надано відповідний рівень доступу, потрібно відновити дані, замасковані за допомогою нереверсивного методу, то процес їх відновлення буде залежати від того, на якому рівні загального сховища даних вони були замасковані. Якщо процес маскування даних відбувся при використанні дочірньої бази, яка має встановлений зв’язок з батьківською, то дані відновити можливо, маючи доступ до обох баз. Якщо ж маскування виконувалось безпосередньо в основній базі, то такі дані відновити буде неможливо. В цьому і полягає суть нереверсивного маскування. Майже всі існуючи та широкорозповсюджені методи маскування даних відносяться до нереверсивного способу. Нижче наведено алгоритм, та коротке представлення даних у замаскованому вигляді.

Таблиця 2.1 - Традиційні методи маскування даних

№	Метод	Опис	Приклад	
			Вхідні дані	Вихідні дані
1.	Випадкова заміна (вигадані дані)	Цей метод замінює вхідні дані подібними до них випадковими значеннями із заздалегідь підготовленого набору.	Джон Говард	Джім Керрі
2.	Старіння дати	У цій техніці атрибут дати або збільшується, або зменшується. Однак діапазони дат повинні бути визначені в допустимих межах, щоб зміна даних не надто сильно змінювала вхідне значення.	25.05.2021	05.05.2020
3.	Числове чергування	У цьому підході, на основі прийнятного відсотка або діапазону, числовий атрибут буде відповідно збільшений або зменшений..	2030	1980
4.	Перестановка даних	У цій техніці атрибут даних буде використовуватися як одиниця набору даних для заміщення. значення перемішуються між рядками таким чином, всі значення залишаються у рядках з порушеною послідовністю.	CODE3M	CODE2F
5.	Техніка обнулення значень	Техніка маскування дезінфікує дані, замінюючи певні вказані символи маскованими символами	5168 5555 6666 7777	5168 #### #### 7777

2.1.1 Техніка обнулення значень

Техніка обнулення оброблює дані таким чином, що певні вказані символи, замінюються символами маски. Наприклад, номер кредитної картки може бути замаскований таким чином:

Вхідне значення (Номер карти)	Вихідне значення (замасковані дані)
«4929 1344»	XXXX XXXX.

Рисунок 2.1 - Техніка обнулення значень

Маскування даних, крім того, що є загальним терміном для процесу анонімізації даних, що в свою чергу, як раз і означає заміну певних даних символом маски (наприклад, як показано вище символом – «X»). Це ефективно маскує вміст даних, зберігаючи ту саму структуру, кількість символів. Також, стовпець з номерами кредитних карток може виглядати інакше, тобто маскування відбувається не для всіх символів поля «номер карти». Наприклад:

Вхідне значення (Номер карти)	Вихідне значення (замасковані дані)
4346 6454 0020 5379	4346 XXXX XXXX 5379
4493 9238 7315 5787	4493 XXXX XXXX 5787
4297 8296 7496 8724	4297 XXXX XXXX 8724

Рисунок 2.2 - Обнулення більшої частини символів

Маскувальні символи видаляють переважно більшу частину конфіденційних даних з запису, зберігаючи при цьому реальний вигляд.

Важливо зазначити, що не складно буде відновити оригінальний номер кредитної картки з маскування наприклад : 4297 8296 7496 87XX. Також слід подбати, щоб не замаскувати потенційно необхідну інформацію. Операція

маскування, така як XXXX XXXX XXXX 5379, позбавить дані емітента картки номера кредитної картки.

2.1.2 Метод перетасовки

Техніка перетасовки використовує існуючі дані як підгтовлений набір для заміщення та переміщує значення між рядками таким чином, щоб у обраному рядку було відсутнє реальне значення, тобто було замінене на значення іншого. Цей спосіб подібний до методу заміни, за винятком того, що дані заміни отримуються з самої таблиці бази даних. По суті, дані в стовпці випадковим чином переміщуються між рядками, доки більше не буде розумного співвідношення з рештою інформації в рядку. Існує певна небезпека в техніці перетасовки. Іншими словами, оригінальні дані все ще є.

Іншим фактором є алгоритм, що використовується для перетасовки даних. Якщо метод перетасовки можна визначити, то потім дані можна легко "відтасувати" обернено. Наприклад, якщо алгоритм перетасовки просто пробіг по таблиці, помінявши дані стовпців між кожною групою з двох рядків, зацікавленій стороні не знадобиться багато роботи, щоб повернути дані у неперетасований стан.

Перетасовка рідко ефективна при використанні на невеликих обсягах даних. Наприклад, якщо в таблиці лише п’ть рядків, можливо, буде не надто складно з’ясувати, який рядок із п’яти перемішаних справді належить до вибраного рядка.

Правила перемішування найкраще застосовувати на великих таблицях, тому що зовнішній вигляд даних залишається незмінним. Цей метод дуже швидкий та нескладний у плані обробки даних, але з великою обережністю потрібно застосовувати його для перемішування рядків

Навіть сьогодні у всіх галузях застосовується метод перетасовки даних. Але знову ж недолік тут стикається з величезним обсягом даних та закономірністю перетасовки.

2.1.3 Техніка старіння числа та дати та числове чергування

Техніка старіння числа та дати змінює існуючі значення у визначеному діапазоні. Наприклад, значення дати народження можуть бути змінені в межах +/- 60 днів. Техніка корисна для числових даних або даних дати. Алгоритм передбачає модифікацію кожного числа чи значення дати у стовпці на деякий випадковий відсоток від його реального значення. Ця техніка має перевагу, забезпечуючи розумне маскування даних, зберігаючи діапазон і розподіл значень у стовпці в існуючих межах .

Наприклад, у стовпці даних про заробітну плату може бути розміщена випадкова дисперсія ± 10%. Деякі значення були б вище середніх, інші нижчими, але все ж таки ці значення були б не надто далекими від їх початкового діапазону.

Поля дати також є хорошим прикладом для цього методу. Наприклад, дати народження можуть варіюватися в довільному діапазоні ± 120 днів, що ефективно маскує особисту інформацію, зберігаючи розподіл.

2.1.4 Метод заміни даних

Цей спосіб замінює наявні дані випадковими значеннями із заздалегідь підготовленого набору даних, тобто ця методика складається із випадкової заміни даних стовпця інформацією, яка виглядає схожою, але абсолютно не пов'язаною з реальними деталями. Наприклад, прізвища в базі даних клієнтів можна модифікувати, замінивши справжні прізвища на прізвища, взяті з випадкового списку. Заміна є дуже ефективною з точки зору збереження зовнішнього вигляду існуючих даних. Недоліком є те, що для кожного заміненого стовпця повинен бути наявний великий запас інформації, яку можна замінити. Наприклад, для модифікації прізвищ шляхом заміни повинен бути доступний список випадкових прізвищ.. Дані заміщення іноді буває дуже важко знайти у великих кількостях, однак будь-яке програмне забезпечення для маскування даних повинно містити набори даних

загальнообов'язкових елементів. При оцінці програмного забезпечення для маскування даних слід враховувати розмір, обсяг та різноманітність наборів даних.

Ще однією корисною функцією, яку потрібно шукати, є можливість створювати власні набори даних та додавати їх для використання в правилах маскування на конкретному підприємстві

2.1.5 Техніки маскування даних, запропоновані на основі дослідження традиційних методів

- Професор університету Адічунчанагири доктор РавиКумар Г.К у 2011 році дослідив найкращі рішення щодо маскування баз даних. Враховуючи обсяг даних, які зазвичай обробляються запитами у сховищі, існуючі рішення шифрування на його думку значно збільшували час відгуку на запит. Він запропонував рішення для маскування даних для числових значень в сховищі на основі оператора математичного модуля (MOBAD), яке можна використовувати з додатковим рівнем програмного забезпечення (не вбудованим у платформу BI). Алгоритм маскування на основі модуля застосовує математичний оператор модуля до числових атрибутів. Він базується на формулі, яка залежить від трьох маскуючих клавіш: дві з них зашифровані приватним ключем, недоступні для будь-яких користувачів, включаючи DBA), а третій ключ відкритий.
- Г.Сарада – ще один індійський професор, який у 2015 році запропонував кілька нових підходів до маскування даних, таких як Мінімально-Максимальна нормалізація, яка виконує лінійне відображення вхідних даних у новий діапазон з нижньою та верхньою межами, що відповідають діапазону розглянутого атрибута. Головною перевагою цього підходу є те, що він підтримує точність між вхідними та замаскованими даними. Крім того, він запропонував метод захисту, який в основному застосовується до даних, що містять категорії, де вхідні дані записуються по рядках / стовпцях, а перетворені дані отримуються шляхом переміщення уздовж стовпців / рядків відповідно.

- Запропоноване компанією Oracle Inc, програмне рішення Oracle Data Masking допомагає зменшити ризик незаконного отримання та розкриття даних, замінюючи вхідні дані вигаданим набором. Oracle розробив комплексний 4-кроковий підхід при використанні технології маскування, який називається «Знайти, оцінити, захистити та перевірити» (FAST). Ці кроки: 1). Знаходження інформаціїї; 2). Оцінка інформаціїї; 3). Захист інформаціїї; 4). Перевірка функціональності
- На основі IBM InfoSphere Optim та DataStage IBM розробила і інтегрувала платформу для захисту та перевірки дотримання правил конфіденційності та управління чутливою інформацією у виробничих та невиробничих базах даних. Адміністратори бази можуть обирати метод, який буде застосований для маскування даних, щоб замінити конфіденційні дані реалістичними вигаданими даними. IBM Optim та DataStage мають у своєму функціоналі арифметичні вирази, генерацію випадкових чи послідовних чисел, методи старіння дати та заміну. модель контекстної зрозумілості

Таблиця 2.2 - Аналіз запропонованих технік маскування даних

Запропонована техніка маскування	Реверсивний/не реверсивний спосіб	Статичний/Динамічний метод	Окремий додаток/Вбудована в систему можливість
Технологія маскування даних від Oracle corp.	Нереверсивний	Статичний	Вбудована можливість
Технологія запропонована IBM	Нереверсивний	Статичний та динамічний	Вбудована можливість
MOBAD	Реверсивний	Статичний та динамічний	Окремий додаток
Мінімально-Максимальна нормалізація	Реверсивний	Статична та динамічна	Окремий додаток

2.2 Реверсивний спосіб маскування даних

Це алгоритм, при якому дані маскуються та можуть бути віднайдені та відновлені. Масковані за цим способом атрибути так само як і при традиційному маскуванні виглядають реалістично, але основна відмінність полягає в тому, що при використанні реверсивного способу користувач може ідентифікувати вхідні дані в основній базі. Реверсивний спосіб маскування реалізується за допомогою таких методів:

1 - Псевдонімізація (методи довільної перетасовки та заміни): метод перемішування атрибутів даних для збереження конфіденційності, що включає маскування певних атрибутів набору даних, які слід зберегти в конфіденційності, з використанням методу перетасовки, що включає також і сортування вже зміненого набору даних та перетворений конфіденційний атрибут відповідно до встановлених критеріїв порядку ранжування.

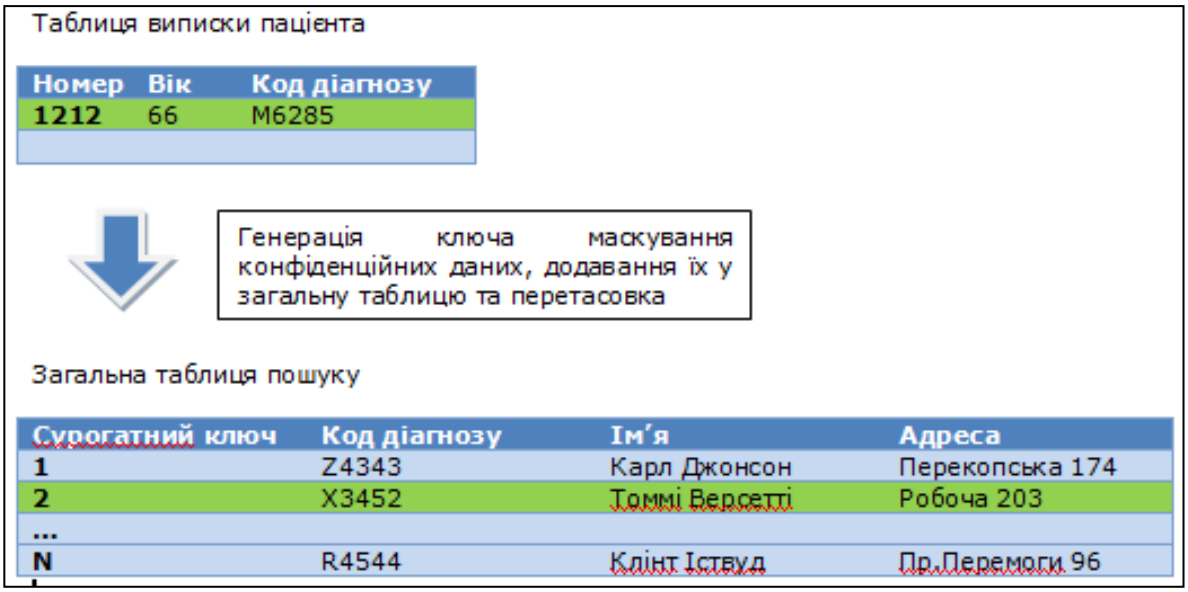

Рисунок 2.3 - Метод псевдонімізації

2 - Шифрування, що зберігає формат (FPE): Цей метод відрізняється своїм призначенням від найпоширеніших методів шифрування, які використовують розширений алгоритм, подібний до шифрування AES, щоб зберегти вхідний формат даних відкритого тексту, так, щоб вони здавалися реальними, Наприклад, код

діагностики “M6285” може бути перетворений на “Z2138”, зберігаючи однакову буквено-цифрову послідовність і формат.

Рисунок 2.4 - Шифрування зі збереженням формату

3 - Метод реверсивного маскування запропонований Ріком Добсоном. Основна ідея цього рішення полягає у використанні таблиці відображення, де визначено значення для кожного заміненого символу. Наприклад, якщо ми маємо число, що має цілочисельний номер типу 11, тоді ми зберігаємо для кожної позиції замінне значення для кожного числа. Наприклад, перші 2 числа - 83, і в таблиці маскування для позиції 1 і цифри 8 ми замінюємо її на число 0, а для позиції 2 - номер 3 - на 6, і в цьому прикладі ми маємо число 06 як замасковане значення для числа 83. Далі буде наведено приклад таблиці відображення за методом Ріка Добсона, що містить такі атрибути:

- Позиція
- Вхідні дані
- Масковані дані

Таблиця 2.3 – Таблиця відображення за методом Добсона

Позиція	Вхідні дані	Масковані дані
1	0	8
1	1	7
1	2	6
1	3	5
1	4	4
1	5	3
1	6	2
1	7	1
1	8	0
1	9	9
2	0	9
2	1	0
2	2	2
2	3	6
2	4	4
2	5	7
2	6	5
2	7	1
2	8	3
2	9	8

Використання цієї таблиці має бути дуже обмеженим, оскільки за допомогою неї дані можуть бути викриті. Отже, таблиця повинна бути доступна в основному лише адміністратору бази даних, який створює її.

Щоб почати фактично маскувати дані за допомогою таблиці відображення, слід створити процедуру, яка бере вибраний стовпець таблиці та замінює кожне значення значенням із таблиці відображення. Процедура повинна використовувати як вхідне значення атрибут, який потрібно замаскувати, а алгоритм повинен

повертатись до початку виконання після маскування першого вхідного значення. Цей метод потребує підготовки до роботи вручну та аналізу даних, щоб переконатися, що маскування працює для кожного замаскованого значення

2.3 Статичний та динамічний способи маскування даних

Маскування даних може відбуватися двома основними способами:

1. Статичне маскування даних: це маскування вхідних даних, що постійно перебувають у фізичному рівні зберігання. Маскування починається з зчитування даних із виробничих наборів, а потім застосовується ряд правил перетворення (оборотні або нереверсивні методи) для отримання реалістичних де-ідентифікованих даних, а потім збереження їх у таблицях даних призначення.

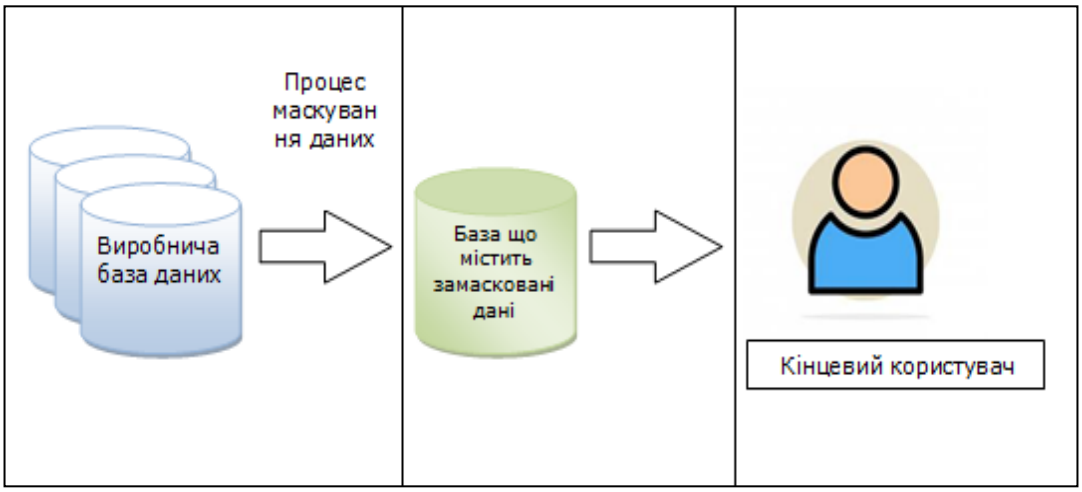

Рисунок 2.5 - Статичне маскування даних

Перваги:

- Конфіденційні дані постійно затьмарюються завдяки застосуванню методів маскування даних під час процесу вилучення, перетворення та завантаження.
- Відсутність необхідності в будь-якому зайвому процесі при отриманні даних на етапі використання.
- Безпечно ділитися такими даними з внутрішніми та зовнішніми зацікавленими сторонами.

Недоліки:

- Процес маскування може зайняти багато часу, залежно від розміру витягнутого набору даних.
- Такий спосіб не може бути легко використаний для резервного копіювання виробничих наборів даних, оскільки йому потрібно застосувати алгоритм зняття маскування до всього набору даних, що маскується, і це може зайняти час.

2. Динамічне маскування даних - це анонімізація конфіденційних даних на рівні презентації під час запиту користувача, залишаючи вихідні дані в стані спокою без змін. Основне використання динамічного маскування даних полягає у застосуванні функцій безпеки на основі ролей до кожного кінцевого користувача в контексті лише для читання для цілей звітування. Для реалізації цього способу використовують рівень проксі для модифікації запиту SQL та повернення модифікованого результату запиту шляхом застосування різних типів традиційних методів маскування.

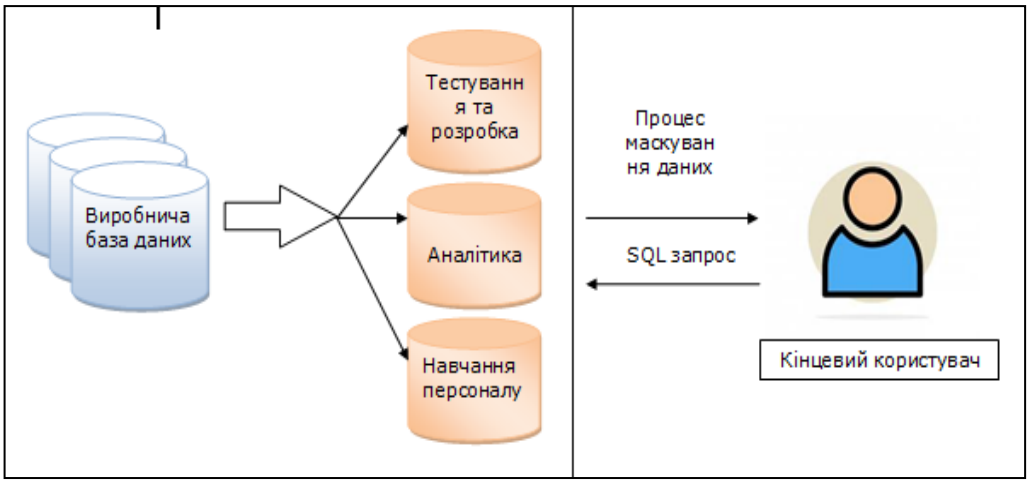

Рисунок 2.6 - Динамічне маскування даних

Переваги:

- Додає додатковий рівень конфіденційності для захисту виділених чутливих даних.
- Керує захистом даних на рівні презентації в підході лише для читання.
- Не потрібно застосовувати прийоми маскування заздалегідь, щоб замаскувати потрібний набір даних

Недоліки:

- Накладні витрати на продуктивність, пов’язані з перевіркою всього трафіку запитів.
- Потребує детального картографування користувачів, наборів даних, полів даних, а також правил, як підтримувати цю конфігурацію матриці.
- Проксі-сервер бази даних є єдиною точкою відмови, і його може обійти зловмисник.

2.4 Цілісність даних при використанні технології маскування

Кожен із описаних на сьогодні методів ефективний для маскування даних для забезпечення конфіденційності. Однак у великих баз даних, що при цьому мають зв'зки між собою виникає додаткове ускладнення. Зокрема, вам потрібна можливість розповсюдження замаскованого елемента даних до всіх пов’язаних таблиць бази, щоб підтримувати цілісність.

Якщо маскований елемент даних (наприклад, телефонний номер) є первинним або вторинним (зовнішнім) ключем у відношенні таблиці бази даних, то це нещодавно замасковане значення даних має бути розповсюджене на всі пов'язані таблиці бази даних. Поширення ключів забезпечує цілісність перетворених даних. Без розповсюдження ключів взаємозв'язок між батьківською та дочірньою таблицями буде розірваний, що призведе до неточності даних тесту. Отже, тестування додатків дасть ненадійні результати. Визначене значення може бути дійсним іменем стовпця, літеральним рядком, виразом чи чимось іншим. Наприклад, припустимо просту модель даних, що складається з двох пов’язаних таблиць: Клієнти та Замовлення, показані на рисунку 2.7.

Таблиця "Клієнти" є основню та батьківською для таблиці "Замовлення", а її стовпець первинного ключа - "Ідентифікатор_користувача" - це 5-значне числове значення.

Рисунок 2.7 - Передача маскованих даних між зв'язаними таблицями

Якщо ідентифікатор замовника маскується за допомогою методу послідовного маскування, масковані значення повинні бути розповсюджені по всіх пов'язаних таблицях для забезпечення цілісності посилань даних. У цьому прикладі стовпці ID, Ім'я та Вулиця маскуються. Зверніть увагу, що ім'я "Томі Версетті" було замасковане як "Томас Шелбі". Назва вулиці також замаскована. Зокрема, метод послідовного маскування був використаний для перетворення вихідного ідентифікатора ID для Томі Версетті з 27645 на 10002. Коли це замасковане значення ID поширюється з таблиці «Клієнти» (батьківської) на всі пов'язані таблиці, ключовий зв'язок між таблицями «Клієнти» та «Замовлення» у тестовій базі даних залишається цілим. Без можливості розповсюдження маскованих значень цілісність даних буде порушена.

2.5 Проблеми при використанні технології маскування даних

У багатьох організаціях часто важливо створювати копії виробничих баз даних для невиробничого використання, наприклад розробки та тестування додатків, особистого навчання, дослідження та формування аналітики. Окрім загроз порушення конфіденційності та незаконного отримання інформації зловмисниками

за межами підприємства, таке багаторазове копіювання даних збільшує також і потенційний ризик розкриття інформації працівниками, яким не надано дозволу до цих даних.

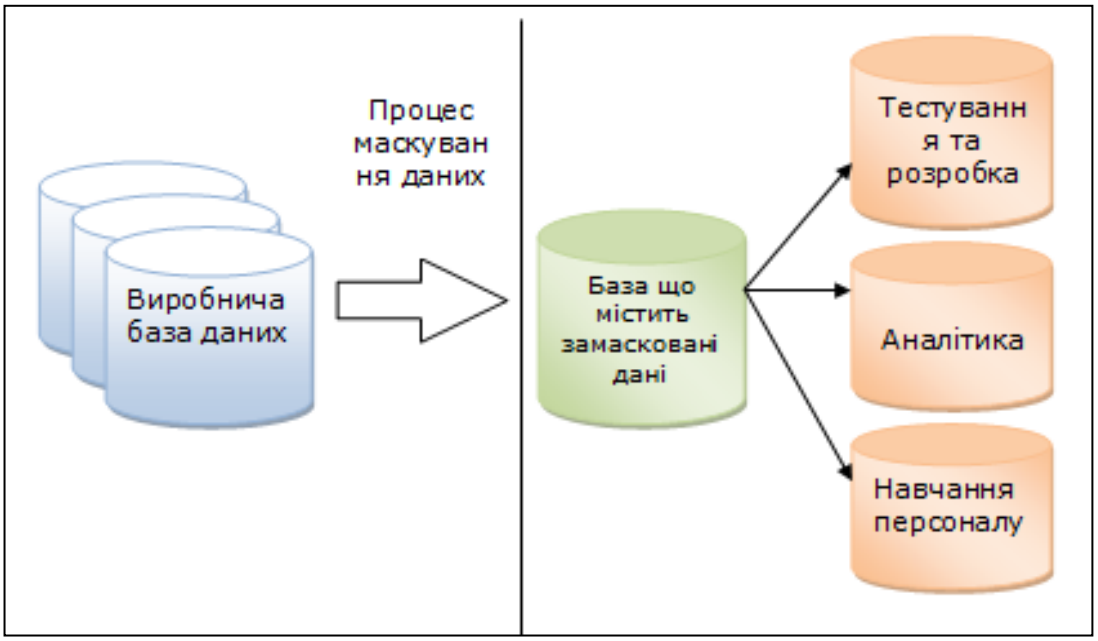

Рисунок 2.8 - Створення виробничих копій баз даних

Вирішення питань конфіденційності даних та їх корисності для загального сховища даних, а також служби аналізу на платформі підпримства є основною проблемою, яка включає труднощі маскування конфіденційних даних від внутрішніх та зовнішніх користувачів, оскільки загальне сховище бере інформацію з різних операційних баз даних і може містити конфіденційні дані у відкритому вигляді.

Традиційний метод маскування даних, що зазвичай використовують, він незворотний, оскільки маскує (де-ідентифікує) в один бік і не підтримує корисність даних для аналізу, звітності та дослідницьких цілей. Кілька постачальників ПЗ та дослідників запропонували та впровадили рішення для маскування даних для тестового середовища, однак більшість із них є незалежними додатками сторонніх розробників.

Традиційні методи маскування даних, сторонні рішення не можуть задовольнити потребу в інтегрованому середовищі аналітики для доставки точної

інформації клієнтам, зберігаючи конфіденційність важливих даних у загальному сховищі даних та на всій платформі. Таким чином, малоймовірно, що корпоративна система, яка використовується на підприємстві, виграє від наявності зовнішнього рішення, яке потрібно доопрацювати та налаштувати відповідно до платформи, що використовується.

Використання класичних методів маскування даних корисно для деідентифікації конфіденційних даних та збереження їх конфіденційності але серйозною проблемою при цьому є втрата якості аналізу даних.

Також, як правило, методи маскування даних розробляються вручну та реалізуються незалежно, спеціально та суб'єктивно для кожного додатка. Такий підхід до спеціального маскування даних вимагає трудомісткого та тривалого у часі випробування та впровадження суперечливих алгоритмів або використання різних методів маскування у декількох програмах взаємодії, що не є логічним та продуктивним рішенням для будь-якої організації.

Отже, для подолання цієї проблеми, найефективнішим рішенням для підприємства було б створення власної унікальної структури маскування даних, що буде інтегрована та актуальна для конкретної інформаційної системи. Ця структура маскування даних повинна включати в собі найефективніші методи, що застосовуються для модифікації даних, та містити повний набір виробничих даних, що існують на підприємстві.

Висновки за розділом 2

Маскування - це технологія, яка напряму залежить від потреб її користувача, згідно цього, спосіб та метод модифікації інформації буде визначатися, основуючись на інтересах користувача, та можливості відновлення оригінальних даних. Маскування надає можливість вибору із широкого набору методів та засобів для забезпечення безпеки, корисного та необхідного особисто вам. Також ця технологія є абсолютно лояльною до обчислювальних можливостей будь-якої платформи,

надаючи можливість вибору способу маскування та доставки модифікованих даних користувачу.

Зміна вхідних оригінальних даних може реалізовуватися шляхом заміни інформації на підроблену, із заздалегідь підготовленого списку, також можлива проста перетасовка значень відповідного поля у базі, або обнуління, тобто заміна певної кількості символів оригінальної інформації. Іншим сценарієм у модифікації вхідних даних числового типу або дати може бути їх зміна на певний прийнятний відсоток або діапазон.

Технологія маскування інформації підтримує цілісність даних, до яких була застосована модифікація, це означає, що, якщо існує зв’язок між декількома таблицями сховища, то дані, які передаються або використовуються у них, будуть залишатися у своєму зміненому вигляді.

РОЗДІЛ 3
СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МАСКУВАННЯ ДАНИХ

3.1 Існуючі програмні рішення для маскування даних

Перед початком роботи над створенням власного додатку для маскування конфіденційних даних, підготовчим етапом є ознайомлення з готовими програмними рішеннями та підходами до захисту конфіденційних даних, для їх подальшого дослідження, аналізу та виділення необхідної інформації, яка буде використана при створенні власного додатку.

На сьогоднішній день існує велика кількість різноманітного програмного забезпечення, а також окремих додатків для маскування та модифікаціїї конфіденційних даних, які вимагають лише їх налаштування для подальшого використання та надання належного рівня безпеки. Одними з найвідоміших виробників, що надають послуги для маскування даних є:

- Mentis
- Oracle
- Informatica
- Talend
- Salesforce
- Delphix

Вибір продукту, що буде використовуватися для модифікації даних на підприємстві, залежить в першу чергу від розміру інформаційних активів організації та кількості співробітників. Невеликі компанії не мають потреби в створенні власного програмного забезпечення, тому частіше за все, для них прийнятним буде обрати готовий варіант з його подальшим налаштуванням. У свою ж чергу, для організацій, що мають у своєму розпорядженні велики інформаційні активи, використання унікального, створенного самостійно ПЗ, буде більш логічним

варіантом, тому що в такому випадку вони не будуть залежати від виробника ПЗ та матимуть повний контроль над процедурою маскування даних.

3.1.1 Рішення на основі надбудови Excel:

Для малих та середніх наборів даних, що містять від 20 до 10000 записів, можна використати існуюче рішення для маскування від компанції Microsoft – функцію модифікації даних, що вбудована в Microsoft Excel. Параметр надбудови відображається в електронній таблиці наступним чином:

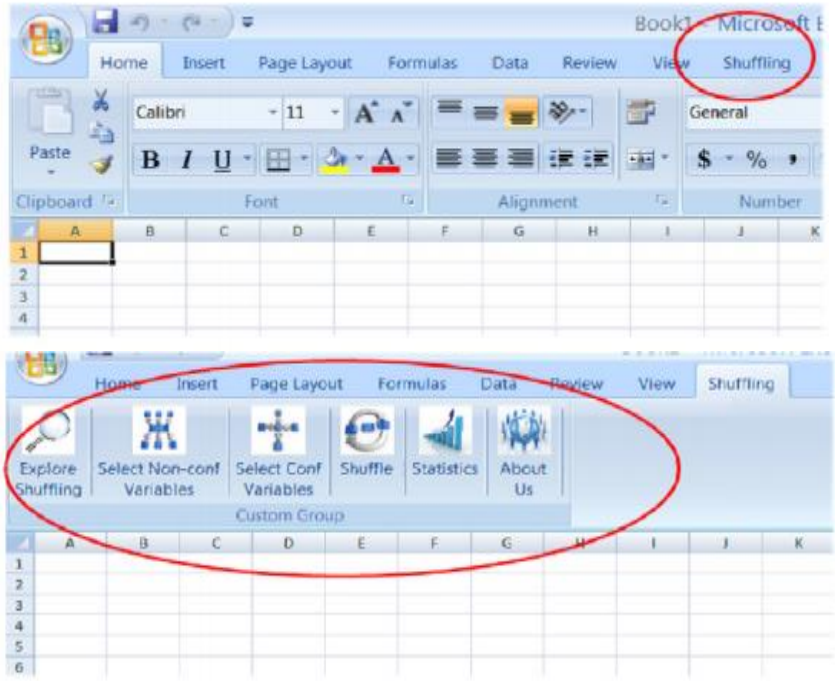

Рисунок 3.1 - Модифікація даних в Excel

Як показано на рисунку, користувач може вводити дані в електронну таблицю, вибирати відповідні стовпці не конфіденційних та конфіденційних даних та виконувати перетасовку. Вкладка статистичних даних дасть основну статистичну інформацію як для вхідних, так і для замаскованих (перетасованих) даних, щоб користувач міг оцінити ефективність модифікації.

Звичайно, важливою перевагою данної функції в Excel є те, що користувач може проводити додатковий аналіз за допомогою програмного забезпечення власноруч, таким чином перевіряючи правильність модифікації.

3.1.2 Веб-додатки

Організації, які використовують набори даних середнього розміру (до 100 000 записів) і яким потрібні лише масковані дані, при цьому вони не хочуть встановлювати додаткове програмне забезпечення, можуть скористатися готовими рішенням яке реалізується за допомогою веб-додатків. На знімках екрана нижче показано веб-інтерфейс. Як видно з рисунка інтерфейс дозволяє завантажувати набір даних, вибирати відповідні змінні для маскування та виконувати перемішування за допомогою однієї кнопки.

Однією з особливостей веб-рішення є те, що воно дозволяє користувачеві зберігати історію зміни файлу, попередніх спроб маскування, так що багаторазові запуски (з різними змінними, що можливо маскуються кожного разу) можуть бути легко відстежені.

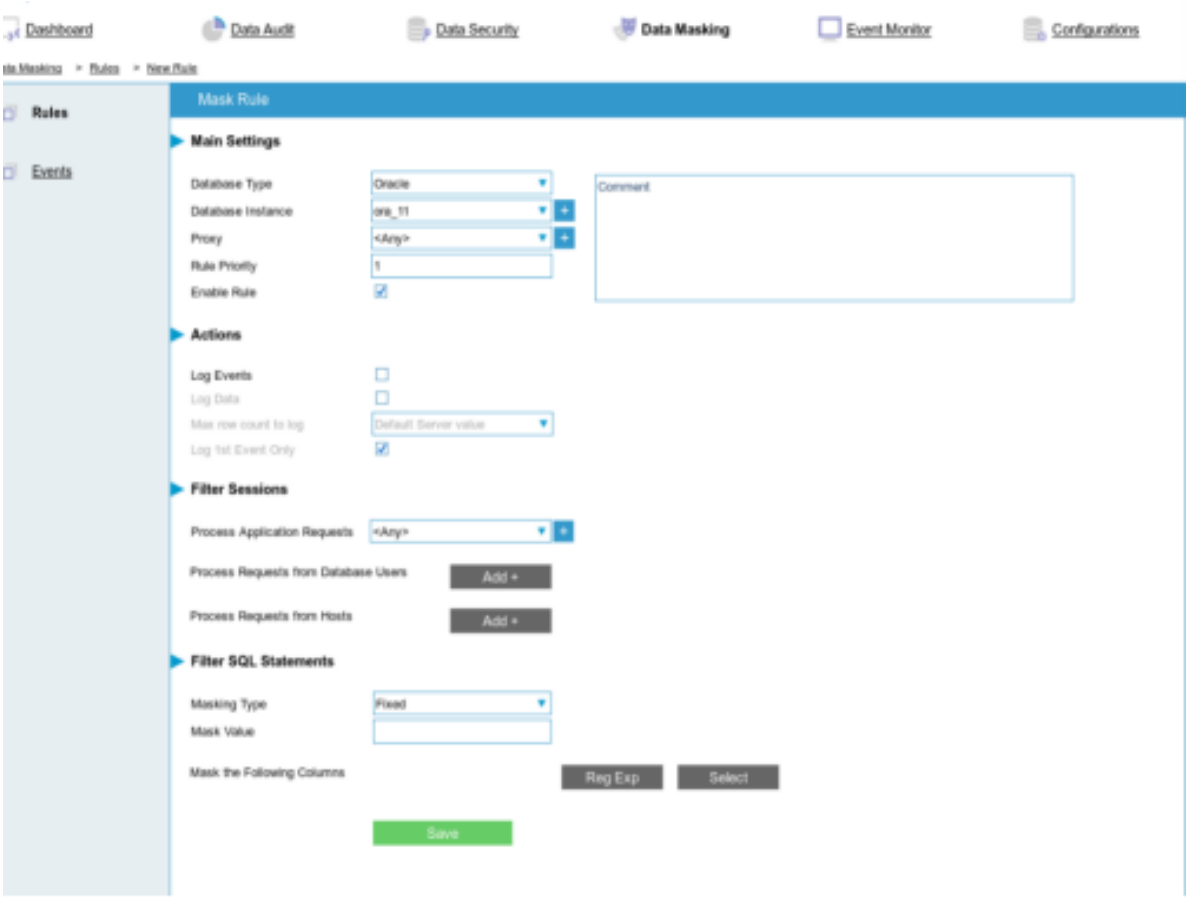

Рисунок 3.2 - Веб-рішення для маскування даних

3.1.3 Стороннє рішення - додаток

Третій вид готового рішення, яке буде продемонстровано – це сторонній додаток на основі будь якої мови програмування. Такий додаток подібний до веб-рішення, за винятком того, що його можна встановити на сайті клієнтів.

Користувачам, які бажають мати повний контроль над даними та програмним забезпеченням цей варіант буде корисним. Цей підхід також дозволяє використовувати ряд вбудованих функцій, які можуть забезпечити не тількі маскування, а ще і аналіз даних.

Додаток може бути масштабований для різних наборів даних. Наведений нижче рисунок показує головний екран програми. Такий підхід дозволяє вибирати які набори даних потрібно проігнорувати, які дані не є конфіденційними, а також інформацію, яку потрібно замаскувати. Як і у веб-застосунку, натискання однієї кнопки виконує модифікацію даних і зберігає як базу з маскованими даними, так і вхідну.

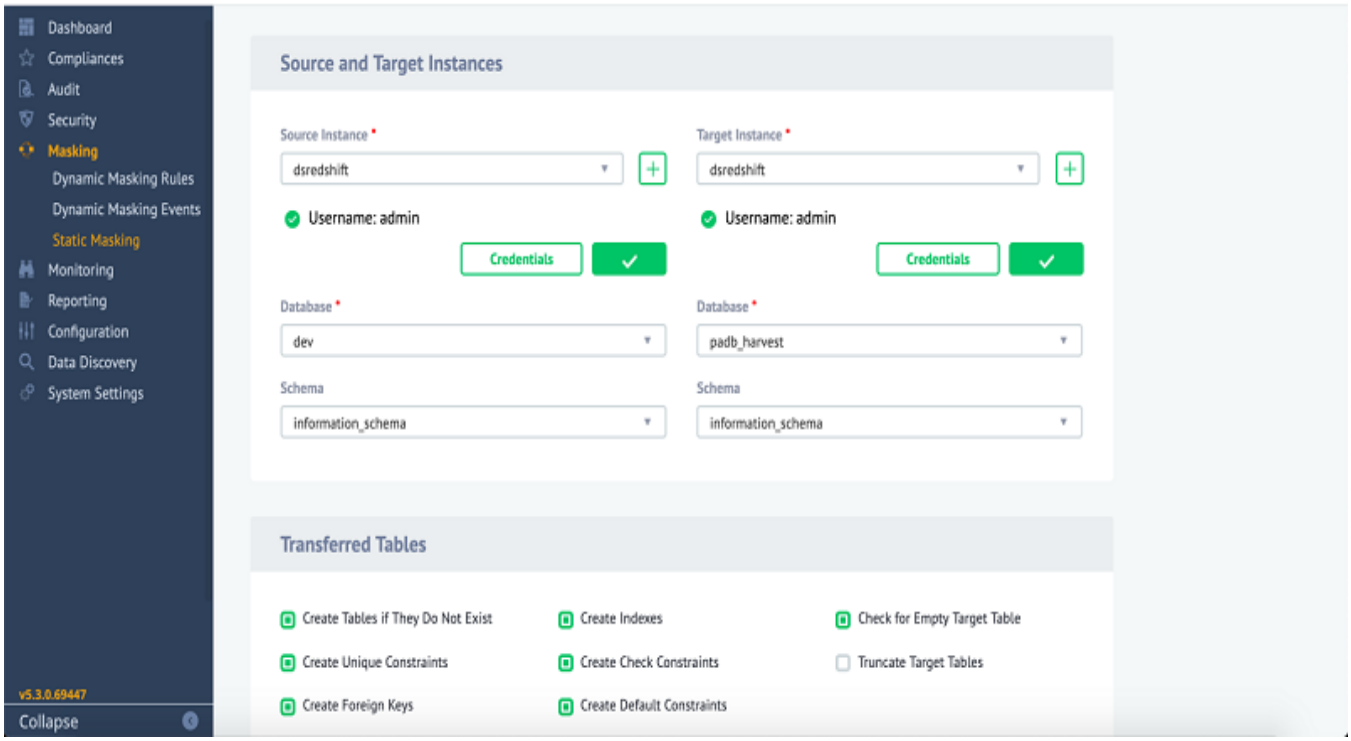

Рисунок 3.3 - Сторонній додаток для маскування даних

3.1.4 Недоліки використання готових додатків

Ознайомившись із вищеописаними готовими підходами та продуктами для маскування даних, проаналізувавши їх особливості, необхідно також дослідити і недоліки, для того, щоб в подальшому уникнути їх при створенні власного рішення.

Однією з проблем є непрозорість сценарію маскування. Компанії, що надають послуги із забезпечення конфіденційності інформації не дають загального доступу до технічної частини їх продукту, це означає, що код виконання процессу маскування даних є скритим. Замовник додатку не може побачити та оцінити алгоритм та процедуру маскування, він отримує вже вихідну базу, що містить модифіковані дані. Адміністратор бази органцізаціїї-замовника лише виконує певне налаштування додатку під платформу підприємства, тому йому надзвичайно важко запропонувати будь-які рекомендаціїї, щодо реалізації процессу маскування компаніїї-постачальнику.

Іншою проблемою є ремонтопридатність. Постачальник послуг може проводити різноманітні оновлення додатку, після чого існує вірогідність порушення цілісності замаскованих конфіденційних даних у базі замовника, що збільшує можливість витоку таких даних. Для збереження надійного рівня захисту чутливих даних в такому випадку, необіхдно використовувати резервні копії баз, що містять масковані елементи та очікувати технічної підтримки від постачальника.

Як вже зазначалося у попередньому розділі данної роботи, сторонні рішення не зможуть повністью задовільнити потребу доставки точної модифікованої інформації клієнтам, зберігаючи конфіденційність чутливих даних у загальному сховищі даних та на всій платформі. Таким чином, малоймовірно, що платформа, яка використовується на підприємстві, виграє від наявності зовнішнього рішення, яке потрібно доопрацювати та налаштувати. Згідно цього, дійдемо до висновку, що власне рішення для маскування буде більш надійним та у майбутньому дасть більший профіт, ніж використання запропонованого іменитими розробниками додатку.

3.2 Створення програмного рішення для маскування даних

3.2.1 Оранізаційні питання процессу розробки

Метою дипломної роботи є розробка програмного рішення для маскування даних. Для реалізації цього завдання необхідно розглянути певні організаційні етапи перед створенням додатку:

1. Вибір можливих прийомів та підходів маскування даних. На цьому етапі потрібно визначити, які способи маскування даних можуть використовуватись, та виділити їх, відкинувши способи і методи модифікаціїї інформації, які точно не будуь використовуватися та не є актуальними для нашого додатку.

2. Відповідність та коректність вибору. На цьому етапі з переліку виділених способів та методів маскування даних потрібно обрати той, що безпосередньо буде використовуватися у нашому программному рішенні. Вибір належного способу та методу маскування даних з переліку обраних, може бути виділений шляхом тестових випробувань та перевірки працездатності певного алгоритму.

3. Аналіз надійність обраного методу. Цей етап передбачає аналіз попереднього кроку, тобто аналіз правильності роботи обраного підходу до маскування, перевірка цілісності даних під час процессу маскування, та їх користності для майбутнього аналізу, на основі готових рішень. Також вирішується питання безпеки, де визначається, чи потрібні масковані дані для створення звітів, та чи є важливою якість замаскованих даних.

4. Безпека та конфіденційність. Цей етап розглядає питання міцності обраного способу маскування. Проводиться аналіз загроз та перевіряється можливість розкриття маскованих даних.

5. Перевірка продуктивності. Цей етап передбачає перевірку обраного підходу до маскування на швидкодію. Перевіряється швидкість виконання алгоритму маскування/демаскування. Також проводиться дослідження, де визначається, як ефективніше проводити модифікацію даних: на рівні запитів, або маскуванням із основної бази.

6. Аналіз сховища даних. Останній етап включає в себе порівняння змінених даних, дослідження зміни їх розміру та процессу повернення до основної бази. Оглянувши існуючі рішення отримуємо інформацію про оригінальний файл бази та кінцевий, після застосування маскування. Після чого виконуємо зіставлення розмірів баз, аналізуємо зміна розміру та визначаємо чи є допустимою така зміна розміру для нашого сховища.

Розглянувши основні організаційні питання перед процессом розробки программного додатку, провівши аналіз та порівняння, мною було обрано традиційний метод маскування даних, так як цей метод надійно модифікує інформацію, створюючи підроблені дані, викриття яких майже неможливе.

3.2.2 План розробки додатку та технічне завдання

Визначившись із способом маскування даних, необхідно перейти до питання створення плану розробки додатку та створення технічного завдання.

План розробки додатку для маскування даних буде складатися з чотирьох-етапного комплексу, до якого входить:

1. Процесс виділення інформаціїї,
2. Оцінка алгоритму маскування,
3. Захист
4. Тестування.

Таблиця 3.1 - Етапи розробки додатку

№	Етап	Опис
1	Виділення інформаціїї.	На цьому етапі передбачається виділення конфіденційних даних із загального сховища, їх каталогізація, та визначення чутливої інформаціїї, що потребує захисту. Мета цього етапу – скласти вичерпний перелік конфіденційних даних, виявити пов'язані із ними зв'язки в таблицях.
2	Оцінка алгоритму маскування	Цей етап передбачає поєднання організаційних питань, з вибору підходу до маскування, з технічними. Аналітики безпеки та адміністратори бази визначають оптимальний метод маскування для заміни конфіденційних даних. Розробники можуть використовувати існуючий метод, скористатися вбудованими функціями платформи на підприємстві, або розробити власну процедуру маскуванн даних

3	Захист	На цьому етапі використовується обраний підхід до маскування даних. Виконується перевірка коректності роботи алгоритму. Обраний метод застосовується до тестових даних, визначається швидкість та надійність роботи рішення. Після проведення вищеописаних дій, алгоритм передається адміністратору бази для наступного кроку.
4	Тестування	На завершальному етапі, алгоритм передається адміністратору бази, який додає програмне рішення до загального сховища та перевіряє його роботу, проводить тестувальне маскування даних, перевіряє чи можна отримати оригінальні дані, далі робить висновок, після якого програмне рішення або повертається на етап створення та перевірки алгоритму роботи, або залишається у сховищі.

Технічне завдання визначається в меті цієї дипломної роботи, а саме – створення програмного рішення та побудова коректно функціонуючого додатка для маскування даних із підтримкою таблиці з вхідними даними, та виводом у таблицю вже замаскованих даних.

3.2.3 Структура додатку

Обраною мовою програмування для створення додатку є Python. Кінцевий проект в обов'язковому порядку повинен включати в себе файл вхідних даних, а також основний файл, в якому буде описаний алгоритм процедури маскування інформаціїї.

У попередньому пункті даного розділу було визначено, що буде використовуватись традиційний спосіб маскування, а саме метод заміни вхідних даних.

Кінцева мета – отримувати модифікований файл вхідних даних, який після виконання буде містити вже замасковану за обраним підходом інформацію. При розробці додатку буде розглянута проблема підключення основного файлу конфігурації до існуючої бази даних. На додаток до цього, буде реалізовано процедуру, що буде створювати випадково-сгенеровані дані для обфускації файлу або таблиці ними.

Вхідні дані, що створенні та подані у вигляді таблиці зі змінними, знаходяться у файлі demographics.csv. Нижче наведено таблицю подання даних та заповнені значення.

Таблиця 3.2 - Вхідні дані файлу demographics.csv

Id	bithDate	Gender	First Name	Last Name	Address	City	Post Code	Email	Phone
1	01-01-2000	Male	Kirill	Zolotaryov	Perekopskaya 193	Kherson	73000	kirl@gmail.com	066-999-55-66
2	31-05-1999	Male	Oleg	Shalatonov	Vasilkivska 17	Kyiv	52000	ole@gmail.com	050-123-45-67

Процедура маскування даних описана в файлі DataMasker.py. Для подальшої модифікації даних було імпортовано перелік необіхдних бібліотек та модулів:

- Time. Модуль time зі стандартної бібліотеки мови програмування Python містить масу корисних методів для роботи з часом. З його допомогою можна

отримувати інформацію про поточну дату і час з точністю до мілісекунд, виводити ці відомості в необхідному форматі, а також керувати ходом виконання програми, додаючи затримки по таймеру.

- Datetime. Модуль datetime з стандартної бібліотеки Python — це збірка з самих різних класів і методів для комфортної роботи з часом і датами: отримування поточної системи дати та часу; обчислення різниці між датами та інші операції; операції для порівняння дати/часу; форматування інформації про дату/час.
- Faker. Faker - це бібліотека Python, з відкритим кодом, що дозволяє вам створювати власний набір даних. Тобто ви можете генерувати випадкові дані із випадковими атрибутами, такими як ім'я, вік, місцезнаходження тощо. Підроблені дані часто використовують для тестування або заповнення баз даних деякими фіктивними даними.
- Dateutil. Модуль dateutil надає потужні розришення до стандартного модуля бібліотеки Python datetime. він потрібен для того, щоб конвертувати строку у формат дати та часу.

Для реалізаціїї технології маскування, при розробці додатку було реалізовано набір функцій, основні з яких будут представлені далі.

1. Функція генерації підробної адреси. За допомогою бібліотеки Faker генеруємо значення адреси, після чого робимо перевірку отриманих значень та проводимо поділ на окремі частини, для їх подальшого внесення у таблицю.

```
def generateAddress() :

    address = {}
    fullAddress = fake.address()
    addressParts = fullAddress.splitlines()
    address["address"] = addressParts[0]

    if "," in addressParts[1] :
        cityStateZip = addressParts[1].split(",")
        stateZip = cityStateZip[1].split()
        address["city"] = cityStateZip[0]
```

Рисунок 3.4 - Лістинг процедури генерації адреси

```
            address[“state”] = stateZip[0]
            address["postalCode"] = stateZip [1]
    else :
            cityStateZip = addressParts[1].split()
            address["city"] = cityStateZip[0]
            address["state"] = cityStateZip[1]
            address["postalCode"] = cityStateZip[2]
    return address
```

Рисунок 3.5 - Лістинг процедури генерації адреси

2. Функція створення фальшивого ім’я. Дана функція створює словник для його подальшого заповлення згенерованими даними. Після цього, спираючись на значення поля «гендер» із вхідної таблиці, перевіряється стать, та відповідно до неї створюються фальшиві дані імені та прізвища.

```
def generateName(gender = None) :

    name = {}

    if gender is None:
        name[‘firstName’] = fake.first_name()
    else :
        if gender == ‘Male’ :
            name[‘firstName’] = fake.first_name_male()
        else:
            name[‘firstName’] = fake.first_name_female()

    name[‘lastName’] = fake.last_name()
    return name
```

Рисунок 3.6 - Перевірка статі та генерація фальшивого імені

3. Функція генерації підробленої дати народження. Значення дати отримується шляхом додавання до початкової дати, згенерованого за допомогою функції timedelta проміжку часу.

```
def add_days(startdate, days):
        return startdate + datetime.timedelta(days=days)
```

Рисунок 3.7 - Створення фальшивого значення дати народження

4. Функції створення фальшивого електронного ящика, номеру телефону та універсального унікального ідентифікатору. Дані функції виконують простий виклик існуючих у бібліотеці Faker методів для створення підробленої інформації.

```
def generateContact() :
   contact = {}
   contact["email"] = fake.email()
   contact["phone"] = fake.phone_number()
   return contact

def generate_uuid() :
        fake = Factory.create()
        uuid = fake.uuid4()
        uuid = uuid.replace("-", "").upper()
        return uuid
```

Рисунок 3.8 - Створення підробного значення поштового ящика

Для того, щоб файл процедури маскування даних отримував потрібну йому інформацію, необхідно створити файл DemographicsMasking.py, в якому буде реалізована функція зчитування та запису даних, відповідно до стовпців таблиці, що міститься у файлі вхідних даних.

1. Функція створення повного набору підробних значень для полів вхідної таблиці. За допомогою цієї функції, стовпці таблиці заповнюються сгенерованими фальшиви даними використовуючи методи, що описані у файлі DataMasker.py.

```
def generateDemographics(input) :

        member = {}
        member["ssn"] = DataMasker.generateSSN()

        bDate = parse(input["birthDate"])
        member['birthDate'] = DataMasker.add_days(bDate, 10).strftime('%m-%d-
%Y')

        member['gender'] = input['gender']

        name = DataMasker.generateName(member['gender'])
        member['firstName'] = name['firstName']
        member['lastName'] = name['lastName']

        address = DataMasker.generateAddress()
        member['address'] = address["address"]
        member['city'] =  address["city"]
        member['postalCode'] = address["postalCode"]

        contact = DataMasker.generateContact()
        member["phone"] = contact["phone"]
        member["email"] = contact["email"]

        return member
```

Рисунок 3.9 - Лістинг функції заповнення полів підробними даними

2. Основна функція main() потрібна для того, щоб після початку виконання програми відкривався файл вхідних даних, вони зчитувалися, додавалися у список і за допомогою функції, що описана вище, заповнювалися сгенерованими

замаскованими даними, після чого записувалися у файл, відповідно до стовпців таблиці вхідного файлу.

```
def main():

    start = time.time()
    list = []

    with open('data/demographics.csv') as inFile:
        reader = csv.DictReader(inFile)
        for row in reader:
                result = generateDemographics(row)
                list.append(result)
    inFile.close()

    with open('demographicsMasked.csv', 'w') as outFile:
        fieldnames = ['ssn','birthDate','gender','firstName','lastName',
                'address','city','state','postalCode','email','phone']
        writer = csv.DictWriter(outFile, fieldnames=fieldnames)
        writer.writeheader()
        for row in list:
                writer.writerow(row)
    outFile.close()

    end = time.time()
    elapsed = end - start
```

Рисунок 3.10 - Лістинг основної конфігураційної функції main()

Якщо ви хочете підключитися до власної бази даних, відкрийте створений файл config.json, щоб змінити деталі підключення до бази. Для створення зв’язку з вашою базою, необхідно змінити значення в об'єкті "db_connection_details". Ініціалізація бази реалізоана у Додатку А.

```
“proxy_port”:2000,
“proxy_source_ip”: “127.0.0.1”,
"db_connection_details": {
      "port": 15432,
      "ip": "127.0.0.1",
      "user":"user",
      "password":"password",
      "database":"test_data"
   }
```

Рисунок 3.11 - Налаштування підключення до бази даних

Підроблені дані, які будуть використання для реалізації технології маскування, можуть бути взяти, як з переліку існуючих значень бібліотеки Faker, або за допомогою генератора, який було створенно на додаток до основного завдання. Інформація генерується на основі методів Faker, після чого йде заповнення нового файлу сгенерованими даними. Дана функція є досить корисною, по перше, як інструмент виробничої перевірки і надання значень для заміни, а , по-друге, з її допомогою можна виконувати обфускація основної таблиці.

3.2.4 Демонстрація роботи программного рішення

Робота додатку починається з заповнення вхідного файлу, даними користувача, згідно полів таблиці.

```
 GNU nano 5.4                                                    demographics.csv
id,birthDate,gender,firstName,lastName,job,address,city,postalCode,email,phone
1,01-01-2000,Male,Kirill,Zolotarev,programmer,Perekopskaya 153,Kherson,73000,kirl@gmail.com,066-999-55-66
2,31-05-1999,male,Oleg,Shalatonov,cybersecurity,Vasilkivska 17,Kyiv,52000,ole@gmail.com,050-123-45-67
```

Рисунок 3.12 - Заповнення таблиці вхідного файлу даними

Наступним етапом роботи додатку є запуск файлу процедури маскування. Для цього запускаємо файл DemographicsMasking.py та чекаємо завершення виконання.

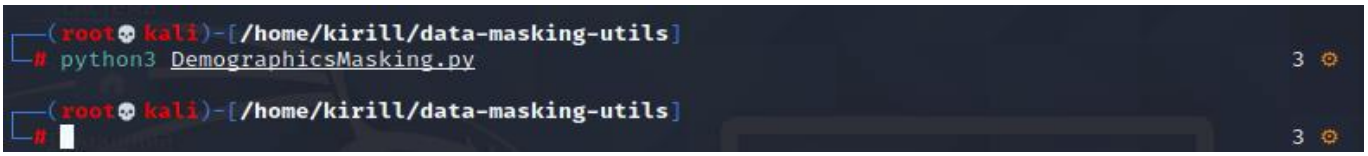

Рисунок 3.13 - Запуск процедури маскування

Масковані дані поміщаються у новий файл, згідно полів таблиці вхідного файлу.

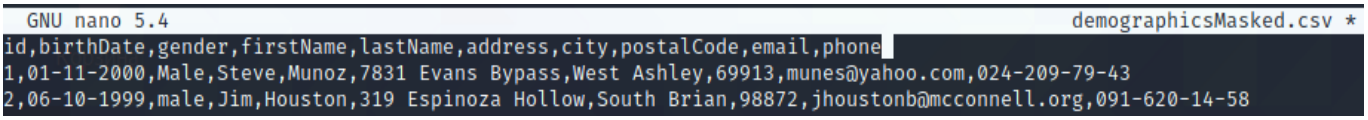

Рисунок 3.14 - Файл маскованих даних

Запуск генератора випадкових значень використовується для створення набору даних, що буде використаний для обфускації кінцевого файлу.

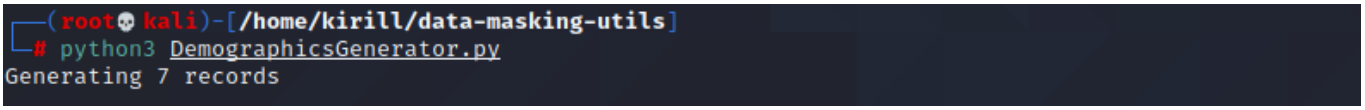

Рис.3.15 - Запуск генератора даних

Створенні генератором дані, поміщуються у новий файл, згідно полів вхідної таблиці. Дані створюються випадковим чином, не спираючись ні на один атрибут.

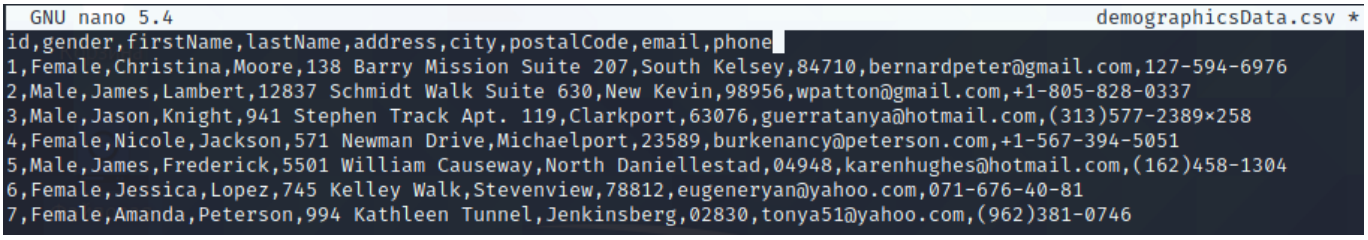

Рисунок 3.16 - Набір створених генератором даних

Сгенеровані дані додаються до передньо замаскованої конкретної інформаціїї, перемішуються та виводяться у кінцевий файл.

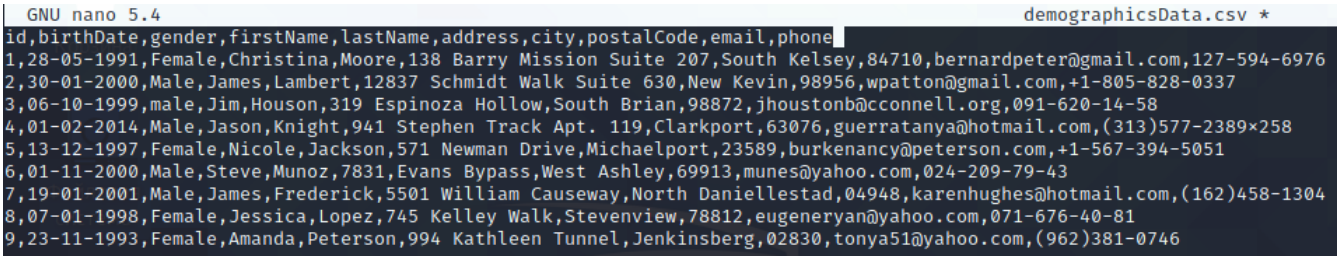

Рис 3.17 - Вивід даних у кінцевий файл

Висновки за розділом 3

У результаті дослідження було прийнято рішення створити власне програмне забезпечення маскування даних. Основною причиною для створення додатку стала непрозорість алгоритму модифікації даних, складність налаштування, та відсутність належного рівня універсальності при застосуванні на різних платформах організацій. Було виявлено та опрацьовано недоліки у розглянутих практичних підходах до процедури маскування, створено план розробки власного додатку, та обрано спосіб і метод модифікації оригінальних даних. Мовою програмування для створення програмного рішення було обрано Python3.

На етапі створення проекту, було визначено набір основних файлів, а також описано функції, методи та бібліотеки, що використовувалися при розробці додатку, розглянувши їх можливості та причини використання.

Як результат, було отримано кінцевий проект, що реалізує технологію маскування даних, отримуючи інформацію з вхідного файлу, що містить таблицю оригінальних значень. Також було створено скрипт-генератор випадкових фальшивих наборів даних, що може бути використаний для обфускації основної таблиці, або як вибірка даних для заміни оригінальних, при проходженні процедури маскування.

ВИСНОВКИ

В процесі написання дипломної роботи було досліджено та проаналізовано технологію маскування даних. Це один із надійних підходів до забезпечення конфіденційності інформації, який робить викриття та віднайдення оригінальних даних майже неможливим. Під час процедури маскування створюється фальшивий набір значень, які є дуже схожими на реальні.

Під час аналізу існуючих способів та методів маскування даних, було виділено їх особливості, алгоритми реалізації, а також визначено слабкі моменти та недоліки.

Важливим питанням при вивчені даної технології стала деяка схожість на процедуру шифрування, тому було проведено їх порівняння, а також аналіз алгоритмів зміни вхідної інформації. Програмні застосунки для маскування даних, також були описані та детально розглянуті під час виконання даної роботи. Визначивши усі проблемні місця готових додатків, було прийнято рішення створення індивідуального програмного застосунку для реалізації технології маскування даних. Причинами створення додатку стали складність налаштування та відсутність прозорості основного алгоритму модифікації.

Підводячи підсумки, слід зазначити, що у результаті виконання дипломної роботи було досягнуто основну мету, а саме - створено програмне забезпечення маскування даних, а також реалізовано, потрібні для її досягнення, додаткові завдання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Aimultiple [Електронний ресурс]. Режим доступу: https://research.aimultiple.com/datamasking/#:~:text=It%20is%20the%20process%20of,shares%20data%20with%20third%20parties

2. “Use data masking to ensecure secure – and compliant – software delivery”, TechnoBeacon [Електронний ресурс]. Режим доступу: https://techbeacon.com/security/use-data-masking-ensure-secure-compliant-software-delivery

3. Muralidhar, K. and R. Sarathy, "Data Shuffling- A New Masking Approach for Numerical Data", (2006) , ст. 58-670.

4. Manjunath T.N, Ravindra S Hegadi, Ravikumar G K."Analysis of Data Quality Aspects in Datawarehouse Systems", International Journal of Computer Science and Information Technologies, , (2010), ст. 477-485

5. Sarathy, R. and K. Muralidhar, "The Security of Confidential Numerical Data in Databases", (2002), 389-403.

6. “Masking data in practice”, Red-gate [Електронний ресурс]. Режим доступу: https://www.red-gate.com/hub/product-learning/data-masker/masking-data-practice

7. “Overview of Data masking methods”, Smartbridge [Електронний ресурс]. Режим доступу: https://smartbridge.com/overview-data-masking-methods/

8. «Маскировка данных», Amp [Електронний ресурс]. Режим доступу: https://amp.ru.autograndad.com/6181353/1/maskirovka-dannykh.html

9. “Practices for securing data”, DataGuise [Електронний ресурс]. Режимдоступу: https://dataguise.com/pdf/Dataguise_WP_Why_Add_Masking.pdf

10. 0.Dalenius, T., and Reiss, S. P. “Data Swapping: A Technique for Disclosure Control,” Journal of Statistical Planning and Inference, (2012), ст. 73-85.:

11. Arxiv [Електронний ресурс]. Режим доступу: https://arxiv.org/ftp/arxiv/papers/1406/1406.5751.pdf

12. “Data masking tools”, Wisdomplexu [Електронний ресурс]. Режим доступу: https://wisdomplexus.com/blogs/data-masking-tools/

13. “Masking sensitive data”, Anti-malware [Електронний ресурс]. Режим доступу: https://www.anti-malware.ru/practice/solutions/Masking-sensitive-data

14. Oracle Inc., "Data Masking Best Practices", White Paper, 2010

15. “Guide to understanding various types of data masking”, Smartdatacollective [Електронний ресурс]. Режим доступу: https://www.smartdatacollective.com/guide-to-understanding-various-types-of-data-masking/

16. Severalnines [Електронний ресурс]. Режим доступу: https://severalnines.com/database-blog/backup-postgresql-using-pgdump-and-pgdumpall

17. Quares [Електронний ресурс]. Режим доступу: https://quares.ru/?id=87531

18. “Signup login form in PHP”, Studentstutorial [Електронний ресурс]. Режим доступу: https://www.studentstutorial.com/php/signup-login-form-in-php-mysql

19. “Add columns tutorial”, Sql [Електронний ресурс]. Режим доступу: https://www.sqltutorial.org/sql-add-column/

20. A. Sen and A. P. Sinha, "A Comparison of Data Warehousing Methodologies", Communications of the ACM, ст. 79-84, 2005.

21. Studopedia [Електронний ресурс]. Режим доступу: https://studopedia.net/6_76694_IdSubject-INT-NOT-NULL-AUTOINCREMENT-TitleSubject-VARCHAR-NOT-NULL-PRIMARY-KEY-idSubject.html

22. “Security evaluation and enhancement by specific mask encodings”, Pastel [Електронний ресурс]. Режим доступу: https://pastel.archives-ouvertes.fr/pastel-00913472/document

23. Lirias [Електронний ресурс]. Режим доступу: https://lirias.kuleuven.be/retrieve/393473

24. Nsuworks [Електронний ресурс]. Режим доступу: https://nsuworks.nova.edu/cgi/viewcontent.cgi?article=2009&context=gscis_etd

25. “Survey on privacy preservation technique”, Ijert [Електронний ресурс]. Режим доступу: https://www.ijert.org/survey-on-privacy-preservation-technique-data-masking

26. "Masking Personal Identifiable SQL Server Data, R. Dobson, [Електронний ресурс]. Режим доступу: https://www.mssqltips.com/sqlservertip/3091/masking-personal-identifiable-sql-server-data/

27. "Dynamic Data Masking", Microsoft SQL Server 2017, [Електронний ресурс]. Режим доступу: https://docs.microsoft.com/en-us/sql/relational-databases/security/dynamic-datamasking?view=sql-server-2017

28. “What is a data warehouse?”, W. H. Inmon. [Електронний ресурс]. Режим доступу: www.cait.wustl.edu/cait/papers/prism/vol1_no1/

29. Voltage Security Solution for Data De-Identification [Електронний ресурс]. Режим доступу: https://www.voltage.com/wpcontent/uploads/Voltage_UC_Data_DeIdentification.pdf

30. “Why Add Data Masking To Your Best Practices for Securing Sensitive Data” , Dataguise inc. [Електронний ресурс]. Режим доступу: www.dataguise.com/2010

31. “Comparisions of Data Masking products", FireCompass [Електронний ресурс]. Режим доступу: http://products.cisoplatform.com/security/market/data-masking-dm

32. “Data masking concept tools”, Medium [Електронний ресурс]. Режим доступу: https://medium.com/petabytz/data-masking-concept-tools-masking-polices-healthcare-data-masking-3c28ca03a30a

33. “Important piece of the data security puzzle”, Encryptech [Електронний ресурс]. Режим доступу: https://www.encryptech.co.za/data-masking-an-important-piece-of-the-data-security-puzzle/

ДОДАТОК А

Ініціалізація під’єднуємої таблиці.

```
SET statement_timeout = 0;
SET lock_timeout = 0;
SET idle_in_transaction_session_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET client_min_messages = warning;
SET row_security = off;

REATE TABLE public.employees (

                    id int not null,
                    first_name varchar(100) not null,
                    last_name varchar(100) not null,
                    gender varchar(1) not null,
                    personal_email varchar(100) not null,
                    ssn varchar(20) not null,
                    birth_date date not null,
                    start_date date not null,
                    office varchar(100) not null,
                    title varchar(100) not null,
                    org varchar(100) not null,
                    accrued_holidays smallint not null,
                    salary int not null,
                    bonus int not null
```

```
);

ALTER TABLE "public"."employees" ADD PRIMARY KEY ("id");
ALTER TABLE public.employees OWNER TO "user";

COPY public.employees (id, first_name, last_name, gender, personal_email, ssn,
birth_date, start_date, office, title, org, accrued_holidays, salary, bonus) FROM stdin;
100170  Cynthia Chapman F cchapman@gmail.com 109-85-5285
        1/14/1998 0:00  1/9/2019  Chicago  Engineer  Platform
        10  88600 3600
100214  Glenn Mendez M gmendez@hotmail.com 195-95-8914
        2/14/1973 0:00  2/6/2008  Chicago  Engineer  DevOps
        2  103600  7300
100249  Raymond Barnett M rbarnett@yahoo.com 458-69-4479
        1/11/1974 0:00  1/2/2009  Austin  VP  DevOps  8
        138200  10500
```

ДОДАТОК Б

Скрипт для генерації випадкових даних.

```
import time
import sys
import csv
import DataMasker
from dateutil.parser import parse

def generateDemographics() :
        member = {}
        member["ssn"] = DataMasker.generateSSN()
        member['creditCard'] = DataMasker.generateCreditCardNumber()
        name = DataMasker.generateName()
        member['firstName'] = name['firstName']
        member['lastName'] = name['lastName']

        address = DataMasker.generateAddress()
        member['address'] = address["address"]
        member['city'] = address["city"]
        member['state'] = address["state"]
        member['postalCode'] = address["postalCode"]

        contact = DataMasker.generateContact()
        member["phone"] = contact["phone"]
        member["email"] = contact["email"]

        return member
```

```
def main(records):
        start = time.time()
        list = []
        for x in range(0, records):
                list.append(generateDemographics())


        with open('demographicsData.csv', 'w') as outFile:
                fieldnames = ['ssn','creditCard','firstName','lastName',
                        'address','city','state','postalCode','email','phone']
                writer = csv.DictWriter(outFile, fieldnames=fieldnames)
                writer.writeheader()
                for row in list:
                        writer.writerow(row)
        outFile.close()
        end = time.time()
        elapsed = end - start

if __name__ == "__main__":
        records = 100
        print('Generating %d records'% records)
        main(records)
```