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
_____Наталія ЛУКОВА-ЧУЙКО
«14» червня 2022р.

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: «Розробка механізмів захисту інформації в мобільних додатках»

Виконавець: студент IV курсу, групи КБ-41

_____ **Андрій ХОМИН**
(підпис) (ім'я прізвище)

	Прізвище, ініціали	**Підпис**
Керівник	Яніна ШЕСТАК	

Нормоконтроль	Юрій ЩЕБЛАНІН	

Київ 2022

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
_____Наталія ЛУКОВА-ЧУЙКО
«01» листопада 2021 р.

ЗАВДАННЯ
на виконання дипломної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

Студентові **КБ-41** **Хомин Андрій Петрович**
(група) (прізвище ім'я по-батькові)

Тема дипломної роботи Розробка механізмів захисту інформації в мобільних додатках

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема дипломної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №5 від 29.10.2021 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Стек технологій для розробки мобільних додатків, програмна частина серверу, алгоритми хешування та аутентифікації

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Нормативно-правова база про захист інформації, структура мобільних-додатків, архітектура клієнт-серверної інфраструктури, основні вразливості клієнт-серверної інфраструктури, захист збережених даних мобільним додатком

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Створення механізму захисту збережених даних мобільним додатком

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання: 29.10.2021 року

Завдання видав _____ Яніна ШЕСТАК
(підпис) (ініціали, прізвище)

Завдання прийняв до виконання _____ Андрій ХОМИН
(підпис) (ініціали, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	29.10.2021 – 02.11.2021	*виконано*
2	Аналіз літератури	03.11.2021 – 14.11.2021	*виконано*
3	Розгляд структури мобільних додатків	15.11.2021 – 12.12.2021	*виконано*
4	Розгляд структури клієнт-серверної архітектури	13.12.2021 – 17.12.2021	*виконано*
5	Створення рекомендацій для захисту клієнт-серверної архітектури	18.12.2021 – 11.04.2022	*виконано*
6	Вибір технологічного стеку	20.04.2022 – 15.05.2022	*виконано*
7	Створення клієнт-серверної архітектури для аутентифікації	16.05.2022 – 23.05.2022	*виконано*
8	Створення мобільного додатку з аутентифікацією	24.05.2022 – 28.05.2022	*виконано*
9	Створення механізму безпечного зберігання даних мобільним додатком	29.05.2022 – 01.06.2022	*виконано*
10	Оформлення пояснювальної записки	02.06.2022 – 06.06.2022	*виконано*
11	Підготовка до захисту	07.06.2022 – 13.06.2022	*виконано*

Завдання видав _____ Яніна ШЕСТАК
(підпис) (ініціали, прізвище)

Завдання прийняв до виконання _____ Андрій ХОМИН
(підпис) (ініціали, прізвище)

Термін подання дипломної роботи до ЕК 06 червня 2022 року

РЕФЕРАТ

Пояснювальна записка дипломної роботи складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел та додатків. Основний текст займає 61 сторінку, включає в себе зміст, вступ, три розділи дипломної роботи, висновки та список джерел. У пояснювальній записці дипломної роботи міститься 18 рисунків.

Метою роботи є реалізація механізмів захисту інформації в мобільних-додатках.

Для досягнення зазначеної мети поставлено наступні завдання:

- дослідити структуру клієнт-серверної архітектури
- провести аналіз вразливостей клієнт-серверної архітектури
- дослідити структуру мобільних додатків
- побудувати клієнт-серверну архітектуру
- створити мобільний додаток із механізмом захисту збереженої інформації

Об'єктом дослідження є процес виявлення та протидії загрозам, що властиві сучасним мобільним додаткам.

Предметом дослідження є набір механізмів, що реалізують механізми захисту інформації в мобільних додатках.

Практичною цінністю отриманих результатів є програмна реалізація механізмів захисту інформації в мобільних додатках.

Ключові слова: мобільний додаток, захист баз даних, захист інформації, клієнт-серверна архітектура, захист серверів.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

AWS – Amazon Web Services
API – Application Programming Interface
(D)DoS – (Distributed) Denial-of-Service
IT – Information Technology
HTTP(S) – HyperText Transfer Protocol (Secure)
URL – Uniform Resource Locator
(No)SQL – (Not only) Structured Query Language
VPN – Virtual Private Network
IOS – iPhone Operating System
APDU – Application Protocol Data Unit
HMAC – Hash-Based Message Authentication Code
ECDSA – Application Protocol Data Unit
DES – Data Encryption Standard
AES – Advanced Encryption Standard
RSA – Rivest Shamir Adleman
REST – Representational State Transfer
Wi-Fi – Wireless Fidelity
IP – Internet Protocol
SSH – Secure Shell
SSL – Secure Sockets Layer
DV SSL – Domain Validation Secure Sockets Layer
OV SSL – Organization Validation Secure Sockets Layer
EV SSL – Extended Validation Secure Sockets Layer
MDC – Multi-Domain Certificate
UCC – Unified Communications Certificate
TLS – Transport Layer Security
ДНК – Дезоксирибонуклеїнова кислота
ЄС – Європейський Союз
ОС – операційна система
СУБД – система управління базами даних
ПЗ – програмне забезпечення

ЗМІСТ

ВСТУП

У сучасному світі нам потрібні інформаційні технології, щоб налагодити швидший зв’язок, підтримувати електронне зберігання та забезпечувати захист інформації. Інформаційні технології мають важливе значення в нашому житті, оскільки вони допомагають впоратися з щоденними динамічними речами. Технології пропонують різні інструменти для прискорення розвитку та обміну інформацією.

Інформаційні технології включають вивчення та застосування комп’ютерів та будь-якого типу телекомунікацій , які зберігають, отримують, вивчають, передають, маніпулюють даними та надсилають інформацію. Інформаційні технології включають поєднання апаратного та програмного забезпечення, яке використовується для виконання основних завдань, які потрібні людям і які використовуються в повсякденному житті.

Зараз багато компаній мають IT-відділи для управління комп’ютерами, мережами та іншими технічними сферами свого бізнесу. IT-вакансія включає комп’ютерне програмування, мережеве адміністрування, комп’ютерну інженерію, веб-розробку, технічну підтримку та багато інших пов’язаних професій.

Оскільки ми живемо у «світі інформації», інформаційні технології стали частиною нашого повсякденного життя. У найближчі десятиліття багато корпорацій створять так звані «IT-відділи» для управління комп’ютерними технологіями, пов’язаними з їхнім бізнесом.

Інформаційні технології допомагають будувати і розвивати комерційний і бізнес-сектор і створювати максимально можливий результат. Час, що витрачається різними секторами на створення бізнесу, тепер зведено до мінімуму завдяки розвитку інформаційних технологій. Він забезпечує електронну безпеку, зберігання та ефективний зв’язок.

Для проведення роботи інформаційним технологіям потрібні комп'ютерні додатки. Комп’ютери з’єднують IT з різними організаціями світу. Це допомагає співробітникам вести облік своїх численних клієнтів різних компаній. Це допомагає

пацієнтам зв’язатися з лікарями онлайн та отримати поради щодо своїх проблем зі здоров’ям. Крім того, система може належним чином керувати записами пацієнтів.

Для збору інформації використовуються програмування/кодування, перетворення даних, пошук і зберігання даних, системний аналіз. Навіть сектор освіти різко змінився з приходом інформаційних технологій. Комп’ютери, програмне забезпечення та Інтернет дуже допомагають, щоб правильно вести бізнес і отримувати очікувані результати.

Зараз компанії мають віртуальні сховища, які є новою формою системи зберігання, яка дозволяє користувачам зберігати або вилучати свої документи. IT-відділ забезпечує потужну комунікаційну систему для ефективного спілкування.

Використання комп’ютерів та Інтернету підвищує якість освіти. Педагогічний метод викладання та навчання вдосконалюється, а IT сприяє вдосконаленню шкільних систем, діяльності учнів та практик викладання.

Студенти більш відкриті для навчання за допомогою сучасних технологій і більше зосереджуються на онлайн-викладанні. Їхні методи навчання залежать від живої взаємодії з вчителями та спеціальних класів для особливих дітей.

Студенти не зобов’язані використовувати той самий старий традиційний метод навчання. І все це стало можливим завдяки впровадженню інформаційних технологій в освітній сфері і важливості технологій можна побачити.

Ауру інформаційних технологій можна побачити майже в усіх сферах, включаючи роботу, навчання, дозвілля та здоров’я. Від міністерств до аудиторій, кожен сектор використовує IT для досягнення найкращих результатів.

Лікарі також використовують інформаційні технології, щоб перевіряти записи, історію пацієнта та призначену їм дозу для відповідного пересування. Використання інформаційних технологій також можна побачити в сільському господарстві та для підвищення продуктивності. Супутники пов’язані з сільським господарством для прогнозування мусонів і смогу. Завдяки технології дронів можливий масовий збір даних, обстеження земель, використання пестицидів, посів насіння, зрошення водою та використання добрив.

Метою дипломної роботи є розробка механізмів захисту інформації в мобільних додатках.

Для досягнення мети дипломної роботи поставлені наступні завдання:

- дослідження методів захисту мобільних додатків, баз даних та серверів;
- аналіз способів зберігання і передавання даних в мобільних додатках;
- реалізація механізму захисту інформації в мобільних додатках.

РОЗДІЛ 1
ДОСЛІДЖЕННЯ МЕТОДІВ ЗАХИСТУ КЛІЄНТ-СЕРВЕРНОЇ ІНФРАСТРУКТУРИ

1.1 Принципи інформаційної безпеки

Інформаційна безпека охоплює інструменти та процеси, які організації використовують для захисту інформації. Це включає налаштування політики, які запобігають доступу неавторизованих людей до ділової чи особистої інформації. Інформаційна безпека - це сфера, що розвивається та охоплює широкий спектр областей, від безпеки мережі та інфраструктури до тестування та аудиту.

Інформаційна безпека захищає конфіденційну інформацію від несанкціонованих дій, включаючи перевірку, модифікацію, запис, а також будь-які порушення чи знищення.

Наслідками інцидентів безпеки є крадіжка приватної інформації, підробка даних і видалення даних. Атаки можуть порушити робочі процеси та завдати шкоди репутації компанії, а також мати відчутні витрати.

Організації повинні виділяти кошти на безпеку та гарантувати, що вони готові виявляти, реагувати на такі атаки, як фішинг, зловмисне програмне забезпечення, віруси, інсайдери та програми-вимагачі, реагувати на них та активно запобігати .

Основними принципами інформаційної безпеки є конфіденційність, цілісність і доступність. Кожен елемент програми інформаційної безпеки повинен бути розроблений для реалізації одного або кількох із цих принципів. Принципи інформаційної безпеки:

- Конфіденційність - заходи щодо конфіденційності покликані запобігти несанкціонованому розголошенню інформації. Метою принципу конфіденційності є збереження конфіденційності особистої інформації та забезпечення того, щоб вона була видимою та доступною лише для тих осіб, які володіють нею або потребують її для виконання своїх організаційних функцій [1].

- Цілісність - узгодженість включає захист від несанкціонованих змін (додавання, видалення, зміни) даних. Принцип цілісності гарантує, що дані є точними і надійними та не змінюються неправильно, випадково чи зловмисно [2].
- Доступність - це захист здатності системи робити програмні системи та дані повністю доступними, коли це потрібно користувачеві (або у визначений час). Мета доступності полягає в тому, щоб зробити технологічну інфраструктуру, програми та дані доступними, коли вони необхідні для організаційного процесу або для клієнтів організації [3].

Інформаційна безпека відрізняється від кібербезпеки як за обсягом, так і за призначенням. Ці два терміни часто використовуються як взаємозамінні, але кібербезпека є підкатегорією інформаційної безпеки. Інформаційна безпека - це широка область, яка охоплює багато областей, таких як фізична безпека, безпека кінцевих точок, шифрування даних і безпека мережі. Це також тісно пов’язане із забезпеченням інформації, яке захищає інформацію від загроз, таких як стихійні лиха та збоїв серверів [4].

Кібербезпека насамперед спрямована на загрози, пов’язані з технологіями, за допомогою методів та інструментів, які можуть запобігти або пом’якшити їх. Іншою пов’язаною категорією є безпека даних, яка зосереджена на захисті даних організації від випадкового або зловмисного доступу неавторизованих сторін.

Політика інформаційної безпеки - це набір правил, якими керуються люди під час використання IT-активів. Компанії можуть створювати політику інформаційної безпеки, щоб гарантувати, що співробітники та інші користувачі дотримуються протоколів і процедур безпеки. Політики безпеки призначені для того, щоб лише авторизовані користувачі могли отримати доступ до конфіденційних систем та інформації.

Створення ефективної політики безпеки та вжиття заходів для забезпечення відповідності є важливим кроком до запобігання та пом’якшення загроз безпеці. Щоб політика була дійсно ефективною, потрібно часто оновлювати її на основі змін компанії, нових загроз, висновків, зроблених із попередніх порушень, а також змін у системах та інструментах безпеки.

Щоб задовольнити потреби та терміновість різних відділів організації, необхідно розгорнути систему винятків із процесом затвердження, що дозволить відділам або особам відхилятися від правил за конкретних обставин.

Існують сотні категорій загроз інформаційній безпеці та мільйони відомих векторів загроз. Ми розглянемо деякі з ключових загроз, які є пріоритетними для команд безпеки на сучасних підприємствах. Основні загрози інформаційної безпеки:

- Незахищені або погано захищені системи - швидкість і розвиток технологій часто призводять до компромісів у заходах безпеки. В інших випадках системи розробляються без урахування безпеки і залишаються в експлуатації в організації як застарілі системи. Організації повинні ідентифікувати ці погано захищені системи та пом’якшити загрозу, захистивши чи виправивши їх, знявши з експлуатації або ізолюючи їх.
- Атаки в соціальних мережах - багато людей мають акаунти в соціальних мережах, де вони часто ненавмисно діляться великою кількістю інформації про себе. Зловмисники можуть здійснювати атаки безпосередньо через соціальні мережі, наприклад, поширюючи шкідливе програмне забезпечення через повідомлення в соціальних мережах, або опосередковано, використовуючи інформацію, отриману з цих сайтів, для аналізу вразливості користувачів, організацій і використання їх для розробки атаки.
- Соціальна інженерія - передбачає, що зловмисники надсилають електронні листи та повідомлення, які обманом спонукають користувачів виконувати дії, які можуть поставити під загрозу їх безпеку або розкрити особисту інформацію. Зловмисники маніпулюють користувачами, використовуючи психологічні тригери, такі як цікавість, терміновість або страх. Оскільки джерело повідомлення соціальної інженерії, здається, є довіреним, люди, швидше за все, підкоряться, наприклад, натиснувши посилання, яке встановлює зловмисне програмне забезпечення на їхній пристрій, або надавши особисту інформацію, облікові дані чи фінансові дані. Організації можуть пом’якшити соціальну інженерію, повідомляючи користувачів про її небезпеку та навчаючи їх виявляти та уникати підозрюваних повідомлень соціальної інженерії. Крім того, технологічні

системи можуть використовуватися для блокування соціальної інженерії на її джерелі або запобігання користувачам виконувати небезпечні дії, такі як натискання невідомих посилань або завантаження невідомих вкладень [5].

- Шкідливе програмне забезпечення на кінцевих точках - користувачі організацій працюють із різноманітними кінцевими пристроями, включаючи настільні комп'ютери, ноутбуки, планшети та мобільні телефони, багато з яких є приватними та не підконтрольними організації, і всі вони регулярно підключаються до Інтернету. Основною загрозою для всіх цих кінцевих точок є шкідливе програмне забезпечення, яке може передаватися різними способами, може призвести до компрометації самої кінцевої точки, а також може призвести до ескалації привілеїв іншим організаційним системам. Традиційного антивірусного програмного забезпечення недостатньо для блокування всіх сучасних форм зловмисного програмного забезпечення, і розробляються більш просунуті підходи до захисту кінцевих точок [6].
- Відсутність шифрування - процеси шифрування кодують дані, щоб їх могли декодувати лише користувачі з секретними ключами. Це дуже ефективно для запобігання втраті або пошкодженню даних у разі втрати або крадіжки обладнання, а також у випадку, якщо зловмисники зламали організаційні системи. На жаль, цей захід часто ігнорується через його складність та відсутність юридичних зобов'язань, пов'язаних із належним виконанням. Організації все частіше впроваджують шифрування, купуючи пристрої зберігання даних або використовуючи хмарні служби, які підтримують шифрування, або використовуючи спеціальні засоби безпеки.
- Неправильна конфігурація безпеки - сучасні організації використовують величезну кількість технологічних платформ та інструментів, зокрема веб-додатків, баз даних та програм «Програмне забезпечення як послуга» або «Інфраструктура як послуга» від таких провайдерів, як Amazon Web Services. Платформи корпоративного рівня та хмарні сервіси мають функції безпеки, але вони повинні бути налаштовані організацією. Неправильна конфігурація безпеки через недбалість або людську помилку може призвести до порушення безпеки. Іншою проблемою є

«зміщення конфігурації», коли правильна конфігурація безпеки може швидко застаріти і зробити систему вразливою, невідомо для IT або співробітників служби безпеки. Організації можуть пом’якшити неправильну конфігурацію безпеки, використовуючи технологічні платформи, які постійно контролюють системи, виявляють прогалини в конфігурації та попереджають або навіть автоматично усувають проблеми конфігурації, які роблять системи вразливими [7].

Інформаційна безпека призначена для захисту організацій від зловмисних атак. Існує два основних типи атак: активні та пасивні. Вважається, що активні атаки важче запобігти, і зосереджено на їх виявленні, пом’якшенні та відновленні від них. Пасивні атаки легше запобігти за допомогою жорстких заходів безпеки.

Активна атака передбачає перехоплення комунікації або повідомлення та зміну його для шкідливого впливу. Існує три поширені варіанти активних атак:

- Переривання - зловмисник перериває вихідну комунікацію та створює нові шкідливі повідомлення, прикидаючись однією зі сторін спілкування.
- Модифікація - зловмисник використовує наявні комунікації та відтворює їх, щоб обдурити одну зі сторін, що спілкуються, або модифікує їх, щоб отримати перевагу.
- Фабрикація - створює підроблені або синтетичні комунікації, як правило, з метою досягнення відмови в обслуговуванні. Це заважає користувачам отримати доступ до систем або виконувати звичайні операції [8].

При пасивній атаці зловмисник відстежує систему та незаконно копіює інформацію, не змінюючи її. Потім вони використовують цю інформацію для порушення роботи мереж або компрометації цільових систем [9].

Зловмисники не вносять жодних змін у комунікаційні або цільові системи. Це ускладнює виявлення. Однак шифрування може допомогти запобігти пасивним атакам, оскільки воно обфускує дані, що ускладнює зловмисникам їх використання.

Інформаційна безпека знаходиться в постійній взаємодії із законами та правилами місць, де організація здійснює бізнес. Норми щодо захисту даних у всьому світі зосереджуються на підвищенні конфіденційності персональних даних і

встановлюють обмеження на те, як організації можуть збирати, зберігати та використовувати дані клієнтів.

Конфіденційність даних зосереджена на ідентифікаційній інформації і в першу чергу стосується того, як ці дані зберігаються та використовуються, включаючи будь-які дані, які можна пов'язати безпосередньо з користувачем, наприклад ім'я, ідентифікаційний номер, дату народження, фізичну адресу або номер телефону. Також можуть включатися такі дані, як публікації в соціальних мережах, зображення профілю та IP-адреси.

Найвідомішим законом про конфіденційність в ЄС є Загальний регламент захисту даних. Цей регламент охоплює збір, використання, зберігання, безпеку та передачу даних, що стосуються резидентів ЄС.

Закон поширюється на будь-яку організацію, яка веде бізнес із громадянами ЄС, незалежно від того, чи знаходиться сама компанія в Європейському Союзі чи за його межами. Порушення інструкцій може призвести до штрафів у розмірі до 4% світових продажів [10].

Основними цілями закону є:

- встановлення конфіденційності персональних даних як основного права людини;
- виконання вимог критеріїв конфіденційності;
- стандартизація того, як застосовуються правила конфіденційності.

Закон передбачає захист таких типів даних:

- особиста інформація, така як ім'я, ідентифікаційний номер, дата народження або адреса;
- веб-дані, такі як IP-адреса, файли cookie, місцезнаходження тощо;
- інформація про стан здоров'я, включаючи діагноз і прогноз;
- біометричні дані, включаючи голосові дані, ДНК та відбитки пальців;
- приватні комунікації;
- фото та відео;
- культурні, соціальні чи економічні дані.

В Україні також існує закон про захист інформації, який називається Закон України “Про інформацію”. Цей Закон регулює відносини щодо створення, збирання, одержання, зберігання, використання, поширення, охорони, захисту інформації. В цьому законі описані основні принципи інформаційних відносин:

- гарантованість права на інформацію;
- відкритість, доступність інформації, свобода обміну інформацією;
- достовірність і повнота інформації;
- свобода вираження поглядів і переконань;
- правомірність одержання, використання, поширення, зберігання та захисту інформації;
- захищеність особи від втручання в її особисте та сімейне життя [11].

1.2 Методи захисту мобільних пристроїв

Безпека мобільних пристроїв - це заходи, призначені для захисту конфіденційної інформації, яка зберігається і передається на ноутбуках, смартфонах, планшетах та інших портативних пристроях. В основі безпеки мобільних пристроїв лежить мета запобігти доступу неавторизованих користувачів до корпоративної мережі. Це один з аспектів повного плану безпеки підприємства [12].

Оскільки більше половини бізнес-комп’ютерів тепер мобільні, портативні пристрої створюють чіткі проблеми з безпекою мережі, яка повинна враховувати всі місця та використання, які потрібні співробітникам мережі компанії. Потенційні загрози для пристроїв включають шкідливі мобільні програми, фішингові шахрайства, витік даних, шпигунське програмне забезпечення та незахищені мережі Wi-Fi. Крім того, підприємства повинні враховувати можливість втрати працівником мобільного пристрою або його крадіжки. Щоб уникнути порушення безпеки, компанії повинні вживати чіткі превентивні заходи для зниження ризику [13].

Безпека мобільного пристрою або керування мобільним пристроєм забезпечує наступне:

- відповідність нормативним вимогам;

- застосування політики безпеки;
- дистанційне керування оновленнями пристрою;
- контроль додатків;
- автоматизована реєстрація пристрою;
- резервне копіювання даних [14].

Перш за все, безпека мобільних пристроїв захищає підприємство від невідомих або зловмисних сторонніх осіб, які можуть отримати доступ до конфіденційних даних компанії.

Захист мобільних пристроїв вимагає багаторівневого підходу та інвестицій у корпоративні рішення. Хоча існують ключові елементи безпеки мобільних пристроїв, кожна організація повинна знайти те, що найкраще підходить для її мережі [15].

Кілька методів безпеки мобільних пристроїв:

- Захист паролем - один із найпростіших способів запобігти несанкціонованому доступу до мобільного пристрою - створити надійний пароль, але слабкі паролі все ще залишаються постійною проблемою, що сприяє більшості злому даних. Ще одна поширена проблема безпеки полягає в тому, що працівники використовують той самий пароль для свого мобільного пристрою, електронної пошти та кожного облікового запису, пов’язаного з роботою. Дуже важливо, щоб співробітники створювали надійні, унікальні паролі і створювали різні паролі для різних облікових записів [16].
- Використовуйте біометричні дані - замість того, щоб покладатися на традиційні методи захисту мобільного доступу, такі як паролі, деякі компанії використовують біометричні дані як безпечнішу альтернативу. Біометрична аутентифікація - це коли комп’ютер використовує для ідентифікації та доступу вимірювані біологічні характеристики, такі як розпізнавання обличчя, відбитків пальців, голосу або райдужної оболонки ока. Кілька методів біометричної аутентифікації тепер доступні на смартфонах, їх легко налаштувати та використовувати працівникам.

- Уникайте загальнодоступного Wi-Fi - мобільний пристрій настільки безпечний, як і мережа, через яку він передає дані. Компанії повинні інформувати співробітників про небезпеку використання загальнодоступних мереж Wi-Fi, хакери можуть легко зламати пристрій, отримати доступ до мережі та вкрасти дані. Найкращий захист - заохочувати розумну поведінку користувачів і забороняти використання відкритих мереж Wi-Fi, незалежно від зручності [17].
- Остерігайтеся програм - шкідливі програми є одними з найбільш швидкозростаючих загроз для мобільних пристроїв. Коли працівник несвідомо завантажує його з робочих або особистих причин, він надає несанкціонований доступ до мережі та даних компанії. Щоб подолати цю зростаючу загрозу, у компаній є два варіанти: проінструктувати співробітників про небезпеку завантаження несанкціонованих програм або взагалі заборонити працівникам завантажувати певні програми на свої телефони.
- Шифрування мобільного пристрою - більшість мобільних пристроїв мають вбудовану функцію шифрування. Користувачам потрібно знайти цю функцію на своєму пристрої та ввести пароль для шифрування свого пристрою. За допомогою цього методу дані перетворюються в код, доступ до якого можуть отримати лише авторизовані користувачі. Це важливо в разі крадіжки і запобігає несанкціонованому доступу [18].

Існує багато аспектів повного плану безпеки. Загальні елементи мобільного рішення безпеки включають наступне:

- Безпека електронної пошти - електронна пошта є найпопулярнішим способом для хакерів для поширення програм-вимагачів та інших шкідливих програм. Для боротьби з такими атаками дуже важливо, щоб підприємства були озброєні розширеною системою захисту електронної пошти, яка може швидше виявляти, блокувати та усувати загрози, запобігти будь-якій втраті даних і захистити важливу інформацію під час передачі за допомогою наскрізного шифрування.
- Захист кінцевої точки - цей підхід захищає корпоративні мережі, до яких віддалено доступні мобільні пристрої. Безпека кінцевих точок захищає компанії, гарантуючи, що портативні пристрої відповідають стандартам безпеки і швидко

сповістять групи безпеки про виявлені загрози, перш ніж вони можуть завдати шкоди. Захист кінцевої точки також дозволяє IT-адміністраторам контролювати робочі функції та стратегії резервного копіювання даних.

- VPN - віртуальна приватна мережа, розширює приватну мережу через загальнодоступну мережу. Це дозволяє користувачам надсилати та отримувати дані через спільні чи загальнодоступні мережі, як якщо б їхні комп'ютерні пристрої були безпосередньо підключені до приватної мережі. Технологія шифрування VPN дозволяє віддаленим користувачам і філіалам отримувати безпечний доступ до корпоративних програм і ресурсів.
- Безпечний веб-шлюз - захищає від загроз онлайн-безпеці, дотримуючись політики безпеки компанії та захищаючи від фішингу та зловмисного програмного забезпечення в режимі реального часу. Це особливо важливо для хмарної безпеки, оскільки цей тип захисту може виявити атаку в одному місці та негайно зупинити її в інших гілках.

Загроза зазнати атаки з боку кіберзлочинців дуже реальна для будь-якого бізнесу, великого та малого. Серйозне порушення даних також не тільки вплине на фінансовий стан організації, але може призвести до довгострокової і навіть постійної шкоди репутації бізнесу.

1.3 Методи захисту серверів

Однією з найпоширеніших причин злому даних є слабка безпека сервера. Оскільки зловмисники розробляють все більш витончені способи атаки на сервери, найкращі методи безпеки серверів дуже важливі для захисту бізнесу та конфіденційних даних [19].

Безпека сервера - це всі процеси та інструменти, які використовуються для захисту конфіденційних даних, ресурсів і активів, що зберігаються на сервері.

Сервери часто стають мішенню кіберзлочинців, оскільки вони, як правило, зберігають конфіденційну та цінну інформацію. Кіберзлочинці завжди намагаються використати вразливі місця в безпеці серверів для фінансової вигоди та інших причин.

У більшості IT-інфраструктур сервери є ядром усієї інфраструктури. Сервер - це те, що дає всім користувачам доступ до тих самих ресурсів, функцій та інформації віддалено. Коли сервер зламано під час атаки, велика ймовірність того, що вся мережа та система також будуть скомпрометовані [20].

Отже, підтримка безпеки сервера, очевидно, важлива. Однак навіть дуже невелика вада, як-от слабкий пароль, невдале оновлення програмного забезпечення та інші відносно прості людські помилки, можуть призвести до зламаного сервера та значних збитків для організації.

Ось чому для забезпечення ефективності безпеки сервера ми повинні враховувати різні рівні - від виявлення та керування потенційними проблемами у вашій мережі до захисту операційних систем сервера, захисту будь-якого програмного забезпечення та додатків, розміщених на вашому сервері, і в найдрібніших рівня, захищаючи конфіденційні та регульовані дані, розміщені на сервері [21].

Поширені вразливості безпеки сервера:

- Слабкі паролі - щоб вгадати слабкі паролі, хакери можуть використовувати зловмисних ботів для атак грубої сили, або атак із заповненням облікових даних. Якщо зловмисники заволодіють обліковими даними адміністратора, вони зможуть отримати доступ до ваших серверів і спричинити порушення даних. Зловмисники також можуть продати ваші облікові дані в мережі. Потрібно переконатися, що використовується достатньо надійний пароль. Можна використовувати служби диспетчера паролів, щоб переконатися, що завжди використовуються надійні паролі.
- Керування патчами - важливо використовувати службу керування виправленнями, щоб переконатися, що будь-які зміни в коді належним чином перевірені перед встановленням та надходять із надійного джерела.
- Помилки оновлення програмного забезпечення - жодне програмне забезпечення не є на 100% досконалим, і відповідальні виробники усувають уразливості свого програмного забезпечення за допомогою виправлень безпеки та виправлень. Однак, коли виробник програмного забезпечення випускає

виправлення, він також оголошує про цю вразливість громадськості. Ось чому завжди повинно оновлюватися програмне забезпечення, як тільки з’являться виправлення. Кіберзлочинці постійно експлуатують вразливості програмного забезпечення, тому використання застарілої версії програмного забезпечення значно підвищить ризики злому даних.

- Неправильно налаштовані мережеві порти - неправильна конфігурація сервера, особливо щодо мережевих портів, може бути легко використана кіберзлочинцями. Дуже важливо налаштувати та оптимізувати сервер відповідно до найкращих методів безпеки сервера.
- Застарілі дані - забуті облікові записи часто використовуються хакерами для отримання доступу до сервера. Обов’язково потрібно проводити періодичні очищення старих і застарілих облікових записів.
- Погана фізична безпека - не всі зовнішні загрози безпеці вашого сервера є цифровими за своєю природою. Коли, наприклад, злодії отримують фізичний доступ до вашого сервера, він також скомпрометований.

Основні правила безпеки сервера:

- Регулярно оновлюйте всі програми розміщені на сервері.
- Налаштуйте сервер відповідно до найкращих методів безпеки сервера. Увімкніть лише необхідні програми та служби та вимкніть усі непотрібні.
- Встановіть усі паролі облікових записів і використовуйте достатньо надійні паролі. Видаліть облікові записи за замовчуванням.
- Регулярно відстежуйте повідомлення, пов’язані з безпекою, пов’язані з вашим сервером.
- Використовуйте SSL і TLS відповідно до даних, розміщених на сервері, для реалізації шифрування та аутентифікації.
- Налаштуйте сервер відповідно до найкращих практик виробника. Це може включати встановлення серверного програмного забезпечення на призначеного хост-провайдера, встановлення необхідних засобів контролю доступу до конфіденційних файлів тощо.

- Створення журналів для майбутніх розслідувань і відновлення і налаштуйте файли журналів, щоб фіксувати всі потенційно ризиковані дії.
- Задокументуйте всі зміни, які ви внесли в систему, розміщену на сервері, і перевірте всі запропоновані зміни перед їх запуском.

Методи захисту сервера від зовнішніх атак:

- Використовуйте лише безпечні з'єднання - рекомендується використовувати SSH для встановлення безпечного з’єднання з вашим сервером. SSH фактично може замінити автентифікацію на основі пароля для вашого сервера, і оскільки паролі завжди вразливі до атак грубої сили, SSH-з’єднання краще для захисту ваших серверів [22].

SSH використовує пару криптографічних ключів з відкритим і приватним ключем. Відкритим ключем можна ділитися з іншими, але приватний ключ повинен надійно зберігатися адміністратором сервера. SSH буде ефективно шифрувати всі обмінювані дані. Додатковим методом захисту вашого з’єднання є використання проксі-серверів. Оскільки за допомогою проксі-сервера ваші користувачі приховані за замаскованою IP-адресою проксі-сервера, зловмисникам важче націлитися на вразливі пристрої користувачів, щоб отримати доступ.

- Встановіть рішення для пом’якшення наслідків роботи з ботами - багато атак, націлені на сервери, стають можливими завдяки використанню шкідливих ботів, наприклад, для атак грубої сили, заповнення облікових даних і DoS-атак. Тому рекомендується захистити свій сервер за допомогою відповідного рішення для пом’якшення наслідків. У зв’язку з складністю сучасних поганих ботів, рекомендовано рішення для керування роботами, яке здатне виявляти на основі поведінки.
- Забезпечте використання VPN - рекомендується використовувати приватну мережу для підтримки безпечного обміну даними на сервері та з нього. У приватній мережі користувачі використовуватимуть приватні, невідстежувані IP-адреси, тому хакерам важче ідентифікувати користувача та знайти вразливі місця. Крім того, при підключенні до сервера через VPN ефективно шифруються дані при передачі з сервера на сервер, тому хакери не зможуть вкрасти дані під час передачі.

1.4 Методи захисту баз даних

Безпека бази даних відноситься до набору інструментів, засобів контролю та заходів, призначених для встановлення та збереження конфіденційності, цілісності та доступності бази даних.

Безпека бази даних повинна стосуватися та захищати наступне:

- дані в базі даних;
- систему управління базами даних;
- будь-які пов’язані програми;
- фізичний сервер бази даних або сервер віртуальної бази даних;
- обчислювальну та мережеву інфраструктуру, яка використовується для доступу до бази даних [23].

Безпека баз даних є складним завданням, яке охоплює всі аспекти технологій і практик інформаційної безпеки. Це також, природно, суперечить зручності використання бази даних.

За визначенням, злом даних - це нездатність зберегти конфіденційність даних у базі даних. Наслідки вашому підприємству завдає злом даних, залежить від ряду наслідків або факторів:

- Зламана інтелектуальна власність - може мати вирішальне значення для вашої здатності підтримувати конкурентну перевагу на своєму ринку. Якщо ця інтелектуальна власність буде вкрадена або розкрита, вашу конкурентну перевагу може бути важко або неможливо зберегти чи відновити.
- Шкода репутації бренду - клієнти або партнери можуть не захотіти купувати ваші продукти чи послуги, якщо вони не відчувають, що можуть довіряти вам, щоб захистити ваші чи їхні дані.
- Безперервність бізнесу - деякі підприємства не можуть продовжувати свою діяльність, доки не буде усунено порушення.
- Штрафи за недотримання - фінансові наслідки недотримання глобальних правил, галузеві правила щодо конфіденційності даних, або регіональні правила щодо конфіденційності даних, можуть бути руйнівними для команії.

- Витрати на усунення порушень та сповіщення клієнтів - на додаток до витрат на повідомлення клієнта про порушення, організація, яка порушила правила, повинна оплатити судову експертизу та слідчі дії, кризове управління, ремонт уражених систем тощо.

Багато неправильних конфігурацій програмного забезпечення, вразливості або моделі необережності чи неправильного використання можуть призвести до порушень. Нижче наведено одні з найбільш поширених типів або причин атак на безпеку бази даних та їх причини.

Інсайдерська загроза - це загроза безпеці з будь-якого з трьох джерел з привілейованим доступом до бази даних:

- зловмисний інсайдер, який має намір завдати шкоди;
- недбалий інсайдер, який робить помилки, які роблять базу даних уразливою для атак;
- зловмисник - стороння особа, яка якимось чином отримує облікові дані за допомогою такої схеми, як фішинг, або шляхом отримання доступу до самої бази даних.

Внутрішні загрози є однією з найпоширеніших причин порушень безпеки баз даних і часто є результатом того, що занадто багато співробітників можуть мати облікові дані привілейованого доступу користувачів.

Людська помилка - причиною майже половини усіх зареєстрованих порушень даних залишаються нещасні випадки, слабкі паролі, обмін паролями та інші нерозумні чи неінформовані дії користувачів.

Використання вразливостей програмного забезпечення баз даних - хакери заробляють собі на життя, знаходячи та націлюючись на вразливості в усіх видах програмного забезпечення, включаючи програмне забезпечення для керування базами даних. Усі основні постачальники комерційного програмного забезпечення для баз даних і платформи керування базами даних із відкритим кодом видають регулярні виправлення безпеки для усунення цих вразливостей, але несвоєчасне застосування цих виправлень може підвищити вашу уязвимість [24].

Атаки ін'єкції SQL/NoSQL - загроза, пов’язана з базою даних, передбачає вставку довільних рядків атаки SQL або NoSQL в запити бази даних, які обслуговуються веб-програмами або заголовками HTTP. Організації, які не дотримуються методів безпечного кодування веб-додатків і не проводять регулярне тестування на вразливості, піддаються цим атакам.

Експлуатація переповнення буфера - переповнення буфера відбувається, коли процес намагається записати в блок пам’яті фіксованої довжини більше даних, ніж йому дозволено. Зловмисники можуть використовувати надлишок даних, що зберігаються в суміжних адресах пам'яті, як основу для здійснення атак.

Атаки відмови в обслуговуванні - під час атаки відмови в обслуговуванні зловмисник засипає цільовий сервер бази даних такою кількістю запитів, що сервер більше не може виконувати інші запити від реальних користувачів, і в багатьох випадках сервер стає нестабільний або аварійний.

Шкідливе програмне забезпечення - це програмне забезпечення, створене спеціально для використання вразливостей. Шкідливе програмне забезпечення може надходити через будь-який кінцевий пристрій, що підключається до мережі бази даних.

Атаки на резервні копії - організації, яким не вдається захистити дані резервного копіювання за допомогою таких самих суворих засобів контролю, які використовуються для захисту самої бази даних, можуть бути вразливими до атак на резервні копії [25].

Загрози на бази даних посилюються наступним:

- Зростання обсягів даних - збір, зберігання та обробка даних продовжує зростати в геометричній прогресії майже в усіх організаціях. Будь-які інструменти або методи безпеки даних мають бути високомасштабованими, щоб задовольнити потреби найближчого та віддаленого майбутнього.
- Розширення інфраструктури - мережеві середовища стають дедалі складнішими, особливо коли підприємства переносять робочі навантаження на багатохмарні або гібридні хмарні архітектури, що робить вибір, розгортання та керування рішеннями безпеки все більш складним.

- Дефіцит навичок з кібербезпеки.

Оскільки бази даних майже завжди доступні через мережу, будь-яка загроза безпеці будь-якому компоненту в межах або частині мережевої інфраструктури також є загрозою для бази даних, і будь-яка атака, що впливає на пристрій або робочу станцію користувача, може загрожувати базі даних. Таким чином, безпека бази даних повинна виходити далеко за межі однієї бази даних.

На додаток до впровадження багаторівневих елементів керування безпекою в усьому мережевому середовищі, безпека бази даних вимагає від вас встановлення правильних елементів керування та політики для доступу до самої бази даних. До них належать:

- адміністративні засоби для керування змінами та конфігурацією бази даних;
- додаткові засоби контролю доступу, шифрування, токенізація та маскування;
- засоби для моніторингу активності бази даних та засоби запобігання втраті даних [26].

Політики безпеки баз даних мають бути інтегровані та підтримувати ваші загальні бізнес-цілі, такі як захист критично важливої інтелектуальної власності та ваші політики кібербезпеки та політики безпеки в хмарі. Контроль безпеки, навчання та освітні програми з питань безпеки, а також тестування на проникнення та стратегії оцінки вразливості мають бути встановлені для підтримки ваших офіційних політик безпеки.

Висновки за розділом 1

Середовища клієнт-сервер користуються популярністю, оскільки вони підвищують ефективність обробки додатків, одночасно знижуючи витрати та отримують максимальну вигоду від спільної роботи всіх ресурсів. Ці переваги досягаються шляхом поділу обробки між клієнтською машиною/програмним забезпеченням і серверною машиною/програмним забезпеченням. Кожен процес працює незалежно, але у співпраці та сумісності з іншими машинами та програмами.

Усю незалежну обробку необхідно виконати для завершення запитуваної послуги. Кооперація обробки додатків дає ще одну перевагу клієнт-сервер, вона зменшує мережевий трафік. Оскільки кожен вузол (клієнт та/або сервер) виконує частину обробки всередині себе, мережеве спілкування можна звести до мінімуму.

Існує три компоненти середовища клієнт-сервер: клієнт, сервер і мережа. Мережа поєднує фізичне та функціональне розділення між клієнтом і сервером.

Проте саме ті характеристики, які роблять клієнт-сервер популярними, також роблять його найбільш вразливим до порушень безпеки. Саме розподіл послуг між клієнтом і сервером відкриває їх для пошкодження, шахрайства та неправильного використання. Розгляд безпеки повинен включати сервери, мережі та клієнтські пристрої.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ МЕТОДІВ ЗАХИСТУ ДАНИХ В ДОДАТКАХ

2.1 Безпечне передавання даних через мережу

Інтернет полегшив нам обмін важливою інформацією один з одним, незалежно від того, пов’язана вона з нашим особистим чи професійним життям. Але якщо передана інформація не буде надійно передана нашим призначеним одержувачам, це може призвести до порушення конфіденційності, що може зайняти багато часу для відновлення. Дуже важливо, що коли ми ділимося або передаємо будь-які дані онлайн, вони повинні бути захищені від зловмисників.

Сертифікат SSL – це цифровий сертифікат, який підтверджує автентифікацію веб-сайту та забезпечує зашифроване з’єднання. SSL - протокол безпеки, який створює зашифроване посилання між веб-сервером і веб-браузером.

Компанії та організації повинні додавати сертифікати SSL на свої веб-сайти, щоб захистити онлайн-транзакції та захистити інформацію про клієнтів. SSL забезпечує безпеку інтернет-з’єднань і не дає зловмисникам читати або змінювати інформацію, передану між двома системами.

З моменту свого заснування близько 25 років тому існувало кілька версій протоколу SSL, кожен з яких у певний момент зіткнувся з проблемами безпеки. Далі з’явилася оновлена перейменована версія - TLS, яка використовується й сьогодні. Однак ініціали SSL застрягли, тому нова версія протоколу як і раніше зазвичай називається старою назвою [27].

SSL працює, гарантуючи, що будь-які дані, передані між користувачами і веб-сайтами, або між двома системами, залишаються неможливими для читання. Він використовує алгоритми шифрування даних під час передачі, що не дозволяє хакерам прочитати їх, коли вони передаються через з’єднання. Ці дані містять потенційно конфіденційну інформацію, таку як імена, адреси, номери кредитних карток або інші фінансові дані.

Процес працює так:

- Браузер або сервер намагаються підключитися до веб-сайту (тобто веб-сервера), захищеного за допомогою SSL.
- Браузер або сервер запитують, щоб веб-сервер ідентифікував себе.
- У відповідь веб-сервер надсилає браузеру або серверу копію свого сертифіката SSL.
- Браузер або сервер перевіряє, чи довіряє він сертифікату SSL. Якщо це так, він сигналізує про це веб-серверу.
- Потім веб-сервер повертає підтвердження з цифровим підписом для початку сеансу, зашифрованого SSL.
- Зашифровані дані обмінюються між браузером або сервером і веб-сервером.

Цей процес іноді називають «рукостисканням SSL». Хоча це звучить як тривалий процес, він відбувається за мілісекунди.

Коли веб-сайт захищений сертифікатом SSL, в URL-адресі з’являється акронім HTTPS (що означає безпечний протокол передачі гіпертексту). Без сертифіката SSL відображатимуться лише літери HTTP, тобто без S для Secure. У адресному рядку URL також відображатиметься значок замка. Це свідчить про довіру та заспокоєння тих, хто відвідує веб-сайт.

Щоб переглянути деталі сертифіката SSL, ви можете натиснути на символ замка, розташований на панелі браузера. Деталі, які зазвичай містяться в сертифікатах SSL, включають:

- доменне ім’я, для якого видано сертифікат;
- якій особі, організації чи пристрої його було видано;
- який центр сертифікації його видав;
- цифровий підпис центру сертифікації;
- асоційовані субдомени;
- дата видачі сертифіката;
- дата закінчення терміну дії сертифіката;
- відкритий ключ.

Веб-сайтам потрібні сертифікати SSL, щоб захистити дані користувачів, підтвердити право власності на веб-сайт, запобігти створенню зловмисниками підробленої версії сайту та передати довіру користувачам.

Якщо веб-сайт просить користувачів увійти, ввести особисті дані, наприклад номери кредитних карток, або переглянути конфіденційну інформацію, таку як переваги для здоров'я чи фінансова інформація, важливо зберігати конфіденційність даних. Сертифікати SSL допомагають зберегти конфіденційність взаємодії в Інтернеті.

Існують різні типи сертифікатів SSL з різними рівнями перевірки. Шість основних типів:

- сертифікати розширеної перевірки (EV SSL);
- перевірені сертифікати організації (OV SSL) ;
- сертифікати, перевірені доменом (DV SSL) ;
- сертифікати SSL підстановки (Wildcard SSL);
- багатодоменні сертифікати SSL (MDC) ;
- сертифікати уніфікованих комунікацій (UCC) .

Сертифікати розширеної перевірки (EV SSL) - це найпотужніший і найдорожчий тип сертифіката SSL. Його зазвичай використовують для високопрофільних веб-сайтів, які збирають дані та передбачають онлайн-платежі. Після встановлення цей сертифікат SSL відображає замок, HTTPS, назву компанії та країну в адресному рядку браузера. Відображення інформації про власника веб-сайту в адресному рядку допомагає відрізнити сайт від шкідливих сайтів. Щоб налаштувати сертифікат EV SSL, власник веб-сайту повинен пройти стандартизовану процедуру підтвердження особи, щоб підтвердити, що він має юридичні повноваження на виключні права на домен [28].

Перевірені сертифікати організації (OV SSL) - ця версія сертифіката SSL має аналогічний рівень гарантії, що й сертифікат EV SSL, оскільки отримати його; власнику веб-сайту потрібно пройти значний процес перевірки. Цей тип сертифіката також відображає інформацію про власника веб-сайту в адресному рядку, щоб відрізнити його від шкідливих сайтів. Сертифікати OV SSL, як правило, є другим за

ціною (після EV SSL), і їх основна мета — шифрувати конфіденційну інформацію користувача під час транзакцій. Комерційні або загальнодоступні веб-сайти повинні встановити сертифікат OV SSL, щоб гарантувати, що будь-яка інформація про клієнтів залишається конфіденційною.

Сертифікати, перевірені доменом (DV SSL) - процес перевірки для отримання цього типу сертифіката SSL мінімальний, і, як результат, сертифікати SSL перевірки домену забезпечують нижчу впевненість і мінімальне шифрування. Вони, як правило, використовуються для блогів або інформаційних веб-сайтів, тобто, які не передбачають збору даних чи онлайн-платежів. Цей тип сертифіката SSL є одним з найдешевших і найшвидших для отримання. Процес перевірки вимагає лише від власників веб-сайтів підтвердити право власності на домен, відповівши на електронний лист або телефонний дзвінок. Адресний рядок браузера відображає лише HTTPS і навісний замок без назви компанії.

Сертифікати SSL підстановки (Wildcard SSL) - дозволяє захистити базовий домен і необмежену кількість субдоменів на одному сертифікаті. Якщо у вас є кілька субдоменів для захисту, тоді покупка сертифіката Wildcard SSL набагато дешевша, ніж покупка окремих сертифікатів SSL для кожного з них. Підстановочні сертифікати SSL мають зірочку * як частину загального імені, де зірочка позначає будь-які дійсні субдомени, які мають той самий базовий домен.

Багатодоменний сертифікат SSL (MDC) - можна використовувати для захисту багатьох доменів та/або субдоменних імен. Це включає комбінацію повністю унікальних доменів і субдоменів з різними TLD (доменами верхнього рівня), за винятком локальних/внутрішніх. Багатодоменні сертифікати не підтримують субдомени за замовчуванням. Якщо вам потрібно захистити як www.example.com, так і example.com за допомогою одного багатодоменного сертифіката, тоді під час отримання сертифіката слід вказати обидва імена хостів.

Сертифікат уніфікованих комунікацій (UCC) - вважаються багатодоменними сертифікатами SSL. Спочатку UCC були розроблені для захисту серверів Microsoft Exchange і Live Communications. Сьогодні будь-який власник веб-сайту може використовувати ці сертифікати, щоб забезпечити захист кількох

доменних імен на одному сертифікаті. Сертифікати UCC перевіряються організаційно і відображають навісний замок у браузері. UCC можна використовувати як сертифікати EV SSL, щоб дати відвідувачам веб-сайту найвищу впевненість за допомогою зеленого адресного рядка.

2.2 Безпечне зберігання ключів в мобільних додатках

Для безпечного зберігання ключів в операційній системі IOS потрібно використовувати IOS Keychain Services. Keychain Services API допомагає зберігати секрети, надаючи програмі механізм для зберігання невеликих біт даних користувача в зашифрованій базі даних, яка називається Keychain. Keychain не обмежується паролями, можна зберігати інші секрети, які явно цікавлять користувача, наприклад дані кредитної картки або навіть короткі нотатки. Також можна зберігати елементи, які потрібні користувачеві, але про які вони можуть не знати. Наприклад, криптографічні ключі та сертифікати, які керуються за допомогою служб сертифікатів, ключів довіри, дозволяють користувачеві брати участь у безпечному зв’язку та встановлювати довіру з іншими користувачами та пристроями [29].

В Andoid потрібно використовувати Android keystore system. Система Android Keystore дозволяє зберігати криптографічні ключі в контейнері, щоб ускладнити вилучення з пристрою. Після того, як ключі знаходяться в сховищі ключів, їх можна використовувати для криптографічних операцій, при цьому матеріал ключа залишається неекспортованим. Крім того, він пропонує можливості для обмеження того, коли і як можна використовувати ключі, наприклад, вимога аутентифікації користувача для використання ключа або обмеження використання ключів лише в певних криптографічних режимах.

Система Android Keystore захищає ключовий матеріал від несанкціонованого використання. По-перше, Android Keystore пом’якшує несанкціоноване використання ключового матеріалу за межами пристрою Android, запобігаючи вилучення ключового матеріалу з процесів програми та з пристрою Android в цілому. По-друге, Android Keystore пом’якшує несанкціоноване використання

ключового матеріалу на пристрої Android, змушуючи програми вказувати авторизоване використання своїх ключів, а потім застосовуючи ці обмеження поза процесами програм.

Ключовий матеріал ключів Android Keystore захищений від вилучення за допомогою двох заходів безпеки:

- Ключовий матеріал ніколи не потрапляє в процес застосування. Коли програма виконує криптографічні операції з використанням ключа Android Keystore, закулісний відкритий текст, зашифрований текст і повідомлення, які підлягають підпису або перевірці, надходять до системного процесу, який виконує криптографічні операції. Якщо процес програми скомпрометований, зловмисник може використовувати ключі програми, але не може витягти їх ключовий матеріал.
- Ключовий матеріал може бути прив’язаний до безпечного обладнання пристрою Android. Коли ця функція ввімкнена для ключа, його матеріал ніколи не відкривається за межами безпечного обладнання. Якщо ОС Android зламано або зловмисник може прочитати внутрішню пам’ять пристрою, зловмисник може використовувати ключі будь-якої програми Android Keystore на пристрої Android, але не витягувати їх із пристрою. Ця функція вмикається лише в тому випадку, якщо захищене обладнання пристрою підтримує певну комбінацію алгоритму ключа, режимів блокування, схем заповнення, з якими дозволено використовувати ключ.

Підтримувані пристрої з ОС Android 9 (API рівня 28) або вище можуть мати Strongbox Keymaster , реалізацію Keymaster, який міститься в апаратному модулі безпеки, як безпечний елемент.

Модуль містить наступне:

- власний процесор;
- безпечне зберігання;
- справжній генератор випадкових чисел;
- додаткові механізми для протидії підробці пакетів і несанкціонованому побічному завантаженню програм;
- безпечний таймер.

Для підтримки малопотужних реалізацій Strongbox підтримується підмножина алгоритмів і розмірів ключів:

- RSA 2048;
- AES 128 і 256;
- ECDSA;
- HMAC;
- APDU розширеної довжини.

Щоб запобігти несанкціонованому використанню ключів на пристрої Android, Android Keystore дозволяє програмам вказувати авторизоване використання своїх ключів під час створення чи імпортування ключів. Після створення або імпорту ключа його авторизації неможливо змінити. Після цього сховище ключів Android застосовує авторизації щоразу, коли використовується ключ.

Підтримувані авторизації на використання ключів поділяються на такі категорії:

- криптографія - алгоритм авторизованого ключа, операції або цілі, схеми заповнення, режими блокування;
- тимчасовий інтервал дії - інтервал часу, протягом якого ключ авторизований для використання;
- автентифікація користувача - ключ можна використовувати, лише якщо користувач був автентифікований досить недавно [30].

2.3 Методи зберігання даних в мобільних додатках

Збережені дані є критичним компонентом інформаційних систем. Не маючи можливості зберігати дані, користувачі повинні були б ввести всі необхідні дані в систему, перш ніж будь-яку обробку можна було б виконати або отримати корисний результат. Інформаційні системи, які спираються на файли даних, бази даних, сховища даних і великі дані, не могли б функціонувати, якби вони не могли зберігати дані як частина системи. У певному сенсі збережені дані - це клей, який утримує інші компоненти системи разом.

Особливі проблеми створює компонент збережених даних мобільних додатків. Пристрої, на яких працюють ці програми, зазвичай мають обмежений обсяг пам’яті. Якщо програма розроблена для використання лише в автономному режимі, вона повинна зберігати всі необхідні дані на пристрої, що обмежує кількість даних, які можна зберігати. Крім того, дані не оновлюються динамічно із зовнішніх джерел у міру зміни умов, а отже, за винятком тих випадків, коли всі дані генеруються на пристрої, вони, швидше за все, застаріли. З іншого боку, якщо програма розроблена для використання в Інтернеті, вона може отримати доступ до необхідних даних у режимі реального часу із сервера, і в цьому випадку обмеження зберігання даних на пристрої не впливає на програму. Однак додаток залежить від бездротового з’єднання з достатньою пропускною здатністю, яка не завжди доступна [31].

Різні підходи до зберігання даних у мобільних додатках можуть мати серйозний вплив на користувача. За допомогою деяких програм користувач оновлює збережені дані та зберігає їх на пристрої. У багатьох ситуаціях дані оновлюються зовнішніми об’єктами, що вимагає доступу до сервера, який, як пояснювалося вище, має потенційні проблеми. Іноді навіть локально збережені дані можуть знадобитися для оновлення даних на сервері, щоб користувач міг поділитися даними між кількома персональними пристроями.

Оскільки додатки для мобільних пристроїв стають все більш поширеними як для особистого використання, так і для бізнесу, дизайн компонента збережених даних стає все більш важливим. Користувачі та компанії не прийматимуть програми, які не відповідають їхнім потребам у даних. Лише добре розроблені програми, які надають потрібні дані в потрібний час, знайдуть прихильність серед користувачів і компаній.

Мобільні програми бувають трьох різновидів щодо компонента збережених даних:

- Офлайнові програми - ці програми зберігають усі свої дані на мобільному пристрої. Дані можуть спочатку заповнюватися під час встановлення програми і, можливо, оновлюватися користувачем пристрою, або спочатку заповнюватися та оновлюватися під час використання програми користувачем

пристрою. Ці програми не повинні бути в мережі для функціонування, їх повна функціональність надається в автономному режимі.

- Інтернет-програми - ці програми залежать від доступу до сервера для своїх збережених даних. Хоча деякі дані можуть зберігатися на мобільному пристрої, програма покладається на дані, що зберігаються на сервері, для своєї функціональності. Ці дані можна оновити на сервері, завантаживши введені користувачем дані з мобільного пристрою на сервер. Його також можуть оновлювати зовнішні органи безпосередньо на сервері, а потім завантажуватись із сервера на мобільний пристрій. До цієї категорії належать програми, які використовуються для електронної комерції. Щоб функціонувати, ці програми мають бути в мережі, їх повна функціональність надається лише в режимі онлайн.
- Синхронізовані програми - ці програми зберігають усі свої дані на мобільному пристрої і, таким чином, можуть використовуватися в автономному режимі, але збережені дані можна оновлювати за допомогою даних із сервера, коли пристрій періодично знаходиться в мережі. Крім того, дані на сервері можуть оновлюватися даними з онлайн-пристрою. Ці програми забезпечують повну функціональність в автономному режимі [32].

При прийнятті рішення про те, який підхід до зберігання мобільних даних використовувати в мобільному додатку, необхідно враховувати фактори, які безпосередньо впливають на зручність використання програми користувачем і її придатність для мобільного бізнесу.

При виборі підходу до збережених даних для мобільних додатків можна врахувати ряд факторів. Чотири фактори, перераховані нижче, були центральними для рішення щодо зберігання даних, оскільки вони безпосередньо впливають на користувача.

- Доступність збережених даних - з офлайновими та синхронізованими програмами збережені дані завжди доступні. Доступність збережених даних для онлайн-програм залежить від наявності онлайн-з’єднання. Користувачі, які мають обмежене підключення до Інтернету, виявлять, що збережені дані для онлайн-

програм менш доступні, ніж для автономних/синхронізованих програм. Доступність може вплинути на можливість користувача отримати доступ до потрібних даних.

- Обсяг збережених даних - обсяг даних, які можна зберігати в автономних і синхронізованих програмах, залежить від обсягу пам’яті мобільного пристрою. Для онлайн-додатків обсяг збережених даних не обмежений мобільним пристроєм і може становити стільки, скільки може зберігати сервер. Користувачі програми з дуже великими обсягами збережених даних можуть виявити, що автономні/синхронізовані програми не надають усі дані, доступні для онлайн-програм. Обсяг може вплинути на здатність користувача мати всі необхідні дані.
- Важливість актуальних даних - збережені дані для офлайн- та онлайн-програм завжди актуальні. Вживаність збережених даних для синхронізованих програм залежить від активності бази даних з моменту останньої синхронізації. Користувачі синхронізованих програм можуть виявити, що деякі збережені дані застаріли до наступної синхронізації, що не стосується офлайн- та онлайн-програм. Вживаність може вплинути на можливість користувача мати доступні актуальні дані.
- Швидкість доступу до збережених даних - доступ до збережених даних для автономної або синхронізованої програми може бути настільки швидким, наскільки це може забезпечити технологія зберігання та обробки мобільного пристрою. Доступ до збережених даних для онлайн-програм залежить від швидкості каналу зв’язку та обсягу даних, до яких здійснюється доступ. Користувачі можуть помітити різницю в швидкості доступу до даних під час використання автономних/синхронізованих програм порівняно з онлайн-програмами. Швидкість може вплинути на здатність користувача своєчасно отримувати доступ до даних.

2.4 Методи шифрування даних

Шифрування - це важливий інструмент, який компанії, використовують для захисту даних, встановлення довіри та підтримки відповідності нормативним вимогам. Шифрування використовується майже в кожній цифровій діловій взаємодії.

Програмне забезпечення для шифрування перетворює великі обсяги даних у зашифрований текст за допомогою алгоритмів. Розшифрувати зашифровані дані можуть лише ті люди або системи, у яких є ключ дешифрування [33].

Існує два основних типи шифрування:

- Симетричне шифрування - схема шифрування з симетричним ключем використовує один симетричний ключ як для шифрування, так і для дешифрування даних. Ключ потрібно надати всім уповноваженим особам. Хоча симетричне шифрування набагато швидше та менш ресурсомістке, ніж асиметричне шифрування, воно також менш безпечне. Симетричне шифрування корисно, коли швидкість обробки важлива, або якщо вам не потрібно ділитися даними з іншою стороною.
- Асиметричне шифрування - цей метод шифрування також називається криптографією з відкритим ключем. Цей метод шифрування використовує два окремих ключа. Один ключ стає відкритим, а один ключ залишається приватним. Відкритий ключ використовується для шифрування даних, а секретний ключ потрібен для їх розшифрування. Асиметричне шифрування забезпечує більшу безпеку, ніж симетричне шифрування, але для деяких цілей воно може бути зайвим, а його процеси можуть сповільнювати транзакції.

Методи шифрування даних:

- Advanced Encryption Standard (AES) - це симетричний алгоритм шифрування, який шифрує фіксовані блоки даних за раз. Ключі, які використовуються для розшифровки тексту, можуть мати довжину 128, 192 або 256 біт. 256-бітовий ключ шифрує дані за 14 раундів, 192-бітний ключ за 12 раундів, а 128-бітний ключ за 10 раундів. Стандарти шифрування AES є найбільш часто використовуваними методами шифрування сьогодні, як для статичних даних, так і для даних, що передаються.
- Triple DES - це симетричне шифрування та розширена форма методу DES, який шифрує блоки даних за допомогою 56-бітового ключа. Triple DES тричі застосовує алгоритм шифрування DES до кожного блоку даних. Потрійний DES зазвичай використовується для шифрування PIN-коду банкомата та паролів UNIX.

- Rivest-Shamir-Adleman (RSA) - асиметричний алгоритм шифрування, який базується на розкладанні на множники добутку двох великих простих чисел. Тільки той, хто знає ці числа, зможе успішно розшифрувати повідомлення. RSA часто використовується під час передачі даних між двома окремими кінцевими точками, але працює повільно, коли великі обсяги даних потрібно зашифрувати.
- Twofish — це метод шифрування, який шифрує блоки даних по 128 біт. Він вважається наступником 64-бітного методу шифрування Blowfish і більш універсальним, ніж його спеціалізований наступник Threefish. Twofish завжди шифрує дані за 16 раундів, незалежно від розміру ключа. Хоча він працює повільніше, ніж AES, метод шифрування Twofish продовжує використовуватися деякими програмними рішеннями для шифрування файлів і папок [34].

Висновки за розділом 2

Безпека мобільних додатків - це захід для захисту програм від зовнішніх загроз, таких як шкідливе програмне забезпечення та інші цифрові шахрайства, які ризикують отримати критичну особисту та фінансову інформацію від хакерів.

Безпека мобільних додатків стала не менш важливою в сучасному світі. Порушення безпеки мобільних пристроїв може не тільки надати хакерам доступ до особистого життя користувача в режимі реального часу, але й розкрити такі дані, як їхнє поточне місцезнаходження, банківську інформацію, особисту інформацію та багато іншого.

При передачі чи зберігання даних у зашифрованому вигляді, які неможливо переглянути, не зіставивши їх з секретним ключем, гарантує, що дані з додатку, навіть якщо потраплять до зловмисника, то не принесуть йому ніякої користі.

РОЗДІЛ 3
РОЗРОБКА МЕХАНІЗМУ БЕЗПЕЧНОГО ЗБЕРІГАННЯ ДАНИХ В МОБІЛЬНОМУ ДОДАТКУ

3.1 Створення бази даних й REST API

Для створення механізму безпечного зберігання інформації нам потрібно створити базу даних, в якій ми будемо зберігати дані про користувачів. Ми будемо використовувати docker для створення бази даних.

Docker, це програмна платформа для створення, запуску та керування контейнерами на серверах і в хмарі.

Раніше було так, що коли ви хотіли запустити веб-додаток, ви купували сервер, встановлювали Linux , налаштовували стек LAMP і запускали програму. Якщо ваша програма стала популярною, ви вміли добре балансувати навантаження , налаштувавши другий сервер, щоб програма не виходила з ладу через занадто великий трафік [35].

Однак часи змінилися, і замість того, щоб зосередитися на окремих серверах, Інтернет побудований на масивах взаємозалежних і резервних серверів у системі, яку зазвичай називають «хмарою». Завдяки інноваціям, таким як простори імен ядра Linux і cgroups , концепція сервера могла бути вилучена з обмежень апаратного забезпечення і замість цього стала, по суті, частиною програмного забезпечення. Ці програмні сервери називаються контейнерами , і вони являють собою гібридну суміш ОС Linux, на якій вони працюють, плюс гіперлокалізоване середовище виконання [36].

Ми будемо використовувати базу даних PostgreSQL. PostgreSQL – це потужна система об’єктно-реляційних баз даних з відкритим вихідним кодом, яка використовує та розширює мову SQL у поєднанні з багатьма функціями, які безпечно зберігають і масштабують найскладніші робочі навантаження даних.

PostgreSQL постачається з багатьма функціями , які допомагають розробникам створювати програми, адміністраторам захищати цілісність даних і створювати

відмовостійкі середовища, а також допомагають вам керувати даними незалежно від того, наскільки великий чи малий набір даних. Окрім того , що PostgreSQL є безкоштовним і відкритим вихідним кодом, він дуже розширюваний.

PostgreSQL намагається відповідати стандарту SQL, якщо така відповідність не суперечить традиційним функціям або може призвести до неправильних архітектурних рішень. Багато функцій, необхідних стандартом SQL, підтримуються, хоча іноді вони мають дещо відмінний синтаксис.

Визначивши технології які ми будемо використовувати для створення інфраструктури, перейдемо до реалізації. Створимо файл docker-compose.yml, в якому опишемо налаштування для docker і бази даних.

docker-compose.yml

```yaml
version: "3.8"

services:
  database:
    image: postgres
    environment:
      - POSTGRES_DB=${POSTGRES_DATABASE_NAME}
      - POSTGRES_USER=${POSTGRES_DATABASE_USER}
      - POSTGRES_PASSWORD=${POSTGRES_DATABASE_PASSWORD}
    volumes:
      - ./database:/var/lib/postgresql/data/
    ports:
      - ${POSTGRES_PORT}:5432

```

Рисунок 3.1 – Файл конфігурацій docker-compose.yml

Спочатку ми обрали версію docker, яку ми будемо використовувати. Після цього ми створюємо сервіс, який має назву “database”, для цього сервісу ми обираємо image, який буде завантажений із DockerHub. Ми обрали image “postgres”, так як будемо використовувати базу даних PostgreSQL. Для налаштування цього image ми додаємо змінні POSTGRES_DB, POSTGRES_USER та POSTGRES_PASSWORD, ці змінні потрібні для налаштування бази даних. Після

цього ми налаштовуємо volumes, тобто ми зіставляємо нашу систему з контейнером, який буде створено. І в кінці ми також зіставляємо порти, які будуть доступні (рис 3.1).

Тепер ми можемо створити базу даних запустивши команду docker-compose up в поточній директорії (рис 3.2).

Рисунок 3.2 – Результат команди docker-compose up

Після успішного створення бази даних ми переходимо до створення REST Server. Для програмування ми будимо використовувати мову JavaScript і платформу NodeJS для запуску нашого коду.

Спочатку запустимо команду npm init, яка створить нам файл конфігурацій package.json. Після цього ми створюємо файл index.ts, з цього файлу буде запускатися наша програма. Також в цьому файлі ми створимо дві кінцеві точки, для реєстрації й входу. Паролі користувачів будуть зберігатися в зашифрованому вигляді, для шифрування будемо використовувати бібліотеку bcrypt (рис 3.3).

index.ts > ...

```ts
const express = require("express");
const { AppDataSource } = require("./src/data-source");
const bcrypt = require("bcrypt");
const { User } = require("./src/entity/User");

export const app = express();
const port = 8080;
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.post("/signin", async (req, res) => {
  try {
    const userRepository = AppDataSource.getRepository(User);

    const user = await userRepository.findOneBy({
      login: req.body.login,
    });
    if (user) {
      const { password, ...data } = user;
      const isCompare = await bcrypt.compare(req.body.password, password);
      if (isCompare) {
        return res.send(data);
      } else {
        return res.status(422).send("Password not valid");
      }
    } else {
      return res.status(404).send("Not found");
    }
  } catch {
    return res.status(500).send("unknown error");
  }
});
app.post("/signup", async (req, res) => {
  try {
    if (req.body.login.length < 4 || req.body.password.length < 8) {
      return res.status(422).send("not valid params");
    }
    const hash = await bcrypt.hash(req.body.password, 10);
    const user = new User();

    user.login = req.body.login;
    user.password = hash;
    user.hash = await bcrypt.genSalt(10);

    const userRepository = AppDataSource.getRepository(User);
    await userRepository.save(user);

    return res.send("Created!");
  } catch {
    return res.status(500).send("unknown error");
  }
});
app.listen(port, () => {
  console.log(`Example app listening on port ${port}`);
});
AppDataSource.initialize();
```

Рисунок 3.3 – Файл index.ts

Я обрав СУБД typeorm, тому що це одна з найпопулярніших СУБД на мові JavaScript. Створимо файл src/data-source.ts, в якому ми налаштуємо підключення СУБД до бази даних (рис 3.4).

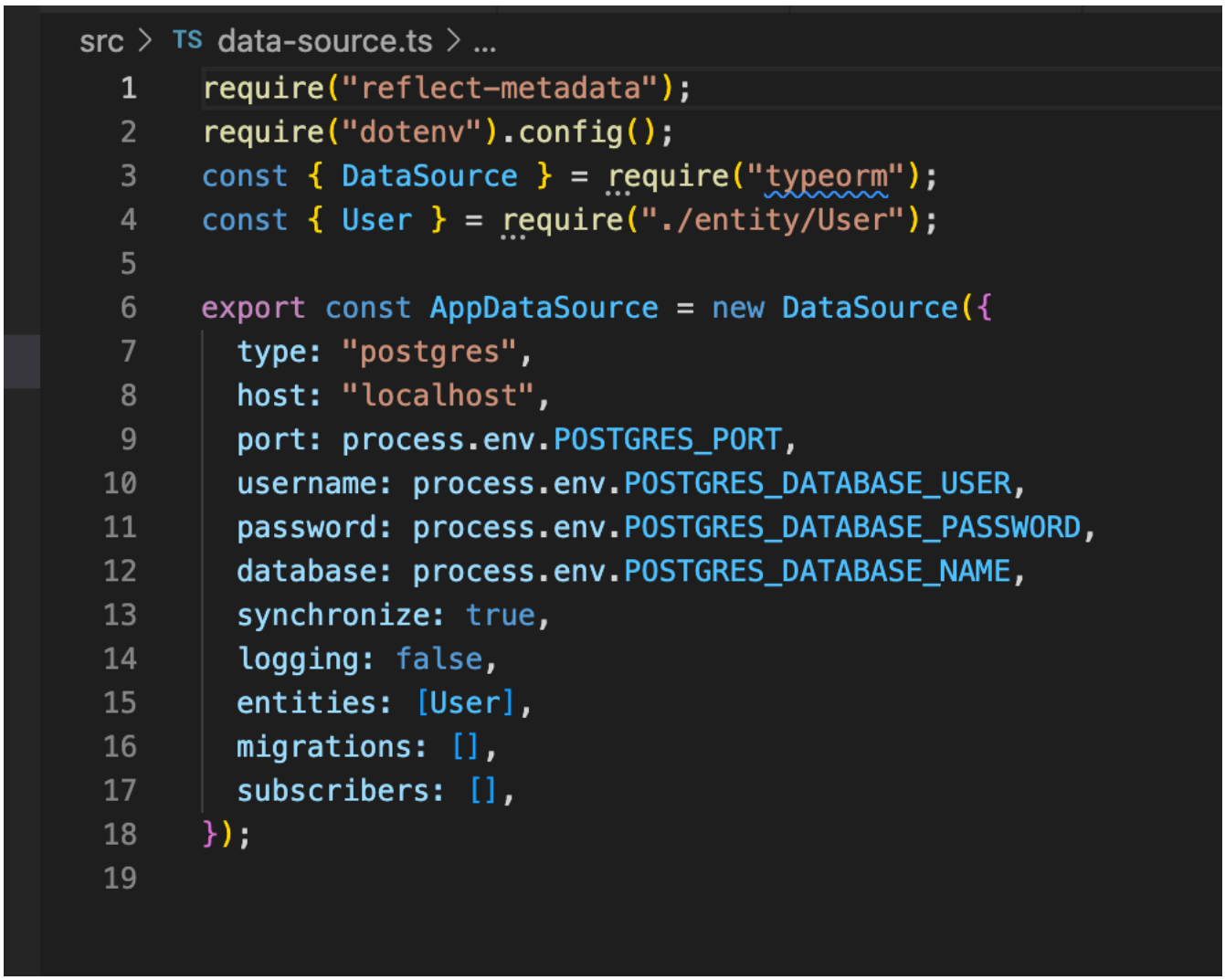

Рисунок 3.4 – Файл data-source.ts

Підключивши СУБД до бази даних, потрібно створити таблицю для зберігання даних про користувачів, для цього створимо файл User.ts й опишемо поля які нам потрібні (рис 3.5).

src > entity > TS User.ts > User

```
1   const { Entity, PrimaryGeneratedColumn, Column } = require("typeorm");
2
3   @Entity()
4   export class User {
5     @PrimaryGeneratedColumn()
6     id: number;
7
8     @Column({ unique: true })
9     login: string;
10
11    @Column()
12    password: string;
13
14    @Column()
15    hash: string;
16  }
17
```

Рисунок 3.5 – Файл User.ts

Щоб запустити REST server запустимо в терміналі команду ts-node ./index.ts, або npm run start (рис 3.6).

```
khomynandrii@MacBook-Pro api % npm run start

> project@0.0.1 start
> ts-node ./index.ts

Example app listening on port 8080
```

Рисунок 3.6 – Запуск REST server

3.2 Створення мобільного додатку з безпечним зберіганням даних

Для створення мобільного додатку ми будемо використовувати React Native. React Native — це фреймворк JavaScript для написання мобільних додатків для iOS та Android. Він оснований на React, бібліотеці JavaScript від Facebook для створення інтерфейсів користувача, але замість того, щоб націлюватися на браузер, він націлений на мобільні платформи. Крім того, оскільки більшість коду, який ви пишете, можна спільно використовувати між платформами, React Native полегшує одночасну розробку як для Android, так і для iOS.

Додаток буде складатися з двох екранів: перший – це екран аутентифікації та реєстрації, а другий – це екран на якому ми можемо завантажити дані й після завантаження наглядно бачити в якому вигляді вони у нас зберігаються. Дані будуть шифруватися за допомогою шифру AES.

Механізм безпечного зберігання даних в мобільному додатку буде працювати так:

- Спочатку користувач повинен зареєструватися. Після реєстрації в базу даних буде записаний токен, за допомогою якого ми будемо шифрувати дані в додатку.
- Потім потрібно увійти за допомогою логіну і паролю, після цього на мольному додатку ми отримаємо токен, який будемо використовувати для шифрування даних, цей токен буде зберігатися в SecureStore.
- На іншому екрані, ми зможемо завантажити дані, вони будуть зашифровані й збережені в звичайний AsyncStorage.

Таким чином всі дані в мобільному додатку будуть безпечно зберігатися й навіть якщо зловмисник отримає дані, то не зможе їх ніяк використати, тому що вони будуть в зашифрованому вигляді. А токен для розшифровки буде надійно зберігатися в SecureStore.

Перед початком розробки розробимо схему роботи мобільного додатка (рис 3.7).

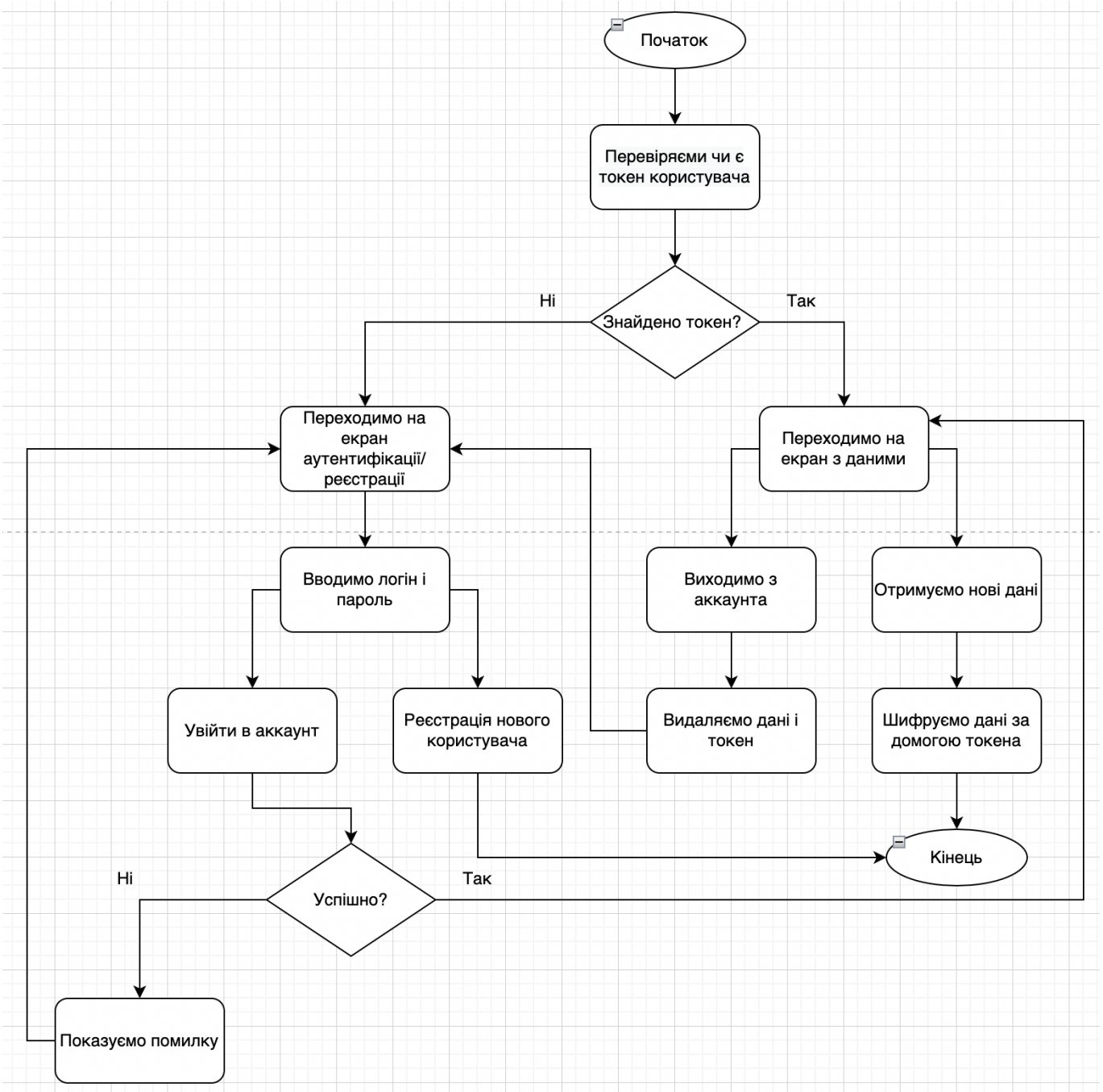

Рисунок 3.7 – Схема роботи мобільного додатку

Створимо екран реєстрації/аутентифікації. На цьому екрані повинні бути два поля для вводу логіну і пароля, дві кнопки “Увійти” та “Зареєструватися” (рис 3.8, 3.9).

```
<SafeAreaView
  style={{
    flex: 1,
    justifyContent: "center",
    alignItems: "center",
    backgroundColor: "rgba(242, 255, 252, 0.8)",
  }}
>
  <Text style={{ fontSize: 24, marginBottom: 50 }}>Додаток</Text>
  <TextInput
    value={login}
    onChangeText={setLogin}
    style={styles.input}
    placeholder="логін"
    placeholderTextColor="rgba(0, 171, 11, 0.8)"
    autoCapitalize="none"
  />
  <TextInput
    value={password}
    onChangeText={setPassword}
    style={styles.input}
    placeholder="пароль"
    placeholderTextColor="rgba(0, 171, 11, 0.8)"
    autoCapitalize="none"
  />
  <Button title="Увійти" onPress={loginFun} />
  <Button title="Зареєструватися" onPress={signUpFun} />
</SafeAreaView>
```

Рисунок 3.8 – Частина коду для відображення компонентів на екрані реєстрації/аутентифікації

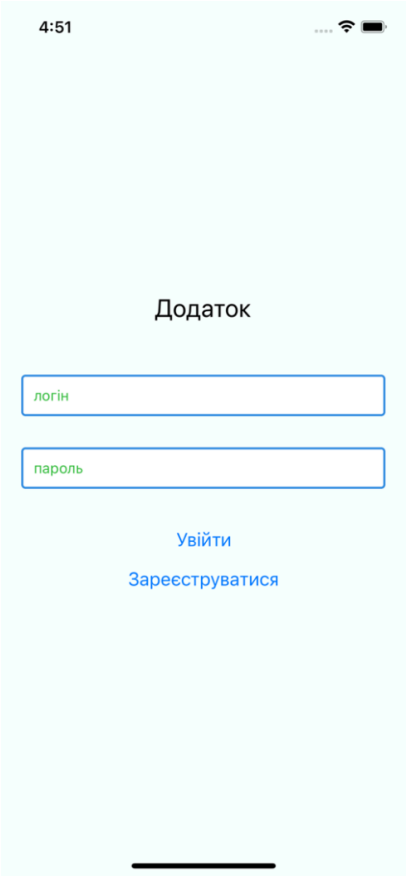

Рисунок 3.9 – Екран реєстрації/аутентифікації

При натисканні на кнопку “Увійти” нам потрібно зробити HTTP запит для аутентифікації й якщо запит буде успішним, то потрібно записати токен який нам прийде в SecureStore і показати повідомлення “ Користувач успішно увійшов”, а якщо буде помилка, то вивести її (рис 3.10, 3.11).

```
const loginFun = async () => {
  try {
    const res = await signIn({ login, password });

    await SecureStore.setItemAsync("hash", res.data.hash);

    Toast.show({
      type: "success",
      text1: "Користувач успішно увійшов",
    });
    await checkStore();
  } catch (e: any) {
    Toast.show({
      type: "error",
      text1: e.message,
    });
  }
};
```

Рисунок 3.10 – Функція яка буде викликатися при натисканні на кнопку “Увійти”

Рисунок 3.11 – Успішний вхід користувача

При натисканні на кнопку “Зареєструватися” нам потрібно зробити HTTP запит для реєстрації й якщо запит буде успішним, то показати повідомлення “ Користувач успішно створений”, а якщо буде помилка, то вивести її (рис 3.12, 3.13).

```
const signUpFun = async () => {
  try {
    await signUp({ login, password });

    Toast.show({
      type: "success",
      text1: "Користувач успішно створений",
    });
  } catch (e: any) {
    Toast.show({
      type: "error",
      text1: e.message,
    });
  }
};
```

Рисунок 3.12 – Функція яка буде викликатися при натисканні на кнопку “Зареєструватися”

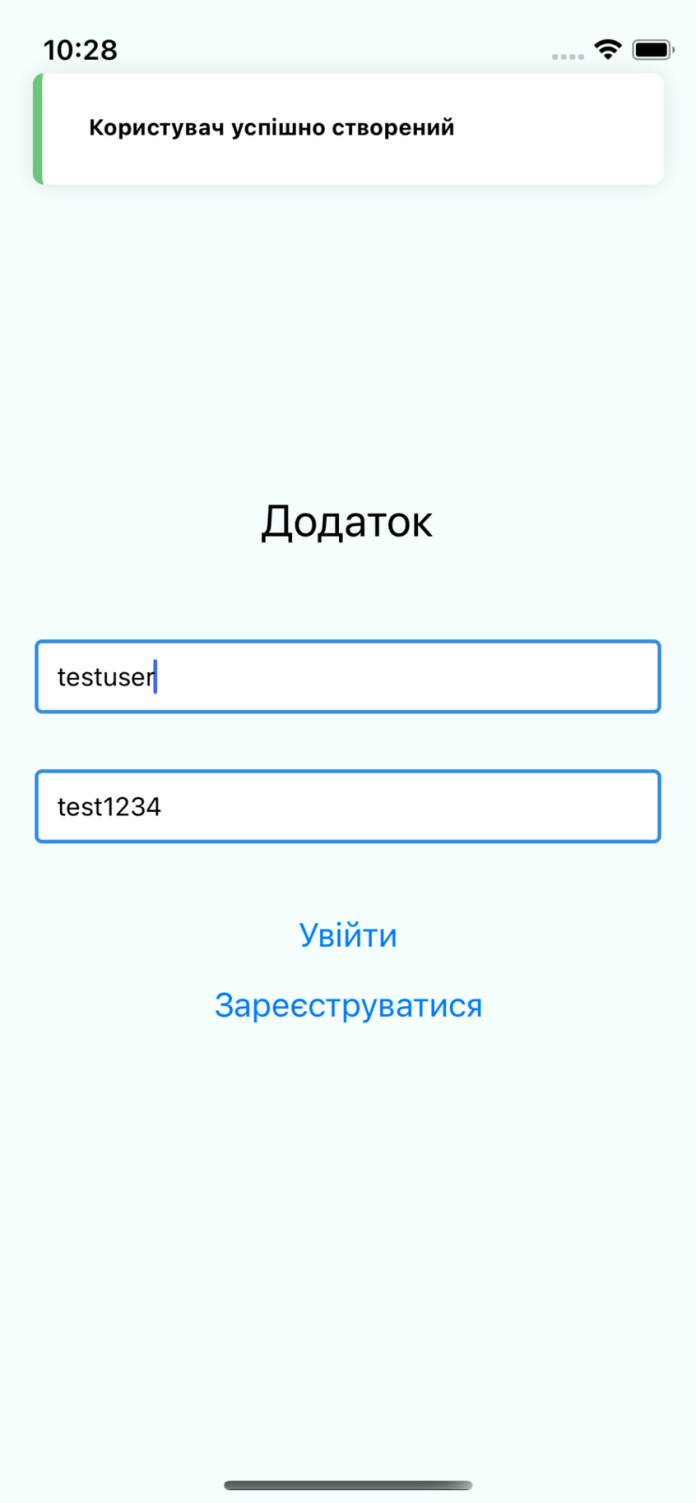

Рисунок 3.13 – Реєстрація нового користувача

Далі створимо екран з даними який буде відображатися лише для аутентифікованих користувачів. На цьому екрані повинні бути дві кнопки “Отримати дані” та “Вийти”, та список з даними, який розділений на дві частини,

зліва декодовані дані, а зправа дані у вигляді в якому вони збережені в мобільному додатку. Також зверху буде показано токен для декодування даних (рис 3.14, 3.15).

```
<SafeAreaView
  style={{
    flex: 1,
    justifyContent: "center",
    alignItems: "center",
    backgroundColor: "rgba(242, 255, 252, 0.8)",
  }}
>
  <View
    style={{
      flexDirection: "row",
      paddingHorizontal: 10,
      marginBottom: 10,
    }}
  >
    <View style={{ flex: 1 }}>
      <Text style={{ fontSize: 13 }}>
        {"Декодовані дані за допомогою ключа: " + key}
      </Text>
    </View>
    <View style={{ flex: 1 }}>
      <Text style={{ fontSize: 13, paddingLeft: 20 }}>
        В такому вигляді дані зберігаються
      </Text>
    </View>
  </View>
  <ScrollView style={{ width: "100%", height: "100%" }}>
    {data.map((item, index) => (
      <View key={index}>
        <Text
          style={{ textAlign: "center", color: "red", marginBottom: 5 }}
        >
          {index + 1}
        </Text>

        <View
          style={{
            flexDirection: "row",
            paddingHorizontal: 10,
            marginBottom: 10,
          }}
        >
          <View style={{ flex: 1, paddingRight: 15 }}>
            <Text>{decodeData(item)}</Text>
          </View>
          <View style={{ flex: 1 }}>
            <Text>{item}</Text>
          </View>
        </View>
      </View>
    ))}
  </ScrollView>
  <Button title="Отпримати дані" onPress={getDataFun} />

  <Button title="Вийти" onPress={logoutFun} />
</SafeAreaView>
```

Рисунок 3.14 – Частина коду для відображення компонентів на екрані з даними

Рисунок 3.15 – Екран з даними, який відображається після аутентифікації

При натисканні на кнопку “Отримати дані” ми повинні викликати функцію, яка зробить HTTP запит за даними. В цій самій функції потрібно закодувати дані за допомогою токену й зберегти їх в AsyncStorage (рис 3.16, 3.17).

```
const getDataFun = async () => {
  const res = await getData(data.length + 1);

  const parseData = await getDataFromStorage();

  const encodeData = crypto.AES.encrypt(res.data.title, key).toString();

  parseData.push(encodeData);

  await AsyncStorage.setItem("data", JSON.stringify(parseData));
  getDataFromStorage().then((data) => setData(data));
};
```

Рисунок 3.16 – Функція для отримання даних

Рисунок 3.17 – Екран з даними після кількох натискань на кнопку “Отримати дані”

Також потрібно створити функцію яка буде викликатися при натисканні на кнопку “Вийти”. Ця функція повинна очищати AsyncStorage з даними та видаляти токен з SecureStore. При виникненні помилки потрібно відображати її для користувача (рис 3.18).

```
const logoutFun = async () => {
  try {
    await AsyncStorage.clear();
    await SecureStore.deleteItemAsync("hash");
    await checkStore();
  } catch {
    Toast.show({
      type: "error",
      text1: "Виникла помилка",
    });
  }
};
```

Рисунок 3.18 – Функція яка буде викликатися при натисканні на кнопку “Вийти”

Висновки за розділом 3

В цьому розділі спочатку було створено інфраструктуру, REST server та механізм для безпечного зберігання даних в мобільному додатку.

Якщо використовувати запропонований механізм захисту, то всі дані в мобільному додатку будуть безпечно зберігатися й навіть якщо зловмисник отримає дані, то не зможе їх ніяк використати, тому що вони будуть в зашифрованому вигляді, а токен для розшифровки буде надійно зберігатися в SecureStore.

ВИСНОВКИ

В ході дипломної роботи були проаналізовані середовища клієнт-сервер архітектури, які користуються популярністю, оскільки вони підвищують ефективність обробки додатків, одночасно знижуючи витрати та отримують максимальну вигоду від спільної роботи всіх ресурсів. Ці переваги досягаються шляхом поділу обробки між клієнтською машиною/програмним забезпеченням і серверною машиною/програмним забезпеченням. Кожен процес працює незалежно, але у співпраці та сумісності з іншими машинами та програмами.

Існує три компоненти середовища клієнт-сервер: клієнт, сервер і мережа. Мережа поєднує фізичне та функціональне розділення між клієнтом і сервером.

Проте саме ті характеристики, які роблять клієнт-сервер популярними, також роблять його найбільш вразливим до порушень безпеки. Саме розподіл послуг між клієнтом і сервером відкриває їх для пошкодження, шахрайства та неправильного використання. Розгляд безпеки повинен включати сервери, мережі та клієнтські пристрої.

Також було досліджено безпеку мобільних додатків від зовнішніх загроз, таких як шкідливе програмне забезпечення та інші цифрові шахрайства, які ризикують отримати критичну особисту та фінансову інформацію від хакерів.

Безпека мобільних додатків стала не менш важливою в сучасному світі. Порушення безпеки мобільних пристроїв може не тільки надати хакерам доступ до особистого життя користувача в режимі реального часу, але й розкрити такі дані, як їхнє поточне місцезнаходження, банківську інформацію, особисту інформацію та багато іншого.

При передачі чи зберігання даних у зашифрованому вигляді, які неможливо переглянути, не зіставивши їх з секретним ключем, гарантує, що дані з додатку, навіть якщо потраплять до зловмисника, то не принесуть йому ніякої користі.

І в підсумку було створено інфраструктуру, REST server та механізм для безпечного зберігання даних в мобільному додатку.

Якщо використовувати запропонований механізм захисту, то всі дані в мобільному додатку будуть безпечно зберігатися й навіть якщо зловмисник отримає дані, то не зможе їх ніяк використати, тому що вони будуть в зашифрованому вигляді, а токен для розшифровки буде надійно зберігатися в SecureStore.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ділова етика [Електронний ресурс]. – Режим доступу: https://stud.com.ua/19298/etika_ta_estetika/konfidentsiynist_privatnist_loyalnist_chesnist_transparentnist_sumlinnist.

2. Нове в законодавстві про інформацію [Електронний ресурс]. – Режим доступу: https://minjust.gov.ua/m/str_35738

3. Конфіденційна інформація та комерційна таємниця [Електронний ресурс]. – Режим доступу: https://www.victorija.ua/dovidnik/konfidentsiyna-informatsiya-ta-komertsiyna-tayemnytsya.html?print=print

4. Політика конфіденційності [Електронний ресурс]. – Режим доступу: https://zakononline.com.ua/confidentiality

5. Як користувачі соціальних мереж стають жертвами вірусів, спамів чи хакерських атак [Електронний ресурс]. – Режим доступу: https://pogliad.ua/news/chernivtsi/yak-koristuvachi-socialnih-merezh-stayut-zhertvami-virusiv-spamiv-chi-hakerskih-atak-228172

6. Шкідливий програмний засіб [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Malware

7. Неправильні конфігурації безпеки та їх наслідки для веб-безпеки [Електронний ресурс]. – Режим доступу: https://corewin.ua/blog/security_misconfigurations/

8. Що таке активна атака? [Електронний ресурс]. – Режим доступу: https://uk.theastrologypage.com/active-attack

9. Що таке пасивна атака? [Електронний ресурс]. – Режим доступу: https://uk.theastrologypage.com/passive-attack

10. Загальний регламент про захист даних [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/General_Data_Protection_Regulation

11. Закон України “Про інформацію” [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2657-12#Text

12. Інтернет-безпека телефону[Електронний ресурс]. – Режим доступу: https://eset.ua/ru/blog/view/33/internet-bezopasnost-telefona-kolichestvo-vredonosnykh-programm-dlya-mobilnykh-devaysov-vozroslo

13. Що таке мобільна безпека? [Електронний ресурс]. – Режим доступу: https://uk.theastrologypage.com/mobile-security

14. Безпека і кібербезпека смартфонів [Електронний ресурс]. – Режим доступу: https://datami.ua/bezpeka-i-kiberbezpeka-smartfoniv/

15. Безпека мобільних пристроїв [Електронний ресурс]. – Режим доступу: https://stud.com.ua/20639/informatika/bezpeka_mobilnih_pristroyiv

16. Кращі поради для захисту Вашого смартфону [Електронний ресурс]. – Режим доступу: https://cybercalm.org/novyny/krashhi-porady-dlya-zahystu-vashogo-smartfonu-ta-personalnyh-danyh-na-2020-rik/

17. Десять правил безпеки мобільних пристроїв [Електронний ресурс]. – Режим доступу: https://blog.allo.ua/ua/desyat-pravil-bezpeki-mobilnih-pristroyiv_2020-01-39/

18. Як захистити свій мобільний телефон [Електронний ресурс]. – Режим доступу: https://worldvision.com.ua/kak-zashchitit-svoy-mobilnyy-telefon/

19. 7 способів захисту сервера [Електронний ресурс]. – Режим доступу: https://habr.com/ru/company/galtsystems/blog/314344/

20. Захист сервера від злому [Електронний ресурс]. – Режим доступу: https://faq.in.ua/articles/20-zakhyst-servera-vid-zlomu.html

21. Методи захисту мережевих серверів від несанкціонованого доступу [Електронний ресурс]. – Режим доступу: http://elartu.tntu.edu.ua/bitstream/123456789/16555/2/VseukrStud_2016v1_Kovalenko_V-Methods_of_protecting_network_69.pdf

22. Технології захисту інформації [Електронний ресурс]. – Режим доступу: https://www.uzhnu.edu.ua/uk/infocentre/get/4186

23. Захист даних у базах даних [Електронний ресурс]. – Режим доступу: https://pidru4niki.com/88683/informatika/zahist_danih_bazah_danih

24. Захист баз даних [Електронний ресурс]. – Режим доступу: https://studfile.net/preview/9999308/page:4/

25. Захист БД від несанкціонованого доступу [Електронний ресурс]. – Режим доступу: https://rdb.dp.ua/uk/chapter_11

26. Шифрування бази даних [Електронний ресурс]. – Режим доступу: https://en.wikipedia.org/wiki/Database_encryption

27. Що таке ssl сертифікат і навіщо він потрібен? [Електронний ресурс]. – Режим доступу: https://freehost.com.ua/faq/faq/scho-take-ssl-sertifikat-i-navischo-vin-potriben/

28. Що таке сертифікат SSL? [Електронний ресурс]. – Режим доступу: https://ua.godaddy.com/help/sho-take-sertifikat-ssl-542

29. Keychain Services [Електронний ресурс]. – Режим доступу: https://developer.apple.com/documentation/security/keychain_services

30. Android keystore system [Електронний ресурс]. – Режим доступу: https://developer.android.com/training/articles/keystore?authuser=2

31. Зберігання даних в Android за допомогою Realm [Електронний ресурс]. – Режим доступу: https://stfalcon.com/ru/blog/post/saving-data-in-android-using-realm

32. 10 кращих хмарних сервісів для зберігання і синхронізації даних [Електронний ресурс]. – Режим доступу: https://root-nation.com/ua/articles-ua/services-ua/ua-best-cloud-services-storages-2020/

33. Що таке шифрування та як воно працює? [Електронний ресурс]. – Режим доступу: https://experience.dropbox.com/uk-ua/resources/what-is-encryption

34. Види шифрування інформації [Електронний ресурс]. – Режим доступу: https://ua5.org/protect/395-vidi-shifruvannya-informaciyi.html

35. Що таке Docker і навіщо він? [Електронний ресурс]. – Режим доступу: https://qagroup.com.ua/publications/shcho-take-docker-i-navishcho-vin/

36. Що таке Docker: простими словами про контейнеризацію [Електронний ресурс]. – Режим доступу: https://blog.ithillel.ua/articles/shcho-take-docker-prostimi-slovami-pro-konteynerizatsiyu