МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРИЗОВАНИХ СИСТЕМ ЗАХИСТУ ІНФОРМАЦІЇ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Комп'ютеризованих систем захисту інформації

_____ Михайло СТЕПАНОВ
«____»_____ 2023 р.

На правах рукопису
УДК 004.056.5:510.22(043.3)

КВАЛІФІКАЦІЙНА РОБОТА

ЗДОБУВАЧА ВИЩОЇ ОСВІТИ ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

Тема: Вебзастосунок для захисту паролів користувачів

Виконавець: Олександр ОЛІЙНИК
Керівник: к.т.н., доцент Микола БРАІЛОВСЬКИЙ
Консультант розділу «Охорона навколишнього середовища»: к.т.н., доцент Тетяна ДМИТРУХА
Нормоконтролер: к.т.н., доцент Микола БРАІЛОВСЬКИЙ

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет: Кібербезпеки та програмної інженерії

Кафедра: Комп'ютеризованих систем захисту інформації

Освітній ступінь: Магістр

Спеціальність: 125 «Кібербезпека»

Освітньо-професійна програма: «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Комп'ютеризованих систем захисту інформації

_____Михайло СТЕПАНОВ
«__» _____ 2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

здобувача вищої освіти Охріменка Андрія Олександровича

1. Тема: *Вебзастосунок для захисту паролів користувачів*
затверджена наказом ректора від «15» вересня 2023 р. № 1814/ст.
2. Термін виконання: з 16.10.2023 р. по 31.12.2023 р.
3. Вихідні дані: реалізувати і описати реалізацію програмного модуля реєстрації користувачів у менеджері паролів; проаналізувати та дослідити загальні поняття щодо менеджерів паролів; проаналізувати та дослідити загрози безпеки даних користувачів.
4. Зміст пояснювальної записки: огляд та аналіз програм безпечного зберігання реєстраційних даних, аналіз загроз менеджерів паролів, реалізація і детальний опис реалізації програмного модуля застосунку захисту паролів користувачів.

5. КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

№ з/п	Етапи виконання кваліфікаційної роботи	Термін виконання етапів	Примітка
1.	Уточнення постановки задачі	16.10.2023	*Виконано*
2.	Аналіз літературних джерел	23.10.2023	*Виконано*
3.	Обґрунтування вибору рішення	26.10.2023	*Виконано*
4.	Збір інформації	30.10.2023	*Виконано*
5.	Огляд та аналіз програм безпечного зберігання реєстраційних даних	13.11.2023	*Виконано*
6.	Аналіз загроз менеджерів паролів	20.11.2023	*Виконано*
7.	Реалізація вебзастосунку для захисту паролів користувачів	08.12.23	*Виконано*
8.	Охорона навколишнього середовища	10.12.23	*Виконано*
9.	Перевірка на антиплагіат	14.12.23	*Виконано*
10.	Оформлення і друк пояснювальної записки	15.12.2023	*Виконано*
11.	Оформлення презентації	17.12.2023	*Виконано*
12.	Отримання рецензій від рецензента	22.12.2023	*Виконано*

6. Консультанти з окремих розділів

Розділ	Консультант (посада, П.І.Б.)	Дата, підпис	
		Завдання видав	Завдання прийняв
Охорона навколишнього середовища	Дмитруха Т.І.		

7. Дата видачі завдання: «16» жовтня 2023 р.

Здобувач вищої освіти Олександр ОЛІЙНИК

(підпис, дата)

Керівник кваліфікаційної роботи Михайло БРАІЛОВСЬКИЙ

(підпис, дата)

РЕФЕРАТ

Кваліфікаційна робота на тему: «Вебзастосунок для захисту паролів користувачів» складається зі вступу, основної частини, що містить 4 розділи, 3 висновки до перших трьох розділів, загального висновку та списку використаної літератури. Загальний обсяг роботи – 22 сторінки. Робота містить рисунок та 1 таблицю. Список використаних джерел включає 27 джерел.

Метою кваліфікаційної роботи є реалізація вебзастосунку для захисту паролів користувачів.

У кваліфікаційній роботі розглянуті небезпеки, які загрожують конфіденційним даним користувачів та методи запобігання цьому.

Проведені дослідження базуються на сучасних методах криптографічного захисту даних, хешування зашифрованих даних та побудові сучасних менеджерів паролів.

Реалізація методу криптографічного шифрування RSA та використання хешування зашифрованих даних в менеджері паролів дозволяє надати підвищений рівень захисту даних користувача та доцільність використання в менеджерах паролів.

Запропоновані методи дозволяють забезпечити швидкий та надійний захист особистих даних користувачів.

Ключові слова: МЕНЕДЖЕР ПАРОЛІВ, ШИФРУВАННЯ, ХЕШУВАННЯ, h2, ЗАХИСТ ДАНИХ

ЗМІСТ

Вступ……7
РОЗДІЛ 1. ОГЛЯД ТА АНАЛІЗ ПРОГРАМ БЕЗПЕЧНОГО ЗБЕРІГАННЯ РЕЄСТРАЦІЙНИХ ДАНИХ……...9
1.1 Загальні відомості……..9
1.1.1 Менеджери паролів у формі локального програмного забезпечення…11
1.1.2 Менеджери паролів у формі онлайн-сервісу……11
1.1.3 Апаратні пристрої менеджерів паролів……12
1.1.4 Переваги менеджерів паролів……12
1.2 Аналіз існуючих менеджерів паролів……14
1.2.1 KeePass……14
1.2.1.1 Управління паролями……..……14
1.2.2 1Password……...16
1.2.3 Dashlane……17
1.2.3.1 Функціональні розділи……19
1.2.3.2 Захист і генерація паролів……20
1.3. Висновок до першого розділу……...…20
РОЗДІЛ 2. АНАЛІЗ ЗАГРОЗ МЕНЕДЖЕРІВ ПАРОЛІВ……22
2.1 Вступ……22
2.2 Блокування менеджерів паролів……24
2.3 Загроза SQLInjection……24
2.3.1 Форми SQLInjection……..25
2.3.2 Технічна реалізація……..……..25
2.4 Загроза Fishing……..27
2.5 Загроза MITM-атака……29
2.5.1 Принцип атак……30
2.5.2 Як виявити MITM-атаку……...……32
2.6 Загроза BruteForce……34
2.6.1 Класифікація brute force……34
2.6.2 Захист від брутфорс-атак……..…36

2.7 Загроза Keylogger…….38
2.7.1 Класифікація……..…40
2.7.2 Види доступу до інформації…….……40
2.8 Загроза Cross-SiteRequestForgery……41
2.8.1 Заходи профілактики, які НЕ працюють……43
2.9 Висновок до другого розділу…….……44
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ ДЛЯ ЗАХИСТУ ПАРОЛІВ КОРИСТУВАЧІВ……..46
3.1 Опис розробленого вебзастосунку для захисту паролів користувачів…...46
3.1.1 Мова програмування……46
3.1.2 Фреймворк Spring MVS……..…….…48
3.1.3 Інструмент автоматизації Maven…….51
3.1.4 HTML на основі Thymeleaf…….…….54
3.1.5 Функція EncryptMessage……..56
3.1.6 Функція DecryptMessage……..56
3.1.7 База даних h2……57
3.1.8 Метод шифрування RSA……..58
3.2 Практична частина вебзастосунку для захисту паролів користувачів…...59
3.2.1 Кодування методом RSA…….…59
3.2.2 Декодування методом RSA……..61
3.2.3 Зміна вхідних та вихідних даних…….…62
3.2.4 Вхід та реєстрація у вебзастосунку…….…62
3.2.5 Запис даних у менеджері паролів……66
3.2.6 Блокування профілю користувача…….…66
3.2.7 Порівняння з уже існуючим аналогом……..67
3.3 Висновок до третього розділу……68
4. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА……69
4.1 Антропогенний вплив на грунти……69
Висновок до дипломної роботи……71
Список використаної літератури……..72

ВСТУП

Актуальність та область застосування. Ми всі живемо в епоху електороники і у кожного з нас є особисті телефони та комп’ютери з виходом в Інтернет. Майже кожна людина має що як найменше дві чи більше електроних адреси, одна з яких може бути зв’заною з їх роботою, а інші особисті. В той же час, як правила використання паролів говорять: «Паролі мусять бути складними та довгими», й «Один пароль − один сайт».

Важко запам'ятати декілька великих і дуже складних паролів, а у деяких випадках цих паролів може бути навіть більше десятка. Зберігати ці паролі на жорсткому диску комп'ютера не є безпечно, а на паперовому або електроному переносному носії досить незручно. І цю проблему вирішують саме менеджери паролів.

Менеджер паролів надійно зашифрує ваші секретні комбінації, а для зберігання даних використовує ваш комп’ютер або мобільний телефон. Якщо така програма має доступ до хмари, ви можете скористатись своїми паролями із будь-якого вашого пристрою. Отже для безпечного захисту всіх ваших паролів від електроних адрес та сайтів вам необхідно пам’ятати лише один єдиний надійний пароль.

Метою кваліфікаційної роботи є реалізація вебзастосунку для захисту паролів користувачів.

Виходячи з поставленої мети, **завданнями дипломної роботи** є:

- Дослідження сучасних програм захисту паролів та їх вразливостей.
- Аналіз загроз менеджерам паролів.
- Розробка та реалізація вебзастосунку для захисту паролів користувачів.

Об’єкт дослідження: захист паролів користувачів веб-менеджером паролів.

Предмет дослідження: аналіз та класифікація менеджерів паролів, переваги та недоліки існуючих менеджерів паролів та аналіз загроз веб-менеджерам паролів.

Новизна: у якості новизни вебзастосунка для захисту паролів користувачів, я ввів повне видалення усіх даних користувача при декількох спробах невірного вводу паролю за для безпеки даних користувача.

Галузь застосування: в епоху інформаційних технологій менеджери паролів стали простим рішенням у запам’ятовуванні та захищеності всіх паролів від ваших онлайн-сервісів, певні ви використовуєте для розваг, інші для спілкування із друзями та близькими, а треті використовуються для роботи. [6]

Практична цінність роботи полягає полягає у тому, що було реалізовано та протестовано програмний модуль для реєстрації, входу та пошуку користувача у вебзастосунок для захисту паролів користувача, що використовує криптографічний алгоритм шифрування RSA, який дозволяє зберегти дані користувача у надійності та безпеці.

РОЗДІЛ 1. ОГЛЯД ТА АНАЛІЗ ПРОГРАМ ДЛЯ БЕЗПЕЧНОГО ЗБЕРІГАННЯ РЕЄСТРАЦІЙНИХ ДАНИХ

1.1. Загальні відомості.

Менеджер паролів − комп'ютерна програма, що може зберігати та вносити нові складні паролі зі зашифрованої бази даних.

Менеджери паролів є одними із найпростіших і найшвидших способів захистити свої дані від несанкціонованого доступу. Вони надають змогу зберегти всі ваші паролі в одному місці, тому вам потрібно лише запам'ятати один головний пароль, проте цей пароль повинен бути надзвичайно надійним для вашого захисту.

Багато людей часто використовують однаковий пароль для кількох облікових записів, що ставить під загрозу безпеку всю їхню інформації у разі порушення якого-небуть з цих облікових записів.

Звісно, підбирати десятки паролів для різних логінів досить складно. Ось тут і допомагає менеджер паролів. Ця програма дозволяє зберігати всі ваші паролі для різних сервісів, тому вам потрібно запам'ятати лише один головний пароль.

Є дві основні проблеми з введенням паролів: люди часто використовують однакові паролі, і вони спочатку вибирають слабкі паролі. З часом усі такі паролі можна зламати. Звичайно, ваша інформація рано чи пізно неминуче опиниться в Інтернеті, і ви мало що можете зробити, щоб цьому запобігти. Але ви можете обмежити шкоду, не використовуючи паролі на кількох сайтах.Проще кажучи, люди дуже погано обирають паролі. Вони використовують передбачувані послідовності літер, прості слова або інші очевидні варіанти, що надає зловмисникам значно більше можливостей для їх вгадування чи злому.

Згідно з аналізом витоків, проведеним охоронною компанією Keeper, найпоширенішим паролем у світі є гнітюче легкий для вгадування «123456». Другим найбільш використовуваним паролем був «123456789», за даними

Keeper, за ним іде передбачуваний «qwerty». Далі було «12345678», а на п’ятому місці було очевидне «111111».Тут на допомогу приходять менеджери паролів. [1]

Менеджер паролів вирішить обидві ці проблеми. Це означає, що ви можете використовувати різні паролі для кожного облікового запису, оскільки вам потрібно запам’ятати лише один. [1]

Він зберігає та шифрує всі ваші дані для входу в сховище, зберігаючи їх у безпеці та недоступними без відповідного головного пароля, який ви встановили та запам’ятали. [1]

Є й інші переваги використання єдиного пароля, наприклад, запам’ятовування всіх імен записів програм, які ви використовуєте лише зрідка. Якщо ви замовляєте в UberEats лише раз на два місяці, ви можете легко завантажити свій пароль (за умови, що ви не використали його повторно), але менеджер паролів зберігатиме його. [1]

Деякі менеджери паролів також мають додаткові функції, які відрізняють їх від конкурентів, зокрема зберігання файлів і створення зашифрованих нотаток. [1]

Менеджери паролів перебувають в одній із трьох таких форм:

- локальне програмне забезпечення;
- онлайн-сервіс;
- апаратний пристрій.

Залежно від типу використовуваного менеджера паролів і можливостей, які він надає, прихована інформація в базі даних паролів зберігається на персональному комп’ютері користувача, на окремому пристрої, який не належить користувачеві, або на сторонніх серверах. Більшість менеджерів паролів просять користувача придумати та запам’ятати так званий головний пароль, щоб надати доступ до інформації, що зберігається в цій базі даних. Існує багато програм для керування паролями, які надають додаткові функції,

що підвищують зручність для користувача, підвищують безпеку бази даних паролів або функцію автозаповнення. [2]

1.1.1 Менеджер паролів у формі локального програмного забезпечення

Як правило, менеджери паролів знаходяться на персональному комп’ютері або мобільному телефоні користувача як програмне забезпечення, встановлене локально. Ця програма може працювати окремо, із захищеною базою даних паролів локально та незалежно від програмного забезпечення менеджера паролів на тому самому пристрої. Крім того, менеджер паролів може запитувати доступ до хмари, щоб база даних паролів, яка залежить від онлайн-сервісу розміщення файлів, зберігалася віддалено від користувача, але керувалася програмним забезпеченням, встановленим на пристрої користувача.Є певні офлайн-менеджери паролів, яким непотрібен доступ в Інтернет, за рахунок цього, можливий витік даних через мережу неможливий. Здебільшого повністю офлайн-менеджер паролів є хоча і не таким зручнішим за онлайн-менеджер паролів, він є значно безпечнішим. [6]

1.1.2 Менеджер паролів у формі онлайн-сервісу

Служба онлайн-менеджера паролів — це веб-сайт, який зберігає ваші облікові дані для входу в безпеку. Такий менеджер паролів є веб-версією звичайного менеджера паролів, який встановлюється на пристрій користувача. [6]

Перевагами онлайн-менеджера паролів перед локальними менеджерами паролів є, по-перше, портативність, тобто їх можна використовувати абсолютно на будь-якому комп’ютері з веб-браузером і підключенням до Інтернету без необхідності встановлення програми локального менеджера паролів. Крім того, значно знижується ризик втрати паролів через крадіжку або пошкодження персонального комп'ютера або мобільного телефону. [6]

Основними недоліками служб онлайн-менеджера паролів є те, що вони вимагають від користувача надання власної інформації, а сама база даних

паролів не розташована на персональному комп'ютері, який використовує користувач. Оскільки основними об'єктами кібератак є хмари та сервери, дуже важливим етапом є спосіб аутентифікації в онлайн-сервісі та вибір надійного ключа для доступу до паролів. Проблеми можуть виникнути через те, що користувачі часто ігнорують безпеку заради власної зручності. Використання одностороннього або двостороннього шифрування також може бути важливим фактором. [6]

1.1.3 Апаратні пристрої менеджерів паролів

Апаратні пристрої — це форма диспетчера паролів на основі маркерів, у якій користувач використовує локально доступний апаратний пристрій, наприклад захищений флеш-накопичувач USB, для автентифікації разом із простим текстовим паролем або замість нього. Деякі системи апаратних пристроїв вимагають програмного забезпечення, завантаженого на ПК, а також апаратного забезпечення та драйверів для належного читання та декодування даних. [6]

Інформація захищена маркером безпеки. Зазвичай користувачеві пропонується багатофакторна аутентифікація шляхом комбінування різних даних: [6]

- щось, що є у користувача, наприклад мобільний додаток, який генеруватиме токен, схожий на смарт-карту, смарт-карту або USB-накопичувач;
- щось, що знає користувач, наприклад PIN-код та/або пароль;
- щось, що є самим користувачем, наприклад біометричні дані, такі як сканер відбитків пальців, сітківки ока чи обличчя.

1.1.4 Переваги менеджерів паролів

Контроль доступу користувача на основі пароля має кілька переваг, а саме те, що його можна легко вбудувати в більшість програмного забезпечення за допомогою API, доступних у багатьох програмах, вони не потребують

серйозних змін у комп’ютері чи сервері, які вже відомі користувачеві за допомогою паролів . Хоча паролі можуть бути досить безпечними, їх слабкою стороною може бути те, як користувач вибирає власний пароль і керує ним за допомогою:

- прості паролі– короткі паролі, паролі, які використовують слова зі словників, або паролі, які не поєднують різні типи символів (великі, малі літери, цифри, розділові знаки), такі паролі легко вгадати;
- паролі, які хтось може знайти–паролі, записані в блокноті, на наліпках, паролі, написані на документі на комп’ютері, у програмі-нагадуванні, на електронному носії або просто у вигляді відкритого тексту;
- один пароль — використання одного пароля для кількох сайтів або облікових записів;
- спільні паролі– користувачі діляться своїми паролями з іншими користувачами, надсилають незашифровані електронні листи з паролем, працівники встановлюють однакові паролі для всіх облікових записів.
- адміністратори дозволяють користувачам, які мають однаковий доступ до інформації, використовувати той самий пароль.

Крім того, менеджери паролів можуть захистити від фішингу та фармінгу. Менеджер паролів, на відміну від людей, може містити сценарій автоматичного входу, який порівнює URL-адресу поточного сайту з URL-адресою в самій базі даних. Якщо ці дві адреси не збігаються, менеджер паролів не заповнює автоматично поле для входу. Цей метод призначений для захисту від подібних веб-сайтів. Саме завдяки цій вбудованій зручності використання менеджера паролів є вигідним, незважаючи на те, що користувачеві потрібно запам’ятати лише кілька паролів. Хоча не кожен менеджер паролів може автоматично обробляти складні процедури входу, які нав’язують банківські веб-сайти, існує багато нових менеджерів паролів, які обробляють досить складні паролі, багатофакторну автентифікацію та багатосторінкові заповнення.

1.2. Аналіз існуючих менеджерів паролів.

1.2.1 KeePass

KeePass — це менеджер паролів, який поширюється під ліцензією GPL. Ця програма підтримує такі алгоритми шифрування паролів у своїх базах даних, як AdvancedEncryptionStandard і Twofish. KeePass перекладено більш ніж на40 різних мов, включаючи українську. Програма має власну портативну версію, яку не потрібно встановлювати. [13]

Рис. 1.2.1. Ярлик менеджера паролів KeePass

1.2.1.1 Управління паролями

Паролі, які зберігаються KeePass, розділені на кілька керованих груп. Далі кожна з цих груп може мати визначальний значок. Ці групи знову можна розділити на підгрупи в деревоподібній організаційній системі.

Крім того, ця програма відстежує час створення, останній доступ, зміни, що відбулися, і закінчення терміну дії кожного збереженого пароля. Файли в KeePass можна зберігати та прикріплювати за допомогою введення пароля або

можна вводити текстові нотатки з деталями пароля. Пов’язана піктограма може бути в будь-якому або кожному введенні пароля. [13]

1.2.1.2 Імпорт і експорт у KeePass

Паролі можна експортувати в різні формати, такі як TXT, XML, HTML і CSV. Експорт даних у форматі XML також можна використовувати в інших програмах і повторно імпортувати в KeePass, але з додаванням плагіна. Експорт у такий формат, як CSV, має багато сумісних сховищ паролів, наприклад закриті комерційні PasswordAgent і PasswordKeeper. Крім того, формат CSV можна імпортувати як таблиці з такої програми, як Microsoft Excel або LibreOfficeCalc / OpenOffice. KeePass може аналізувати, аналізувати та імпортувати вихідні дані TXT CodeWalletPro, комерційного сховища паролів із закритим кодом. [13]

1.2.1.3 Автоматичне введення, глобальні гарячі клавіші та «drag-and-drop»

Менеджер паролів KeePass має глобальні гарячі клавіші автоматичного введення. Якщо ця програма працює у фоновому режимі, а сама база даних відкрита і користувач використовує гарячу клавішу, KeePass шукатиме потрібний запис і виконає автоматичний введення. Абсолютно всі поля, такі як назва, ім'я користувача, URL і пароль, можна вводити в інших вікнах.

Обробка буфера обміну в Windows може дозволити користувачеві клацнути правою кнопкою миші будь-яке поле в списку паролів, щоб скопіювати його вміст у буфер обміну. Програма може очистити буфер обміну при необхідності через деякий час, після того як користувач скопіює один із паролів. KeePass забезпечує захист від моніторів буфера обміну. [13]

1.2.1.4 Вбудований генератор полів

KeePass має вбудований генератор полів, який генерує випадкові паролі. Ці паролі можуть бути створені шляхом взаємодії користувача, наприклад випадкових рухів миші та введення з клавіатури. [13]

1.2.2 1Password

1Password — це менеджер паролів, розроблений AgileBitsInc. Цей менеджер надає користувачам місце для зберігання різних паролів, ліцензій на програмне забезпечення та іншої конфіденційної інформації у віртуальному сховищі, заблокованому головним паролем, захищеним PBKDF2. За замовчуванням зашифроване сховище для користувачів розміщено на серверах AgileBits за щомісячну плату. [14]

Рис. 1.2.2. Екран запуску менеджера паролів 1Password

1.2.2.1 Синхронізація файлу пароля

1Password можна налаштувати за допомогою 1Password.com, платної служби синхронізації сервера, яка підтримується розробником і на основі підписки. У 2017 році 1Password.com представив функцію режиму подорожі для абонентів, яка дозволяє обходити введені паролі, які не призначені як

безпечні для подорожей, з локальної пам’яті пристрою, зменшуючи вплив органів влади в країні, що розблоковують. [15]

1.2.2.2 Розширення для браузера

1Password інтегрується з веб-браузерами, включаючи Safari, Firefox, Chrome, Opera і Edge. Розширення може запам’ятовувати логін для веб-сайтів, автоматично вставляти логіни для веб-сайтів і генерувати випадкові паролі. [15]

1Password також пропонує окреме розширення під назвою 1Password X, доступне для Chrome, Opera і Firefox,. 1Password X було розроблено для роботи без супутньої настільної програми, проте вимагає підписки на 1Password.com. [15]

Що стосується мобільних пристроїв, то 1Password пропонує інтеграцію з браузерами та програмами на пристроях Android і iOS за допомогою різних методів. Більш зручні способи заповнення й збереження інформації для входу передбачені в iOS 12 та AndroidOreo відповідно. [15]

1.2.3 Dashlane

Dashlane — це безкоштовний менеджер паролів із двофакторною автентифікацією, генератором паролів і системою злому облікових записів, яка дозволяє зручно зберігати паролі, щоб ви могли швидко знайти будь-який пароль на будь-якому сайті. Також є платна версія. Програма доступна для Microsoft Windows, macOS, Android і AppleiOS. Додаток не вимогливий до апаратного і програмного забезпечення, споживає мінімум ресурсів і не впливає на швидкість роботи системи і роботу комп'ютера в цілому. [16]

Рис. 1.2.3.1. Ярлик менеджера паролів Dashlane

Розробники надали чудовий інструмент для знайомства з інтерфейсом - інтегрований майстер захисту. Він допоможе вам правильно заповнити всі необхідні поля для доступу та налаштувати параметри синхронізації. Програма може використовуватися як єдиний контейнер для заповнення конфіденційних даних у всіх популярних браузерах, включаючи мобільні пристрої. Однак для того, щоб синхронізувати інформацію між гаджетами і робочим столом, необхідно придбати ліцензійну версію програми. [16]

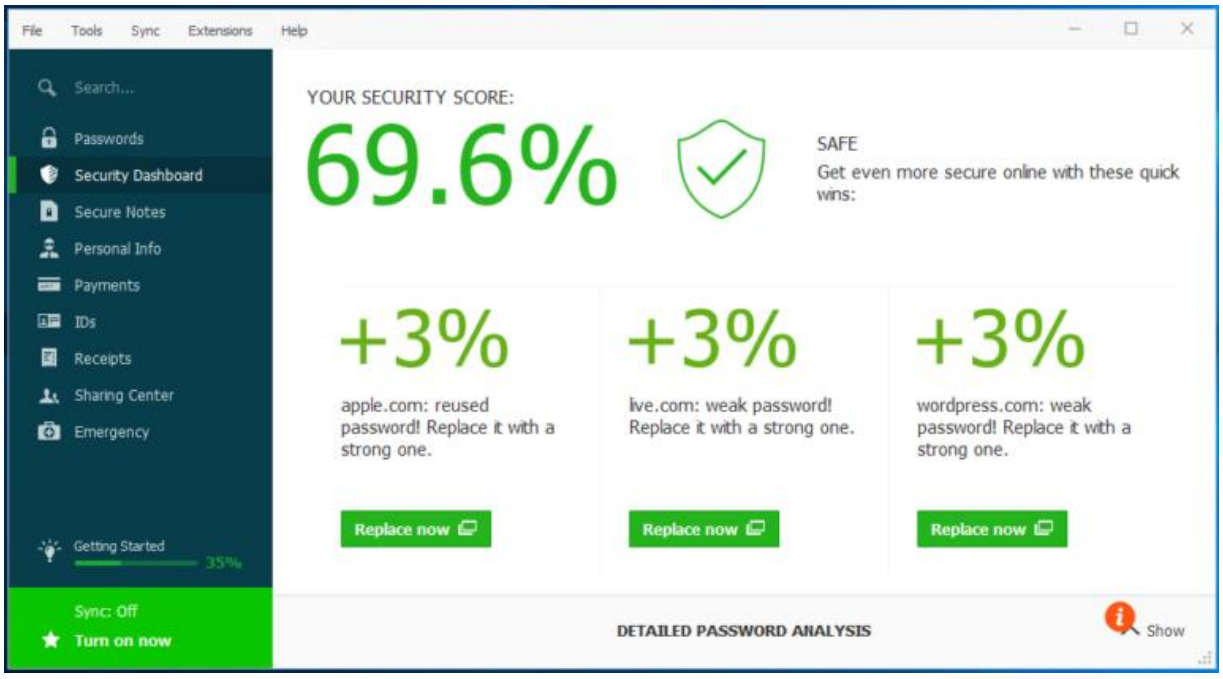

Рис. 1.2.3.2. Головне меню менеджера паролів Dashlane

Особливості основних функцій Dashlane:

- Менеджер паролів дозволяє автоматично авторизуватися в системі;
- Окремий рівень захисту онлайн-банкінгу;
- Програма здатна автоматично змінювати слабкі паролі, роблячи їх більш складними (можливість редагування);
- Спеціальні алгоритми шифрування даних надійно захищають ваш обліковий запис Dashlane;
- Спеціальна панель безпеки показує рівень захищеності паролів від злому і дасть наказ покращити захист;
- Існує система миттєвого сповіщення про спроби створення зловмисних облікових записів.

Також у додатку є генератор паролів, можливість синхронізації та загального доступу з мобільним додатком DashlanePasswordManager.

1.2.3.1 Функціональні розділи.

Збереження паролів. Усі паролі зберігаються в розділі Паролі за групами. Тут ви можете редагувати наявні паролі та створювати нові. Доступні логін/пароль, адреса сайту та поля для приміток. Також є можливість запитати головний пароль під час спроби отримати доступ до запису. Коли ви входите на сайт, для якого вже маєте пароль, Dashlane автоматично введе вашу інформацію. [17]

Примітки. У програмі є розділ «Примітки». Передбачається зберігати такі речі, як паролі програм, паролі Wi-Fi, секретні запитання та взагалі все, що не можна зберігати в розділі паролів. Нотатки можуть бути захищені паролем основної програми. Нотатками можна безпечно ділитися, як і паролями. Просто натисніть SendThisSecurely та введіть свою електронну адресу. Адресат отримає електронний лист із посиланням на ваш пароль. Після перегляду посилання буде знищено. [17]

Розділ персональної інформації. Тут ви можете зберігати особисту інформацію. Ваше ім'я (російською та англійською) з параметрами для

реєстрації на сайтах. Адреси, які будуть використовуватися при заповненні полів для доставки товару. Пошта та мобільні телефони. Dashlane може автоматично вставляти всю цю інформацію у відповідні поля за потреби, що значно спрощує процес реєстрації або придбання продукту. [17]

Паспорти. У розділі IDs можна зберігати паспортні дані та СНІОР. Крім того, програма нагадає про закінчення терміну дії документа. [17]

Гаманець. Тут ви можете ввести дані банківської картки і Dashlane допоможе вам заповнити всю інформацію про борг при покупці в інтернет-магазині, а рахунок автоматично збережеться в розділі «Надходження». Ви можете встановити адресу доставки картки за умовчанням. [17]

1.2.3.2 Захист і генерація паролів

Поєднання надійності і доступності менеджера паролів Dashlane робить його дійсно гідним представником свого класу, якщо говорити про платну версію. Програма починає захищати ПК користувача відразу після встановлення, автоматично збираючи в свою базу даних всі раніше збережені форми з усіх попередньо встановлених браузерів. [17]

Таким чином, відразу після завершення установки утиліта готова до повноцінної роботи по заміні паролів користувачів на нові, згенеровані програмою. При цьому, якщо користувач вирішить заповнити нову форму, вона буде автоматично занесена в базу даних, а автоматично згенерований пароль матиме найвищі характеристики безпеки. [17]

1.3. Висновок до розділу

Проаналізувавши менеджери паролів, можна виокремити вимоги до створення додатку менеджер паролів користувачів:

- Менеджер паролів мусить має бути програмного, онлайн-сервісу або апаратного виду.
- Інтерфейс додатку повинен бути простим та інтуїтивно зрозумілим для користувача.

- Менеджер паролів створюється з метою спростити життя користувачам, тому розповсюджуватись повинен безкоштовно або за мінімальну ціну.

РОЗДІЛ 2. АНАЛІЗ ЗАГРОЗ МЕНЕДЖЕРІВ ПАРОЛІВ

2.1 Вступ

Менеджери паролів можуть мати кілька вразливостей. Якщо паролі в диспетчері паролів зберігаються незашифрованими, злочинець все одно може отримати до них доступ, лише отримавши локальний доступ до машини.

Деякі менеджери паролів використовують вибраний користувачем головний пароль або парольну фразу для створення ключа, який використовується лише для шифрування безпечних паролів. Таким чином, безпека цього підходу залежить від надійності вибраного вами пароля або фрази-паролю, а також від того, чи зберігаються пароль або фраза-пароль локально, де їх може переглянути шкідлива програма або хтось інший. Слабкий головний пароль робить усі інші безпечні паролі вразливими.

Як і в будь-якій системі, яка передбачає введення пароля, основний пароль також можна атакувати або виявити за допомогою клавіатурного журналу або акустичного криптоаналізу. Деякі менеджери паролів намагаються використовувати віртуальні клавіатури, щоб зменшити цей ризик, але це все ще вразливість для реєстраторів ключів, щоб робити знімки екрана даних, введених у менеджер паролів. Але цей ризик можна зменшити за допомогою багатофакторної аутентифікації пристрою.

Деякі менеджери паролів містять генератори паролів. Такі паролі можна вгадати, якщо сам менеджер паролів використовує слабкий генератор випадкових чисел замість використання криптографічного захисту.

Надійний менеджер паролів включає обмежену кількість неправильних записів автентифікації, які дозволені до того, як цей менеджер паролів буде заблоковано та вимагає повторної активації ІТ-службами. Такий захист є найкращим від грубої атаки.

Менеджери паролів, які не перешкоджають зберіганню на жорсткому диску, не перешкоджають отриманню незашифрованих паролів із жорсткого диска комп’ютера.

Що стосується веб-менеджерів паролів, які виконують свою роботу через браузер користувача, вони мають багато недоліків, таких як:

- відсутність дозволу - так звана помилка автентифікації або авторизації. Такі проблеми існують у менеджерах паролів, які дозволяють користувачам ділитися своїми обліковими даними з іншими користувачами.
- недоліки закладки. Онлайн-менеджери паролів часто використовують закладки для входу користувачів. Однак, якщо закладки реалізовані неправильно, шкідливий веб-сайт може використати це, щоб отримати пароль користувача.
- недоліки інтерфейсу користувача. Можливо, менеджер паролів пропонує користувачеві увійти за допомогою div. Це може становити загрозу безпеці, оскільки URL-адреса, яка відображається браузером у момент, коли користувач вводить свій пароль, насправді не є адресою менеджера паролів. Фішер може використати цей метод, створивши підроблений div і отримати облікові дані користувача. Набагато безпечніший підхід — відкрити нову вкладку, де користувач зможе самостійно увійти в менеджер паролів.
- недоліки Інтернету. Поширені веб-уразливості також можуть бутиприсутніми в веб-менеджерах паролів. Зокрема, такі уразливості як XSS і CSRF можуть використовуватися хакерами для отримання паролів.

Крім того, усі менеджери паролів мають той недолік, що будь-якому шкідливому програмному забезпеченню чи хакеру потрібно буде знати лише один пароль, щоб отримати доступ до абсолютно всіх паролів у базі даних. Крім того, менеджери паролів мають стандартизовані місця та методи зберігання паролів, які можуть бути використані зловмисним програмним забезпеченням.

2.2 Блокування менеджерів паролів

Багато різних веб-сайтів намагалися заблокувати менеджери паролів, але часто відмовлялися, коли публічно заперечували. Причиною цього був захист від автоматизованих атак, захист користувачів від фішингу, блокування зловмисного програмного забезпечення або комбінація цих дій. Клієнтське програмне забезпечення безпеки IBM Guardian має деякі чіткі налаштування, які блокують менеджери паролів. Таке блокування критикують експерти з інформаційної безпеки, оскільки воно робить користувачів менш захищеними, а їхнє обґрунтування є фальшивим. [6]

2.3 Загроза SQLInjection

SQL-ін’єкція — це техніка виконання коду, яка використовується для атак на програми, керовані даними, під час яких зловмисні оператори SQL вставляються в поле введення для виконання. SQL-ін’єкція має використовувати вразливість системи безпеки в додатках, що постачаються програмним забезпеченням, наприклад, якщо вхідні дані неправильно відфільтровано на екранні символи рядкового літералу, вбудовані в оператори SQL, або введені користувачем дані не строго введені та виконуються несподівано. SQL-ін’єкція в основному відома як векторна атака для веб-сайтів, але може використовуватися для атаки будь-якого типу бази даних SQL. [18]

Атаки SQL-ін’єкцій дозволяють зловмисникам підробити облікові дані, підробити існуючі дані, спричинити проблеми з відмовою в обслуговуванні, такі як анулювання транзакцій або зміна балансів, повністю відкрити всі дані в системі, знищити дані або іншим чином зробити їх недоступними та стати адміністраторами сервера бази даних . [18]

Рис. 2.4 Зображення SQL-ін’єкції

2.3.1 Форми SQLInjection

Ін'єкція SQL (SQLI) була визнана однією з 10 найбільших уразливостей веб-додатків у 2007 та 2010 роках за версією OpenWebApplicationSecurity. У 2013 році SQLI була визнана атакою номер один у десятці найкращих за версією OWASP. Існує чотири основні підкласи реалізацій SQL: [18]

- класичний SQLI;
- сліпа або логічна SQL-ін'єкція;
- система управління базою даних специфічний SQLI;
- складовий SQLI:
    - SQL-ін'єкція + недостатня автентифікація
    - SQL-ін'єкція + DDoSатаки
    - SQL-ін'єкція + перехоплення DNS
    - SQL-ін'єкція + XSS

2.3.2 Технічна реалізація

1) Неправильно відфільтровані escape-символи

Ця форма реалізації виникає, коли введені користувачем дані не фільтруються для контрольних символів, а потім передаються в оператор SQL.

Це призводить до потенційних маніпуляцій операторами, які виконуються в базі даних кінцевим користувачем програми. [18]

Наступний рядок коду ілюструє вразливість:

*statement = "SELECT * FROM users WHERE name = '" + userName + "';"*

Цей код SQL призначений для отримання записів зазначеного імені користувача з таблиці користувачів. Проте, якщо змінна userName якимось чином створена зловмисником, оператор SQL може зробити більше, ніж задумав автор коду. Наприклад, встановлення змінної "userName" на: [18]

'OR' 1 '=' 1

або використання коментарів навіть для блокування решти запиту. Три рядки мають прогалину в кінці:

*'OR' 1 '=' 1 '-' OR '1' = '1' {'OR' 1 '=' 1 '/ **

відображає один із наступних операторів SQL батьківською мовою:

*SELECT * FROM users WHERE name = '' OR '1' = '1';*

*SELECT * FROM users WHERE name = '' OR ' 1 '=' 1 '-';*

Якби цей код був використаний в процедурі аутентифікації, цей приклад можна було б використати для примусового вибору кожного поля даних (*) від усіх користувачів, а не від одного конкретного імені користувача, як передбачав кодер, оскільки оцінка «1» = « 1" завжди правда [18]

Наступне значення «userName» у наведеному нижче операторі видалить таблицю «users», а також вибере всі дані з таблиці «userinfo» за допомогою API, який дозволяє кілька операторів: [18]

*a '; DROP TABLE users; SELECT * FROM userinfo WHERE 't' = 't*

Це введення відображає остаточний оператор SQL наступним чином і вказано:

*SELECT * FROM users WHERE name =' a '; DROP TABLE users; ВЫБРАТЬ * ОТ userinfo WHERE 't' = 't';*

У той час коли більшість реалізацій SQL Server дозволяють виконувати кілька операторів за допомогою одного виклику, певні API SQL, такі як mysql_query() PHP, не дозволяють це з міркувань безпеки. Це не дозволяє

зловмисникам вводити абсолютно окремі запити, але також не дозволяє їм змінювати запити. [18]

2) SQL-ін'єкція другого порядку

Друге впровадження SQL виконується, якщо надіслане значення містить зловмисні команди, які зберігаються та не підтримуються знову. У деяких випадках програма може правильно закодувати оператор SQL і зберегти його як дійсний SQL. Решта цієї програми без елементів керування захистом від впровадження SQL може виконувати цей збережений оператор SQL. Для цієї атаки потрібні додаткові знання про те, як надається використане представлене значення. Автоматизовані сканери безпеки веб-додатків нелегко виявляють цей тип оператора SQL, і вам може знадобитися вручну перевірити, чи є докази того, що це робиться. [18]

2.4 Загроза Fishing

Фішинг - це вид шахрайства в Інтернеті, метою якого є отримання доступу до ідентифікаційних даних користувачів, тобто облікових записів. [19]

Фішинг здійснюється шляхом розсилки електронних повідомлень-повідомлень від імені різних платіжних систем, банків, соціальних мереж, поштових служб тощо. Лист містить логотип, повідомлення та посилання на сайт. Щоб змусити користувача ввести свої конфіденційні дані у відповідну форму, використовуються різні причини (блокування облікового запису при ігноруванні повідомлення, збій в системі). В результаті шахраї отримують доступ до особистих рахунків, банківських рахунків і облікових записів користувачів. [19]

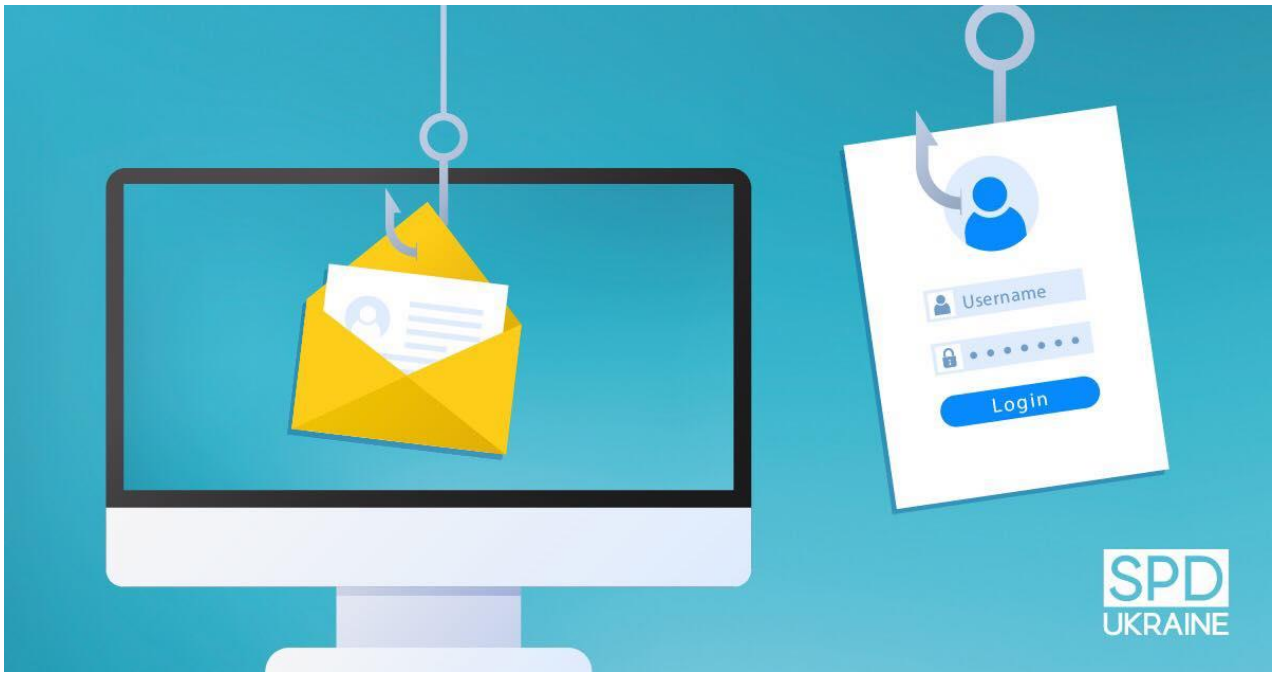

Рис. 2.4 Зображення фішингу

Методи фішингу:

- Використання посилань. Викрадення персональних даних здійснюється через новостворені фішингові сайти, які, як правило, швидко блокуються пошуковими системами. Такі сайти розміщуються на доменах, максимально схожих на реальні, їх дизайн мало чим відрізняється від оригінального. Посилання на фішинговий сайт розміщено в тілі отриманого електронного листа. [19]
- Фермерство. Крадіжка персональних даних відбувається безпосередньо на офіційному сайті. Фермери на DNS-сервері змінюють URL цього сайту на фейковий, на якому користувач проходить процедуру ідентифікації. Заміну побачити неможливо. [19]
- Обхід фільтрів. Алгоритм спам-фільтрів надає змогу розпізнавати текст листків, тому фішери використовують зображення в листках, а не текст. [19]
- Вішинг. Крадіжка даних здійснюється за допомогою телефонного зв'язку. У листі вказано номер телефону, за яким можна передзвонити для вирішення проблеми з банківським рахунком. Для цього оператор

або автовідповідач просить продиктувати номер рахунку та пін-код платіжної картки. [19]

- Розбиття. Посилання на фішинговий сайт надсилаються в SMS-повідомленнях. [19]
- Розсилання вірусів. Під час переходу за фішинговим посиланням на комп'ютер користувача завантажується троянська програма, кейлоггер, шпигун чи інший вірус. [19]

2.5 Загроза MITM-атака

Атака типу man-in-the-middle (MITM), що буквально означає "людина посередині" - це тип кібер-атаки , у якому зловмисники перехоплюють розмову чи передачу даних шляхом підслуховування, або прикидаючись його легальним учасником. Жертві здаватиметься, що відбувається стандартний обмін інформацією, але, вставивши себе в «середину» схеми забезпечення розмови чи передачі, зловмисник може непомітно перехопити інформацію, як показано на рис. 2.5.

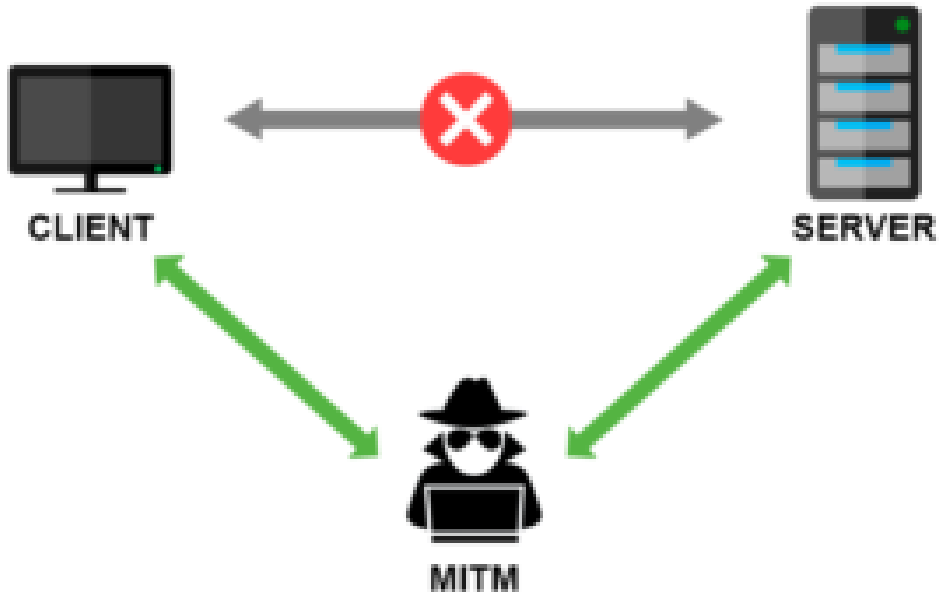

Рис. 2.5 Схема MITM-атаки

Метою MITM-атаки є отримання конфіденційних даних, таких як дані банківського рахунку, номери банківських карток або облікові дані для входу,

які можуть бути використані для скоєння подальших злочинів, таких як крадіжка особистих даних або незаконні перекази коштів. Оскільки MITM-атаки здійснюються в режимі реального часу, вони часто залишаються непоміченими, поки не стає занадто пізно.

2.5.1 Принцип атак

Успішна MITM-атака включає дві конкретні фази: перехоплення і дешифрація.

Перехоплення. Перехоплення передбачає, що зловмисник втручається в процес передачі даних з/в мережу жертви, перехоплюючи їх за допомогою «підставної» мережі, перш ніж дані будуть реально відправлені адресату або надійдуть до мережі жертви. Фаза перехоплення - це, по суті, те, як зловмисник вводить себе як "людину посередині". Зловмисники часто роблять за допомогою створення в громадському місці підробленої точки доступу Wi - Fi , для підключення до якої не потрібний пароль. Якщо жертва підключається до такої «підставної» точки доступу, то зловмисник отримує доступ до будь-якого онлайн обміну даними, який вона виконує.

Як тільки зловмисник успішно вклиниться між жертвою та іншою бажаною стороною обміну інформації, він зможе використати різні методи для продовження атаки:

IP -спуфінг (підміна IP- адрес): Кожен пристрій, підключений до Wi-Fi, має свою адресу Інтернет-протоколу (IP), який відіграє центральну роль у тому, як взаємодіють комп'ютери та пристроїв Мережі. Спуфінг (підміна) IP-адреси передбачає, що зловмисник змінює IP-пакети, щоб видати себе за комп'ютерну систему жертви. Коли жертва намагається отримати доступ до URL-адреси, підключеної до цієї системи, вона неусвідомлено відправляється на сайт зловмисника.

ARP - спуфінг: Під час заміни протоколу дозволу адрес (ARP) зловмисник використовує фальсифіковані повідомлення ARP, щоб зв'язати свою MAC-адресу з легальною IP-адресою жертви. Підключивши свою MAC-

адресу до IP-адреси жертви, зловмисник отримує доступ до будь-яких даних, відправлених на її IP-адресу.

DNS - спуфінг: Підміна сервера доменних імен (DNS), також відома як «отруєння» DNS-кешу, передбачає, що зловмисник змінює IP-адресу DNS-сервера, щоб мати можливість перенаправляти веб-трафік жертви з передбачуваного реального веб-сайту на шахрайський веб-сайт, який дуже нагадує оригінальний. У цьому випадку жертва впевнена, що вона підключається до оригінального веб-сайту, і якщо жертва авторизується за допомогою свого облікового запису, то зловмисники зможуть отримати доступ до персональних, реєстраційних даних та іншої конфіденційної інформації.

Дешифрація. MITM -атака не зупиняється лише фазі перехоплення. Після того, як зловмисник отримає доступ до зашифрованих даних жертви, вони повинні бути розшифровані, щоб зловмисник міг їх прочитати та використовувати у своїх шкідливих цілях. Для розшифровки даних жертви може бути використаний ряд методів без попередження користувача або появи в програмі жертви будь-якого попередження:

Підміна HTTPS (HTTPS -спуфінг): Підміна HTTPS - це метод обману вашого браузера, в результаті якого браузер «вважає», що завантажуваний веб-сайт безпечний і автентичний, хоча це не так. Коли жертва намагається підключитися до захищеного сайту, до її браузера надсилається підроблений сертифікат, який натомість приводить жертву на шкідливий веб-сайт зловмисника. Це дає зловмиснику доступ до будь-яких даних, якими жертва ділиться на цьому сайті.

Перехоплення SSL (SSL Hijacking): Щоразу, коли ви підключаєтеся до незахищеного веб-сайту, адреса якого в полі для URL-адреси починається з «HTTP», ваш сервер автоматично перенаправляє вас на захищену версію HTTPS цього сайту. При перехопленні SSL зловмисник використовує свій власний комп'ютер та сервер для перехоплення цього перенаправлення, що дозволяє йому переривати будь-яку інформацію, що передається між

комп'ютером користувача та сервером. Це дозволяє кібер-злочинцеві отримати доступ до будь-якої конфіденційної інформації, яку користувач використовує під час свого підключення до цього веб-сайту.

SSL Stripping : SSL stripping передбачає, що зловмисник перериває з'єднання між користувачем та веб-сайтом. Це робиться шляхом зниження рівня захищеного HTTPS-з'єднання користувача до небезпечної HTTP-версії веб-сайту. У цьому випадку користувача підключають до незахищеного сайту, тоді як зловмисник підтримує з'єднання із захищеним сайтом, роблячи дії користувача видимими для зловмисника навіть у незашифрованому вигляді.

2.5.2 Як виявити MITM-атаку

Якщо ви не займаєтеся активним пошуком ознак того, що ваші онлайн-повідомлення були перехоплені або скомпрометовані, виявити атаку man - in - the - middle може бути вкрай складно. Хоча таким атакам легко залишитися непоміченими, є певні речі, на які вам слід звернути увагу при перегляді веб-сторінок - в основному це стосується URL -адреси у вашому адресному рядку.

URL -адреса захищеного веб-сайту починається з « HTTPS ». Якщо в URL -адресі відсутня буква « S » наприкінці і вона читається як « HTTP », це відразу сигналізує про те, що ваше з'єднання небезпечне. Вам також слід звернути увагу на SSL -значок замочка зліва від URL -адреси, який також позначає захищений веб-сайт.

Крім того, будьте обережні при підключенні до громадським мережам Wi-Fi . Як обговорювалося вище, кібер-злочинці часто шпигуть за публічними мережами Wi - Fi і використовують їх для атаки man - in - the - middle . Найкраще ніколи не вважати, що загальнодоступна мережа Wi - Fi є легальною, і взагалі уникати підключення до невідомих та загальнодоступних мереж Wi - Fi .

Як підготуватися та запобігти. Хоча і важливо знати, як виявити потенційну MITM- атаку, але найкращий спосіб захиститися від подібних атак

- це запобігти їх у принципі. Настійно радимо вам дотримуватися наших рекомендацій :

- Уникайте мереж Wi - Fi , які не захищені паролем , і ніколи не використовуйте загальнодоступну публічну мережу Wi-Fi для виконання таких конфіденційних операцій, які вимагають передачі ваших персональних даних
- Використовуйте підключення через віртуальну приватну мережу (Virtual Private Network , VPN) – особливо при підключенні до Інтернету у громадському місці. VPN шифрує вашу онлайн-активність і не дозволяє зловмиснику "прочитати" ваш Інтернет-трафік і отримати доступ до ваших персональних даних, таких як паролі або інформацію про банківський рахунок.
- Виходьте при роботі на критично важливих веб-сайтах (наприклад, на сайті онлайн-банку) відразу після того, як ви закінчите роботу з цим сайтом, щоб унеможливити перехоплення вашої сесії зловмисником.
- Дотримуйтесь правильних звичок по відношенню до паролів , наприклад, ніколи не використовуйте однакові паролі для різних облікових записів, використовуйте менеджер паролів для забезпечення максимального рівня їх безпеки.
- Використовуйте мультифакторну авторизацію для всіх ваших облікових записів.
- Використовуйте фаєрвол для забезпечення безпечних Інтернет-з'єднань.
- Використовуйте антивірусну програму для захисту ваших пристроїв від шкідливих програм.

У міру того, як наш світ, пов'язаний з цифровими технологіями, продовжує стрімко розвиватися, зростає і рівень розвитку кібер-злочинності та використання вразливостей у системах безпеки. Турбота про те, щоб навчити себе передовим методам інформаційної безпеки, має вирішальне значення для захисту від атак man - in - the - middle та інших видів кібер-загроз.

Принаймні наявність потужної та надійної антивірусної програми має велике значення для забезпечення безпеки та збереження ваших даних.

2.6 Загроза BruteForce

Brute force – це спроба підібрати пароль або ключ шифрування. Метод полягає в послідовному переборі різних комбінацій символів доти , доки одна з них не підійде. Пошук пароля ведеться не вручну: для цього використовуються спеціальні програми та послуги.

Таким чином, атака brute force – це злом системи з метою отримати доступ до захищених даних.

Концепція брутфорсу - хакер краде пароль

Рис. 2.4 Зображення BruteForce

2.6.1 Класифікація brute force

Існує кілька видів атак:

- злом ПК;
- зламування електронної пошти;
- злом облікових записів;
- злом сайту.

Атака може бути спрямована як на приватну особу, так і на організацію.

Атака на ПК. Це вид персонального злому: хакер намагається підібрати пароль користувача, щоб отримати доступ до особистих даних та керування комп'ютером. Для цього застосовують програмне забезпечення та шахрайські схеми отримання інформації. Наприклад, зловмисник у телефонній розмові або онлайн-спілкуванні з жертвою кібератаки може з'ясувати такі дані:

- кличку домашньої тварини;
- дату народження та інші значні числові комбінації;
- імена дітей та родичів.

Отримані відомості допомагають при зломі, особливо якщо в якості пароля використовуються повні слова із змістом: наприклад, Ivan2007 або Ivan_Petrov. У цьому випадку йдеться про « словникову атаку » , коли для брутфорсу застосовуються словники.

Атака на електронну пошту. Кіберзлочинці намагаються зламати електронну поштову скриньку користувача, щоб отримати доступ до його облікового запису та інформації в листах. Для цього використовуються уразливості у поштовому сервісі.

Атака на облікові записи. Один із різновидів brute force — брут-чек. Як виглядає схема:

- при реєстрації в соцмережах та на онлайн-ресурсах користувачі вказують email;
- на вказаний email приходять логін та пароль для доступу до облікового запису;
- хакери за допомогою спеціального програмного забезпечення знаходять такі листи в пошті користувачів;
- ПЗ видає перелік логінів та паролів, за якими хакери можуть увійти в акаунти.

Злом облікових записів загрожує втратою облікового запису, витоком особистого листування та конфіденційної інформації. Допустимо, користувач зберігає в пошті або чаті в соцмережах такі дані:

- фотографію паспорта;
- реквізити банківської картки;
- паролі до 1С, CRM та інших робочих програм.

В результаті атаки brute force всі ці дані потрапляють до рук шахраїв, що може призвести до фінансових та репутаційних втрат.

Атака на сайт. Метод базується на пошуку вразливостей веб-сайту. Мета атаки — отримати доступ до домену, бази передплатників та бізнес-даних.

Часто сайти зламують для розсилки реклами: шахраї використовують домен, щоб поштові сервіси не впізнали листи як спам. Також хакери можуть вимагати гроші за відновлення доступів, шантажувати власника сайту видаленням даних та здійснювати інші протиправні дії.

2.6.2 Захист від брутфорс-атак

Зламування облікових записів і підбір паролів є кіберзлочином. Наслідки брутфорсних атак - штраф або позбавлення волі до семи років (стаття КК РФ 272 " Неправомірний доступ до комп'ютерної інформації ").

Однак для запобігання злочинам потрібно піклуватися про кібербезпеку самостійно:

- Користувачі повинні відповідально підходити до зберігання персональних даних;
- компанії - впроваджувати інструменти для захисту бізнесу від кіберзагроз.

Надійні паролі. Чим складніший пароль, тим складніше його підібрати. За статистикою, 77% зламаних паролів містять менше восьми символів. Додавання розділових знаків і спеціальних символів значно ускладнює брутфорс. Тому рекомендується використовувати паролі від восьми знаків, що складаються з букв, цифр та спецсимволів.

Двофакторна автентифікація. Спосіб заснований на багатоетапній перевірці користувача. Для підтвердження особи використовується не тільки пароль, а й додатковий фактор, наприклад:

- код із СМС або email;
- пуш-повідомлення;
- токен.

Двофакторна автентифікація робить атаку складнішою та підвищує рівень захисту даних.

Біометрія. Замість пароля можна використовувати метод ідентифікації за допомогою біометричних даних. Наприклад, для входу в програму або розблокування телефону потрібно відсканувати палець або використовувати систему розпізнавання облич.

При цьому не варто повністю покладатися на біометрію. Вже з'явилися системи підбору зображення відбитків, які дозволяють розблокувати смартфони на iOS та Android без сканування пальця власника. Тому захист доступу до смартфону не звільняє від необхідності ставити паролі на пошту, месенджери, банківські програми.

Захист додатків. На рівні розробників сайтів та програм можна обмежити кількість спроб введення пароля. Наприклад, після трьох невдалих спроб система блокує можливість входу на кілька хвилин чи годин.

Цей метод не є панацеєю від кібератак, але дає можливість вчасно виявити загрозу та сповістити користувача про спробу злому.

Навчання користувачів. Щоб запобігти витоку даних на корпоративному рівні, важливо забезпечити дотримання правил інформаційної безпеки на рівні кінцевих користувачів. Для цього потрібно навчити співробітників використовувати надійні паролі, розпізнавати спам та фішинг.

Захист бізнесу. На рівні адміністратора можна заборонити співробітникам здійснювати дії, які можуть призвести до кібератаки. Наприклад, з метою безпеки багато компаній відключають можливість

встановлення сторонніх програм на робочі комп'ютери та інші пристрої, підключені до корпоративної мережі.

Також деякі організації закривають доступи до онлайн-ресурсів, дозволяючи користуватися лише сайтами, необхідними для роботи. А в поштовому клієнті можна настроїти примусове оновлення пароля через кожні два чи три місяці.

Щоб забезпечити комплексний захист компанії, потрібно застосовувати набір інструментів з урахуванням галузевої специфіки. Таку можливість надають рішення для захисту бізнесу від кіберзагроз. Переваги продуктів:

- допомагають побудувати повний захист від цифрових ризиків;
- відповідають вимогам законодавства щодо роботи з персональними даними;
- підбираються індивідуально залежно від масштабу бізнесу, ніші та завдань.

Використання надійного ПЗ. Часто для брутфорс-атаки потрібна вразливість. Чим складніша IT-структура, чим більше програмних продуктів використовує компанія, тим вищі ризики витоку даних чи злому. Тому важливо вибирати програмне забезпечення з високим захистом від кіберзагроз. Наприклад, продукти для бізнесу від Сбера створені відповідно до банківських стандартів безпеки та відповідають світовим вимогам щодо захисту даних.

2.7 Загроза Keylogger

Кейлоггери — це особливо підступний тип шпигунського програмного забезпечення, яке може записувати та викрадати послідовні натискання клавіш (і багато іншого), які користувач вводить на пристрої. Термін кейлоггер або «реєстратор натискань клавіш» не потребує пояснень: програмне забезпечення, яке реєструє те, що ви вводите на клавіатурі. Однак клавіатурні шпигуни також можуть дозволити кіберзлочинцям підслуховувати вас,

спостерігати за вами на системній камері або прослуховувати через мікрофон вашого смартфона.

Рис. 2.4 Зображення Keylogger

Кейлоггери — це поширений інструмент для корпорацій, який IT-відділи використовують для усунення технічних проблем у своїх системах і мережах або для таємного спостереження за співробітниками. Те саме стосується, скажімо, батьків, які хочуть стежити за діяльністю своїх дітей. Підозрілі подружжя — ще один ринок для кейлоггерів.

У всіх таких випадках, якщо організація чи особа, яка завантажує та встановлює кейлоггер, насправді є власником пристрою, то це абсолютно законно. І в Інтернеті є тисячі комерційно доступних кейлоггерів, які рекламують себе саме для такого використання.

Однак клавіатурні шпигуни викликають занепокоєння, коли за ними стоять зловмисники. І вони точно не володіють зараженим пристроєм. Ви не знаєте, що вони зламали ваш комп’ютер; залежно від типу кейлоггера, він може викрадати будь-які паролі, які ви ввели, періодично робити знімки екрана, записувати веб-сторінки, які ви переглядаєте, захоплювати ваші надіслані електронні листи та будь-які сеанси обміну миттєвими

повідомленнями, а також конфіденційну фінансову інформацію (наприклад, номери кредитних карток, PIN-коди та банківські рахунки), а потім надішліть усі ці дані через мережу на віддалений комп'ютер або веб-сервер. Там особа, яка керує програмою журналювання, може отримати все це, потенційно надсилаючи третім особам у злочинних цілях.

2.7.1 Класифікація

Клавіатурні шпильки бувають принаймні двох широких різновидів — апаратні пристрої та більш звичне програмне забезпечення. Апаратні пристрої можуть бути вбудовані у внутрішнє обладнання ПК або являти собою непомітний плагін, який таємно вставляється в порт клавіатури між центральним процесором і кабелем клавіатури, щоб він перехоплював усі сигнали під час введення. Але це означає, що кіберзлочинець повинен мати фізичний доступ до ПК, поки вас немає, щоб встановити апаратні кейлоггери.

Програмні кейлоггери набагато легше познайомити та встановити на пристрої жертв, тому цей різновид набагато більш поширений. На відміну від інших видів зловмисного програмного забезпечення, програмні кейлоггери не становлять загрози для систем, які вони самі заражають. Насправді вся суть кейлоггерів полягає в тому, щоб працювати за лаштунками, винюхуючи натискання клавіш, у той час як комп'ютер продовжує нормально працювати. Але навіть якщо вони не завдають шкоди апаратному забезпеченню, клавіатурні шпигуни безперечно становлять загрозу для користувачів, особливо коли вони викрадають конфіденційні дані.

2.7.2 Види доступу до інформації

Кейлоггери вторгаються в комп'ютери (і комп'ютери Mac, і Android, і iPhone) так само, як і інші шкідливі програми. Вони встановлюються, коли ви клацаєте вкладений файл, який вас обдурили відкрити — найчастіше через те, що ви потрапили на схему соціальної інженерії чи хитро сплановану фішингову експедицію. Вкладення можуть надійти вам електронною поштою,

текстовим повідомленням, миттєвим повідомленням, у соціальних мережах або навіть під час відвідування легітимного, але зараженого веб-сайту, який використовує вразливість у ньому та видаляє завантажене шкідливе програмне забезпечення. Крім того, кейлоггери рідко приходять самостійно. Той самий троян , який доставляє клавіатурний шпигун, може підключити до вашої системи інше зловмисне програмне забезпечення, наприклад рекламне ПЗ , шпигунське ПЗ , програмне забезпечення-вимагач або вірус .

Зараження апаратного кейлоггера відбувається, якщо хтось отримує доступ до вашого розблокованого пристрою, що може сприяти будь-якій кількості сценаріїв. Скажімо, шахрай якимось чином встановлює штекер кейлоггера в USB-порт клавіатури комп'ютера кредитного спеціаліста банку. Це дає оператору кейлоггера всі види даних, які можна використовувати під час виконання звичайних обов'язків кредитного спеціаліста. Комп'ютери відділу корпоративної бухгалтерії – ще одна багата мета. А що, якщо ви вирішите скористатися громадським комп'ютером, щоб зробити покупки? Остання особа, яка користується комп'ютером Інтернет-кафе, може бути наступною, хто використає ваші конфіденційні дані.

2.8.1 Загроза Cross-SiteRequestForgery

Підробка міжсайтового запиту (CSRF) — це атака, яка змушує кінцевого користувача виконувати небажані дії у веб-програмі, у якій він наразі пройшов автентифікацію. За допомогою соціальної інженерії (наприклад, надсилання посилання електронною поштою чи чатом) зловмисник може обманом змусити користувачів веб-програми виконати дії за вибором зловмисника. Якщо жертва є звичайним користувачем, успішна атака CSRF може змусити користувача виконувати запити на зміну стану, як-от переказ коштів, зміна адреси електронної пошти тощо. Якщо жертвою є обліковий запис адміністратора, CSRF може скомпрометувати всю веб-програму. [22]

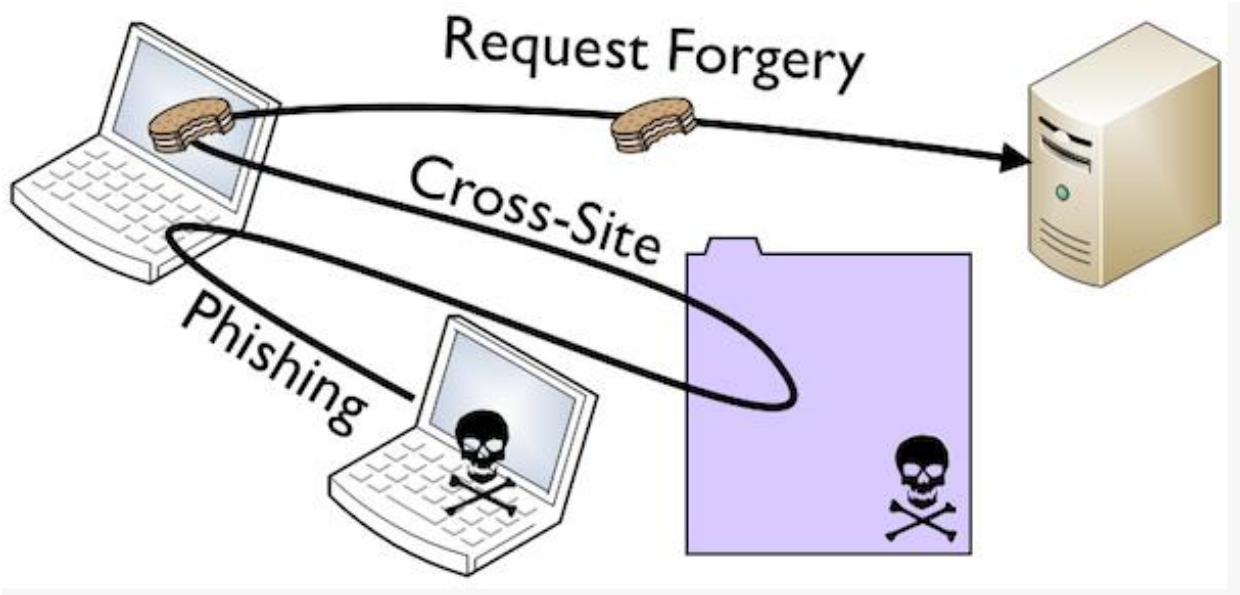

Рис. 2.8 Схема загрозиCross-SiteRequestForgery

CSRF – це атака, яка обманом змушує жертву надіслати зловмисний запит. Він успадковує ідентифікаційні дані та привілеї жертви для виконання небажаної функції від імені жертви (хоча зауважте, що це не стосується CSRF входу, спеціальної форми атаки, описаної нижче). Для більшості сайтів запити браузера автоматично включають будь-які облікові дані, пов'язані із сайтом, наприклад файл cookie сеансу користувача, IP-адресу, облікові дані домену Windows тощо. Таким чином, якщо користувач наразі автентифікований на сайті, сайт не зможе відрізнити підроблений запит, надісланий жертвою, від законного запиту, надісланого жертвою. [22]

CSRF-атаки націлені на функції, які спричиняють зміну стану на сервері, наприклад, зміна адреси електронної пошти чи пароля жертви або покупка чогось. Змушення жертви отримати дані не приносить користі зловмиснику, оскільки зловмисник не отримує відповідь, а жертва. Таким чином, CSRF-атаки спрямовані на запити, що змінюють стан. [22]

Зловмисник може використовувати CSRF для отримання особистих даних жертви за допомогою спеціальної форми атаки, відомої як CSRF для входу. Зловмисник змушує неавтентифікованого користувача увійти в обліковий запис, яким керує зловмисник. Якщо жертва цього не усвідомлює, вона може додати особисті дані, наприклад дані кредитної картки, до

облікового запису. Потім зловмисник може знову увійти в обліковий запис, щоб переглянути ці дані, а також історію активності жертви у веб-додатку. [22]

Іноді можна зберегти CSRF-атаку на самому вразливому сайті. Такі вразливості називаються «збереженими дефектами CSRF». Це можна зробити, просто зберігаючи тег IMG або IFRAME у полі, яке приймає HTML, або за допомогою більш складної міжсайтової атаки сценаріїв. Якщо атака може зберігати CSRF-атаку на сайті, серйозність атаки посилюється. Зокрема, ймовірність підвищується, оскільки жертва з більшою ймовірністю перегляне сторінку, на якій міститься атака, ніж якусь випадкову сторінку в Інтернеті. Імовірність також збільшується, тому що жертва напевно вже автентифікована на сайті. [22]

2.8.2 Заходи профілактики, які НЕ працюють

З часом було розроблено низку помилкових ідей щодо захисту від атак CSRF. Ось кілька, яких ми рекомендуємо уникати. [22]

Використання секретного файлу cookie. Пам’ятайте, що всі файли cookie, навіть секретні , надсилатимуться з кожним запитом. Усі маркери автентифікації надсилатимуться незалежно від того, чи був кінцевий користувач обманом надісланий запит. Крім того, ідентифікатори сеансу просто використовуються контейнером додатка, щоб зв’язати запит із певним об’єктом сеансу. Ідентифікатор сеансу не підтверджує, що кінцевий користувач мав намір надіслати запит. [22]

Приймаються лише запити POST. Програми можуть бути розроблені так, щоб приймати лише запити POST для виконання бізнес-логіки. Помилкове уявлення полягає в тому, що оскільки зловмисник не може створити шкідливе посилання, атака CSRF не може бути виконана. На жаль, ця логіка невірна. Існує багато методів, за допомогою яких зловмисник може обманом змусити жертву надіслати підроблений запит POST, наприклад проста форма, розміщена на веб-сайті зловмисника, із прихованими значеннями. Ця форма може бути запущена автоматично за допомогою

JavaScript або може бути запущена жертвою, яка думає, що форма зробить щось інше. [22]

Багатокрокові транзакції. Багатокрокові транзакції не є адекватною профілактикою CSRF. Поки зловмисник може передбачити або вивести кожен крок завершеної транзакції, CSRF можливий. [22]

Переписування URL. Це може розглядатися як корисна техніка запобігання CSRF, оскільки зловмисник не може вгадати ідентифікатор сеансу жертви. Однак ідентифікатор сеансу користувача відображається в URL-адресі. Ми не рекомендуємо виправляти один недолік безпеки, запроваджуючи інший. [22]

HTTPS. HTTPS сам по собі нічого не захищає від CSRF. Однак HTTPS слід розглядати як передумову надійності будь-яких профілактичних заходів. [22]

Перевірка заголовка посилання. Це не працює на практиці, оскільки заголовок посилання може бути легко підроблений зловмисником. Крім того, деякі користувачі або веб-переглядачі можуть не надсилати заголовок посилання через налаштування конфіденційності чи політики, що призводить до помилкових спрацьовувань. Крім того, існують ситуації, коли реферер може бути нульовим, наприклад, коли користувач переходить на сайт із закладки або будь-якого іншого ресурсу без традиційної URL-адреси. У цих сценаріях законні запити можуть бути помилково сприйняті як потенційні атаки CSRF, що призведе до появи більшої кількості хибних позитивних позначок. [22]

2.9 Висновок до розділу

Проаналізувавши загрози я дійшов до таких висновків:

- Щоб у певній мірі захистити себе від атак фішингу, достатньо не переходити по посиланню від незнайомих користувачів.

- Захист від атак грубої сили − тобто захист від атак перебором, для цього достатньо встановити тимчасове блокування користувача при вводі неправильного логіну та паролю певну кількість раз.
- Захист від кейлоггера на вашому пристрої – блокування його до саме встановлення, для цього вам потрібні гарне антивірусне програмне забезпечення.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ ДЛЯ ЗАХИСТУ ПАРОЛІВ КОРИСТУВАЧІВ.

3.1 Опис розробленого вебзастосунку для захисту паролів користувачів.

3.1.1 Мова програмування

Java - широко використовується мова програмування для написання інтернет-додатків. Мова Java широко використовувалася протягом більш як двох десятиліть. Мільйони програм Java використовуються і сьогодні. Java - це багатоплатформна, об'єктно-орієнтована і сетецентрична мова, яка сама по собі може використовуватися як платформа. Це швидка, безпечна та надійна мова програмування для всього: від мобільних додатків та корпоративного ПЗ до додатків для роботи з великими даними та серверних технологій.

Оскільки Java є безкоштовною та універсальною мовою, на ній створюються локалізовані та розповсюджувані програми:

- Розробка ігор. Багато популярних мобільних, комп'ютерних і відеоігор створено на Java. Навіть сучасні ігри, в яких використовуються передові технології, такі як машинне навчання або віртуальна реальність, створюються за допомогою технології Java.
- Хмарні обчислення. Мова Java часто називають WORA (Write Once and Run Anywhere – «Напиши один раз, запускай будь-де»), що робить його ідеальним для децентралізованих хмарних додатків. Постачальники хмарних послуг вибирають мову Java для запуску програм на широкому спектрі базових платформ.
- Великі дані. Мова Java використовується для механізмів обробки даних, які можуть працювати зі складними наборами даних та великими обсягами даних у режимі реального часу.
- Штучний інтелект. Java – це джерело бібліотек машинного навчання. Завдяки своїй стабільності та швидкості мова стала вибором № 1 для розробки додатків штучного інтелекту, таких як обробка природної мови та глибоке навчання.

- Інтернет речей. Мова Java використовується для програмування датчиків та апаратного забезпечення периферійних пристроїв, які можуть незалежно підключатися до Інтернету.

Секрет популярності Java полягає у простоті його використання. Деякі причини, через які розробники віддають перевагу Java іншим мовам програмування:

- Високоякісні навчальні ресурси. Оскільки Java існує вже давно, для нових програмістів є безліч навчальних ресурсів. Детальна документація, вичерпні друковані матеріали та курси допомагають розробникам протягом усього навчання. Крім того, новачки можуть почати писати код Core Java, перш ніж переходити на Advanced Java.
- Вбудовані функції та бібліотеки. При використанні Java розробникам не потрібно щоразу писати нову функцію з нуля. Як альтернатива Java надає багату екосистему вбудованих функцій та бібліотек для розробки низки додатків.
- Активна підтримка спільноти. Java має багато активних користувачів і спільноту, яка може підтримати розробників, коли вони стикаються з труднощами при написанні коду. ПЗ Java також регулярно підтримується та оновлюється.
- Високоякісні інструменти розробки. Мова Java пропонує різні інструменти для підтримки автоматизованого редагування, налагодження, тестування, розгортання та керування змінами. Ці інструменти роблять програмування на Java економічним та швидким.
- Незалежність від платформи. Код Java може працювати на будь-якій базовій платформі, такій як Windows, Linux, iOS або Android без перезапису. Таким чином, мова особливо ефективна в сучасному середовищі, де програми запускаються на декількох пристроях.
- Безпека. Користувачі можуть завантажувати ненадійний код Java через мережу і запускати його в безпечному середовищі, в якому він не може

завдати жодної шкоди. Ненадійний код не може заразити вірус хост-системи, а також не може читати або записувати файли з жорсткого диска. Рівні безпеки та обмеження у Java також легко налаштовуються.

Усі мови програмування є засобом спілкування із машинами. Апаратне забезпечення машини реагує лише на електронний зв'язок. Мови програмування високого рівня, такі як Java, відіграють роль моста між людською та апаратною мовою. Для використання Java розробники повинні розуміти дві речі.

1. *Мова Java та API*. Це зовнішній інтерфейс між розробником та платформою Java.
2. *Віртуальна машина Java*. Це внутрішній зв'язок між платформою Java та базовим апаратним обладнанням.

Віртуальна машина Java діє як додатковий рівень абстракції між платформою Java та базовим апаратним забезпеченням машини. Вихідний код Java може працювати лише на машинах, на яких встановлена віртуальна машина Java (Java Virtual Machine, JVM). Відповідь питанням, навіщо потрібна віртуальна машина Java, лежить у історії програмування.

3.1.2 Фреймворк Spring MVC

Фреймворк Spring MVC забезпечує архітектуру шаблону «Модель — Подання — Контролер» (Модель — Відображення (далі — Подання) — Контролер) за допомогою слабозв’язаних готових компонентів. Шаблон MVC розділяє аспекти програми (логіку введення, бізнес-логіку та логіку інтерфейсу користувача), забезпечуючи слабкий зв’язок між ними, як показано на малюнку 2.1.2.1.

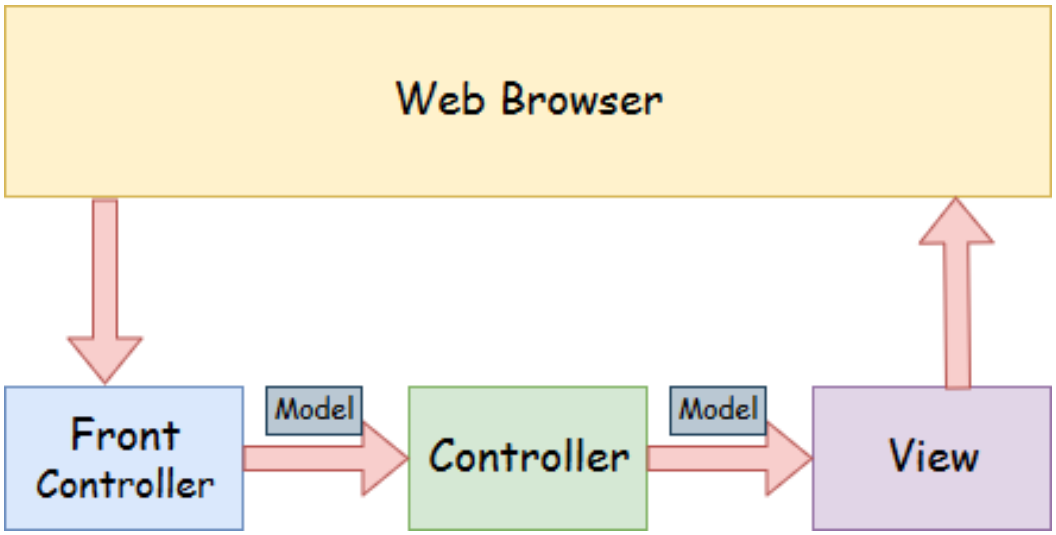

Рис. 3.1.2.1 Модель роботи фреймворкаSpring MVC

• Модель інкапсулює (комбінує) дані програми, загалом вона складатиметься з POJO.

• View відповідає за відображення даних моделі, зазвичай генеруючи HTML, який ми бачимо в нашому браузері.

• Контролер (Controller) обробляє запит користувача, створює відповідну модель і передає її відображення в View.

Вся логіка Spring MVC побудована навколо DispatcherServlet, який отримує й обробляє всі HTTP-запити (від інтерфейсу користувача) і відповіді на них. Нижче наведено послідовність подій, що відповідають вхідному запиту HTTP:

• Після отримання HTTP-запиту DispatcherServlet викликає інтерфейс HandlerMapping, який визначає, який Контролер слід викликати, а потім надсилає запит потрібному Контролеру.

• Контролер приймає запит і викликає відповідний метод служби на основі GET або POST. Викликаний метод визначає дані моделі на основі певної бізнес-логіки та повертає назву представлення до DispatcherServlet.

• Використовуючи інтерфейс ViewResolverDispatcherServlet, він визначає, який View використовувати на основі отриманого імені.

• Після створення представлення DispatcherServlet надсилає дані моделі у вигляді атрибутів до представлення, яке зрештою відображається у браузері.

Вся логіка роботи Spring MVC побудована навколо DispatcherServlet , який приймає та обробляє всі HTTP-запити (з UI) та відповіді на них. Робочий процес обробки запиту DispatcherServlet'ом проілюстрований на наступному рисунку:

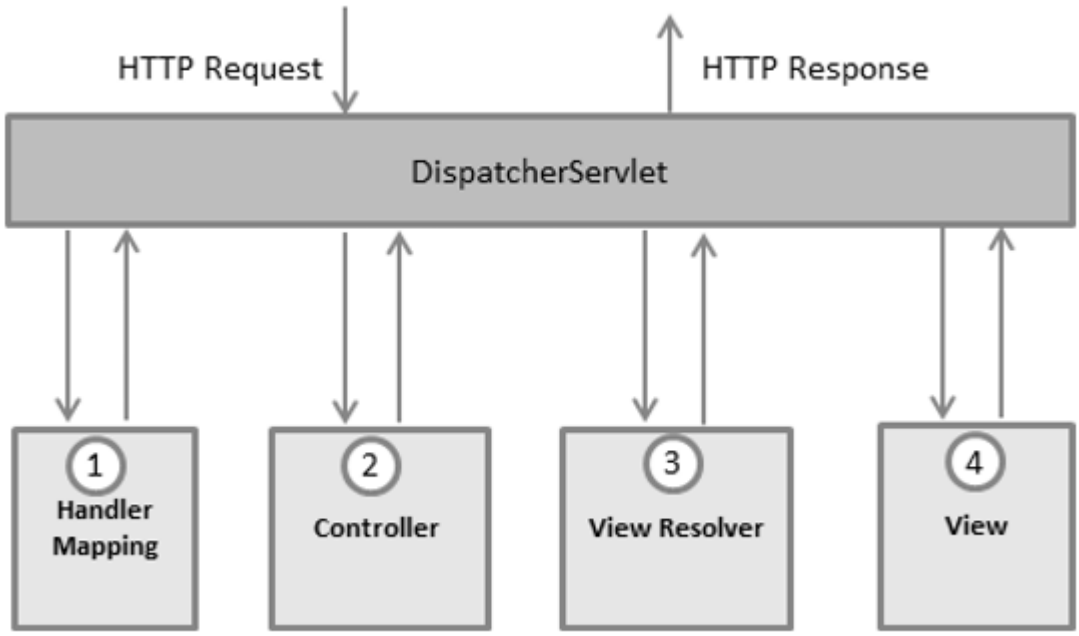

Рис. 3.1.2.2 Процес обробки запиту DispatcherServlet'ом

Нижче наведено послідовність подій, що відповідає вхідному HTTP-запиту:

- Після отримання HTTP-запиту DispatcherServlet звертається до інтерфейсу HandlerMapping , який визначає, який Контролер має бути викликаний, після чого надсилає запит у потрібний Контролер.
- Контролер приймає запит і викликає відповідний службовий метод, що базується на GET або POST. Викликаний метод визначає дані Моделі, що базуються на певній бізнес-логіці і повертає в DispatcherServlet ім'я Вида (View).

- За допомогою інтерфейсу ViewResolver DispatcherServlet визначає, який Вигляд потрібно використовувати на основі отриманого імені.
- Після того, як Вигляд (View) створений, DispatcherServlet відправляє дані Моделі у вигляді атрибутів Вигляд, який в кінцевому підсумку відображається в браузері.

Всі вищезгадані компоненти, а саме HandlerMapping , Controller і ViewResolver , є частинами інтерфейсу WebApplicationContext extends ApplicationContext , з деякими додатковими особливостями, необхідними для створення web-додатків.

3.1.3 Інструмен автоматизації Maven

ApacheMaven — це фреймворк для автоматизації складання проектів на основі опису їх структури у файлах мовою POM (Project ObjectModel), яка є підмножиною XML. Проект Maven опубліковано спільнотою ApacheSoftwareFoundation, яка формально є частиною Джакартського проекту.

Назва системи є словом на ідиш, значення якого приблизно можна виразити як «збирач знань».

Maven забезпечує декларативне, а не обов'язкове створення проекту. Файли опису проекту містять його специфікацію, а не окремі команди виконання. Усі завдання обробки файлів, описані в специфікації, виконуються Maven за допомогою їх обробки послідовністю вбудованих і зовнішніх плагінів.

Інформація для створення проекту, який підтримується ApacheMaven, міститься у файлі XML під назвою pom.xml. Під час запуску Maven перевіряє, чи файл конфігурації містить усі пропуски даних і чи всі дані записані синтаксично правильно.

Мінімальна конфігурація включає версії файлу конфігурації, назву проекту та версію. За допомогою pom.xml налаштовуються залежності від

інших проектів, окремі фази процесу побудови проекту (buildprocess), список плагінів, які реалізують порядок складання.

Великі проекти можна розділити на кілька модулів або підпроектів, кожен зі своїм власним POM. Операції модуля можна запускати через спільний кореневий POM за допомогою однієї команди.

Файли POM підпроекту можуть успадковувати конфігурацію від інших файлів конфігурації. При цьому всі файли конфігурації за замовчуванням обов'язково успадковуються від файлу «Super POM». Super POM забезпечує конфігурацію за замовчуванням, таку як стандартна структура каталогів, яка використовується плагінами за замовчуванням, прив’язана до фаз життєвого циклу.

Maven підтримує принцип угоди конфігурації, який полягає в тому, що розглядувані аспекти вимагають конфігурації тоді і тільки якщо цей аспект не задовольняє певну специфікацію. Як наслідок, це дозволяє зменшити обсяг необхідної конфігурації без втрати гнучкості. Одним із наслідків застосування цього принципу є те, що немає необхідності явно вказувати шляхи до файлів, що спрощує вміст pom.xml. Однак майже всі стандарти, на які спирається Maven, можна змінити за допомогою спеціальної конфігурації.

Maven використовує принцип архетипів Maven. Архетип — це шаблонний інструмент, кожен з яких визначається зразком або моделлю, на основі якої створюються похідні.

Стандартна структура каталогу є однією з реалізацій принципу архетипу Maven. Наступна структура показує найважливіші каталоги проекту Java:

- Кореневий каталог проекту: файл pom.xml і всі наступні підкаталоги
  - src: усі вихідні файли
    - src/main: вихідні файли для продукту
      - src/main/java: вихідний текст Java
      - src/main/resources: інші файли, які використовуються під час компіляції або виконання, наприклад файли властивостей

- src/test: вихідні файли, необхідні для організації автоматизованого тестування
  - ❖ src/test/java: тестове завдання JUnit для автоматизованого тестування
- target: усі файли Maven, створені під час роботи
  - target/classes: скомпільовані класи Java

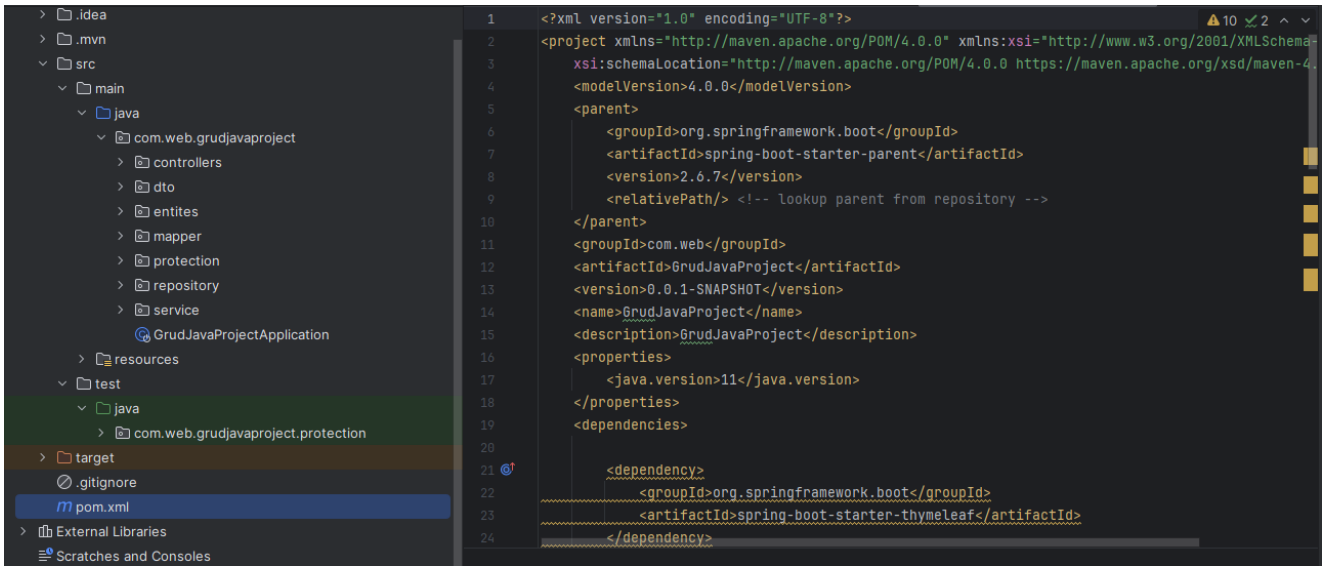

Рис. 3.1.3 Інструмен автоматизації Maven в дії

Життєвий цикл проекту maven — це список іменованих фаз, який визначає порядок дій під час його побудови. Життєвий цикл Maven містить три незалежних порядку виконання:

• clean–життєвий цикл очищення проекту. Містить наступні фази:

1. попередньо очистити
2. чистий
3. дочистка

- default – основний життєвий цикл, що містить такі фази:

1. validate – виконується перевірка, чи структура проекту є повною та правильною.
2. generate-sources
3. process-sources
4. generate-resources
5. process-resources

6. compile − компілюються вихідні тексти.
7. process-test-sources
8. process-test-resources
9. test-compile
10. test− зібраний код тестується заздалегідь підготовленим набором тестів.
11. package − Упаковка відкомпільованих класів та інших ресурсів. Наприклад, JAR-файл.
12. integration-test − програмне забезпечення в цілому або його великі модулі піддаються інтеграційному тестуванню. Перевіряється взаємодія між складовими частинами програмного продукту.
13. install − установка програмного забезпечення на локальний Maven-репозиторій, щоб зробити його доступним для інших проектів поточного користувача.
14. deploy − стабільна версія програмного забезпечення поширюється на віддалений Maven-репозиторій, щоб зробити його доступним для інших користувачів.

- site − життєвий цикл створення проектної документації. Складається з фаз:

1. pre-site
2. site
3. post-site
4. site-deploy

Стандартні життєві цикли можна доповнити функціональністю за допомогою плагінів Maven. Плагіни дозволяють вставляти нові кроки в стандартний цикл (наприклад, розгортати на сервері додатків) або розширювати існуючі кроки.

3.1.4 HTML на основі Thymeleaf

HTML (HyperTextMarkupLanguage) є основним будівельним блоком Інтернету. Він визначає зміст і структуру веб-контенту. Технології, відмінні

від HTML, зазвичай використовуються для опису вигляду/відчуття (CSS) або функціональності/поведінки (JavaScript) веб-сторінки.

Гіпертекст («гіпертекст») стосується посилань, які з'єднують веб-сторінки одна з одною або в межах одного веб-сайту чи між веб-сайтами. Посилання є фундаментальним аспектом Інтернету. Завантажуючи вміст в Інтернет і посилаючи його на сторінки, створені іншими людьми, ви стаєте активним учасником Всесвітньої мережі.

HTML використовує розмітку для відображення тексту, зображень та іншого вмісту у веб-браузері. Розмітка HTML включає спеціальні «елементи», такі як <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , , < img>, <aside>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <output>, <progress>, <video> та багато інших.

Елемент HTML відрізняється від іншого тексту в документі за допомогою «тегів», які складаються з імені елемента, оточеного «<» і «>». Назва елемента всередині тегу не чутлива до регістру. Тобто воно може бути написано з великої чи малої літери, або змішано. Наприклад, тег <title> можна записати як <Title>, <TITLE> або будь-яким іншим способом.

Thymeleaf — це сучасний механізм шаблонів Java на стороні сервера для веб-середовища та офлайн.

Основна мета Thymeleaf — створити елегантні, природні шаблони для вашого робочого процесу розробки — HTML, який можна правильно відображати в браузерах, а також працювати як статичні прототипи, забезпечуючи тіснішу співпрацю в командах розробників.

Завдяки модулям для SpringFramework, безлічі інтеграцій із вашими улюбленими інструментами та можливістю підключати власні функції, Thymeleaf ідеально підходить для сучасної веб-розробки HTML5 JVM, хоча він може зробити набагато більше.

Шаблони HTML, написані в Thymeleaf, все ще виглядають і працюють як HTML, дозволяючи фактичним шаблонам, які запускаються у вашій програмі, продовжувати працювати як корисні артефакти дизайну.

```
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
@Column(name = "card_id")
private Long id;
private String name;
private String password;

@ManyToOne
@JoinColumn(name = "person_id")
private PersonEntity person;
```

Рис. 3.1.4 Приклад дизайну

3.1.5 Функція EncryptMessage

Функція EncryptMessage шифрує повідомлення для забезпечення конфіденційності. EncryptMessage дозволяє програмі вибирати серед криптографічних алгоритмів, які підтримуються вибраним механізмом. Функція EncryptMessage використовує контекст безпеки, на який посилається дескриптор контексту. Деякі пакунки не містять повідомлень для шифрування чи дешифрування, а натомість надають хеш цілісності, який можна перевірити.

Ця функція доступна лише як механізм SASL.

3.1.6 Функція DecryptMessage

Функція DecryptMessage розшифровує повідомлення. Деякі пакети не шифрують і не розшифровують повідомлення, але виконують і перевіряють цілісність хешу.

Постачальник безпеки дайджесту (SSP) забезпечує конфіденційність шифрування та дешифрування повідомлень, якими обмінюються клієнт і сервер, лише як механізм SASL.

EncryptMessage і DecryptMessage можна викликати одночасно з двох різних потоків в одному контексті інтерфейсу постачальника підтримки безпеки (SSPI), якщо один потік шифрує, а інший розшифровує. Якщо шифрується більше одного потоку або розшифровується більше одного потоку, кожен потік повинен отримати унікальний контекст.

3.1.7 База даних h2

h2 - відкрита кросплатформова СУБД, повністю написана мовою Java. Незважаючи на малий розмір (трохи більше 1 МБ), h2 має наступні особливості:

- Дуже швидкий JDBC API з відкритим кодом
- Вбудований і серверний режими; бази даних на диску або в пам'яті
- Підтримка транзакцій, багатоверсійний паралелізм
- Консольна програма на основі браузера
- Зашифровані бази даних
- Повнотекстовий пошук
- Чиста Java з невеликим розміром: розмір файлу jar близько 2,5 МБ
- Драйвер ODBC

Для розробки СУБД не використовуються сторонні бібліотеки. Весь необхідний функціонал реалізований розробниками самостійно, щоб полегшити розгортання програми. Фактично вся СУБД поміщається у файл розміром 1 МБ, а весь архів разом із прикладами та вихідним кодом — 5 МБ.

СУБД дуже корисна при розробці, оскільки дозволяє створювати тестові набори даних безпосередньо в пам’яті, має розширені можливості запитів, численні функції, сумісні з великими базами даних, її легко налаштувати, а також має детальну та зрозумілу документацію. Для невеликих програм СУБД може бути використана сама по собі, тому що вона легко вбудовується в додаток.

3.1.8 Метод шифрування RSA

RSA − це криптографічний алгоритм з відкритим ключем, що базується на обчислювальній складності факторизації достатньо великих цілих чисел.

RSA є першим з алгоритмів такого типу, який придатний як для шифрування, так і для цифрового підпису. Цей алгоритм застосовується для великої кількості різних криптографічних застосунків.

Алгоритм RSA складається із чотирьох етапів:

1. Генерація ключів.
2. Шифрування даних.
3. Розшифрування даних.
4. Розповсюдження ключів.

Безпека алгоритму RSA побудована на складності факторизації великих цілих чисел. Цей алгоритм використовує два різних ключа, так відомі, як відкритий ключ (public) та секретний ключ (private), а разом вони утворюють пару ключів (keypair). Метод RSA добре тим, що відкритий ключ, яким було зашифроване повідомлення не потрібно приховувати, чи залишати його в таємниці. Цей ключ використовується лише для шифрування даних, а ось розшифрувати повідомлення можна лише використавши відповідний секретний ключ.

Генерація ключів. Для того, щоб згенерувати пару ключів, необхідно виконати дії в такому порядку:

1. Вибрати два простих числа p таq, кожне з яких приблизно 512 біт завдовжки.
2. Обчислити добуток n=p*q.
3. Обчислити функцію Ейлера φ(n)=(p-1)*(q-1).
4. Вибрати ціле число e, яке є взаємопросте із φ(n) та в межах 1<e<φ(n).
5. Використати розширений алгоритм Евкліда знайти число d, таке що, e*d=1(modφ(n)).

Число n називають модулем, а числа e таd− відкритою та секретною експонентами відповідно. Пара чисел (n, e) являються відкритою частиною

ключа, а (n,d) − таємною частиною. Після генерації ключів такі числа, як p та q можуть бути стертими, але ні в якому разі не бути розкритими.

Шифрування. Для того, щоб користувач A надіслав повідомлення M користувачу B, користувач A повинен перетворити M в ціле число m, так щоб були такі рамки $0 \leq m \leq n$ використавши узгоджений оборотний протокол, який ще відомий, як схеми доповнення. Після чого користувач A повинен обчислити зашифрований текст c, використавши відкритий ключ e користувача B за допомогою рівняння:

$$c = m^e \bmod n.$$

Таке обчислення досить легке, навіть якщо використати 500-бітні числа з використання модульного числа в ступінь. Після чого користувач A передає c користувачу B.

Розшифрування. Для розшифрування повідомлення m, користувач B повинен обчислити таку рівність:

$$m = c^d \bmod n.$$

У ході розшифрування рівності, відновлюється вхідне повідомлення:

$$c^d = (m^e)^d = m^{ed} \bmod n.$$

Виходя чи умови

$$ed = 1 \bmod \varphi(n)$$

випливає:

$ed = k * \varphi(n) + 1$ для якогось цілого числа k, отже

$$m^{ed} = m^{kx\varphi(n)+1} \bmod n.$$

Згідно з теоремою Ейлера:

$$m^{kx\varphi(n)+1} = m \bmod n$$

$$c^d = m \bmod n.$$

3.2 Практична чатина вебзастосунку для захисту паролів користувачів

3.2.1 Кодування методом RSA

За кодування відповідає така частина коду класу Protection:

```
publicStringencryptMessage(Stringmessage){
```

```
StringencodedMessage;
try {
cipher.init(Cipher.ENCRYPT_MODE, this.publicKey);

byte[] secretMessageBytes = message.getBytes(StandardCharsets.UTF_8);
byte[] encryptedMessageBytes = cipher.doFinal(secretMessageBytes);

encodedMessage = encode(encryptedMessageBytes);

    } catch (InvalidKeyException | IllegalBlockSizeException |
BadPaddingException e) {
thrownewRuntimeException(e);
    }
returnencodedMessage;
  }
```

Приклад: вхідними даними являються предсатвленні дані у лапках − "Spring", "Oliinyk", "JAVA", "JUnit4","1234567890".

Результат:

```
aJidH1Whkc5ACPK6LzDVUeAzwYQFneInKD/PZHew5CuIWfKp6dm89yA7u1a2l
cv8W3Pp7KT2E1EEGHsIZ4zIi+loW5wtNrNgcZKJ1TIwh+
FUuh3vYLRDdZfb5eq3bHj2KBhxzg6AKLTErKUsKe/qNCa9zM9tOr6CzZbcTuMO
rtufWj1AkD05agD2+c98PLyQEuflJT6p+58LbZjG5ku2Vt8dVxqoxSrcoM4WfzCM
R892bOkIxvVf+kSjcrmvTjLVwRsr4A6kh9GX69tfXakOYLdlfksT3GhYJjMeVb8O6
MgsAjJLdPMTDcYPnFqClWLKz9pqg
M7TmLaOMYv4Off+xz8FcImfvHAs81tR1wiEZJIn/h55Gsit6OmVztVhUY10PcJ+L
y1hmofmofmV+vwsW86fFCcZMNI8xo6cbIjYaB+IG50nOi0CEoVH9LKGu+h55G
sit6OmVztVhUY1QPQ4t/AUnb7QYVrzbo8PtdsyubTlV03gh6nihJ2xGxnJZunXML2
yPUzzcNt4uHsI6EmHen4cD9wF2XvY/KpPNxfwUtVRLG++qNTPbOrovKRJ5btxC
mG+R59rRnplQ59StDD6chzosFKx28jSJb/Uwqf0MedNhDrEkRLR6/IH4RTYqA8I
6cJey8NLpzDAc20GLpiPZA9YeLQU466cQ
```

VxHHlzAabvTkHDvWybtoU/x/rQ2Ro8153bI524rR7UKza5X5FRmm+MZtFwxRP7
39DMNURmBiIQEbbP8qHbOFMuFgGSicznSP3ZhiiWUFZ3XAuEdZ6w6H6ksolN
aqfPiBK43oiZ9O2/s5WF+DoyhwZO1UXMMoL3ReKx1f42NCl3ZZTMuyAjUcxxJx
npLQ/scQYZLVPLKkprierNZWumT/zxGcGW94zbSkL8lz4caWAAeg65IJJVgDiXPo
vs3opWGbIaBFKI/XenqHO6kNPq3pQ4Yz+CIM1mxO6Km8tuVsRbS8b+x2ZwvXQ
h1bKzKO2bFi47QzeJAhB0XR5UjaAvxy0g
IqyTRGi75dGe7RZB4AKPIwIJimv/RTMgQULJVI8hC8f0nIScHEd9AQzDDU2+
kZCWNw465w391Tl7/mRCeCEan3NjkCtpDC0zJPkVO4MylGwEAo150q9GPmz7
LuRfoqgYHJ2Y1Iv8xgdXFIcvQ+cSCr+M/XgKwqKBUOgGi/mIZE+uT55Fl0VPu7
HzyJ1hznMr9s3yf1JOMr9TAmLVlw47QqG+k3MsrLp3lBP70ahXpg2EpdddNdED
X7S66kPMq5TtJdcKSaf58tVXgtu8d6MNXrfrd267miU5vOvGCnSIP5AVbmL49A4
PuBC1cXhPGsUSdjDqqG4fKKgrmJcIf1fw
QoPnHCe9WVbanuhsVKjAluKIyBQOltTO+pOT0wrmEO8tvWrM+uHuEYyPFoK
cRfL9ksV0G8aREsOtpPx7b8ZH3XDh53gJB24GJeUw+rRH6TqyAlPb/Gz6bEgYZd
oVueZRhoOcn07HL6tuiuyAU6po7TI2JGaEOF/QY7NkmyazzOntSlTNEkOkpzbhB
WXWohfGPZKA0ENZOrbyH4NgkO1D30NwjA6Y+Lrw4mTQgsr1ziAiLNjsop4Z6i
9bfF07f5SGtWLTOlqzc4k3WSU1lEeVVHtLyiasbka0Gc6vcwAst85YeJJY/TU7x8/C
pvW99xYVAz2ftlKOKItiLd1kYK9ZbnXg

3.2.2 Декодування методом RSA

За декодування відповідає представлена нижче частина коду:

```
publicStringdecryptMessage(StringencodedMessage) {
StringdecryptedMessage;
try {
cipher.init(Cipher.DECRYPT_MODE, this.privateKey);

byte[] decryptedMessageBytes = cipher.doFinal(decode(encodedMessage));
decryptedMessage = newString(decryptedMessageBytes,
StandardCharsets.UTF_8);
```

```
    } catch (InvalidKeyException | IllegalBlockSizeException |
BadPaddingException e) {
thrownewRuntimeException(e);
    }
returndecryptedMessage;
  }
```

3.2.3 Зміна вхідних та вихідних даних

При кодуванні дані необхідно перевести із байта в строку, саме за це відповідає приведена нижче частина:

```
privateStringencode(byte[] data){
return Base64.getEncoder().withoutPadding().encodeToString(data);
```

Знову ж таки, при декодуванні дані вже потрібно перевести із строки в байт, для того, щобми могли їх переглянути:

```
privatebyte[] decode(Stringdata){
return Base64.getDecoder().decode(data);
```

3.2.4 Вхід та реєстрація у вебзастосунку

Для того, щоб зайти в локальну мережу, спочатку необхідно запустити сам проект. Запуск проекта показаний на рисунках 3.2.4.1, рисунку 3.2.4.2 та рисунку 3.2.4.3.

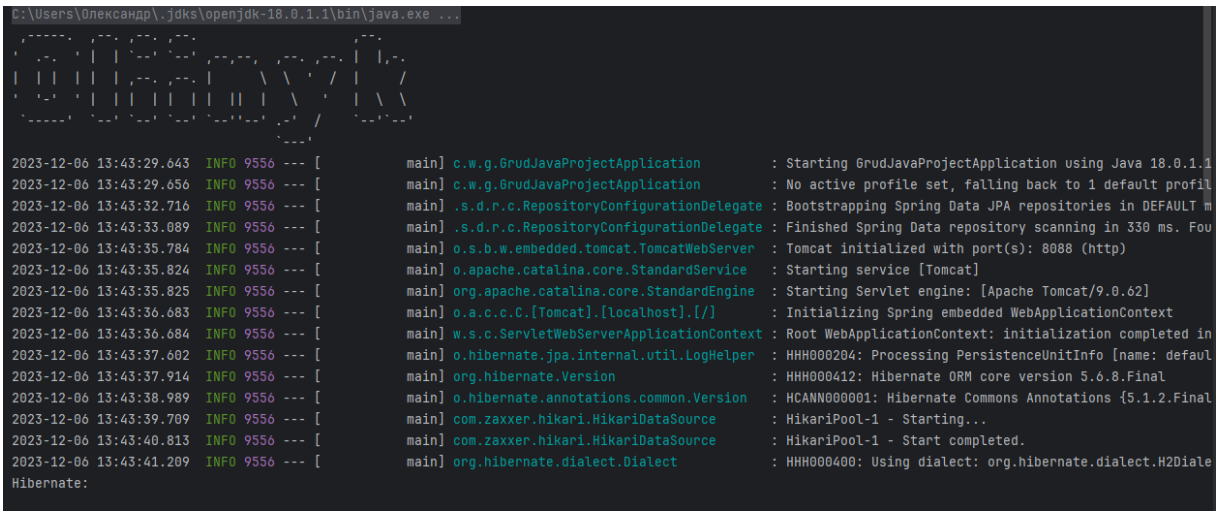

Рис. 3.2.4.1 Запуск проекта

```
    drop table if exists card CASCADE
Hibernate:

    drop table if exists people CASCADE
Hibernate:

    create table card (
       card_id bigint generated by default as identity,
        name varchar(255),
        password varchar(255),
        person_id bigint,
        primary key (card_id)
    )
Hibernate:

    create table people (
       person_id bigint generated by default as identity,
        age integer not null,
        first_name varchar(255),
        last_name varchar(255),
        login varchar(255),
        person_password varchar(999),
        phone integer not null,
```

Рис. 3.2.4.2 Запуск проекта

```
        wrong_password_count integer not null,
        primary key (person_id)
    )
Hibernate:

    alter table card
       add constraint FK1d3gys07js1m0n2et5mv3txx7
       foreign key (person_id)
       references people
2023-12-06 13:43:44.548  INFO 9556 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hiber
2023-12-06 13:43:44.589  INFO 9556 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence un
2023-12-06 13:43:49.264  WARN 9556 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefor
2023-12-06 13:43:50.786  INFO 9556 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8088 (http) with context pat
2023-12-06 13:43:50.833  INFO 9556 --- [           main] c.w.g.GrudJavaProjectApplication         : Started GrudJavaProjectApplication in 23.142 seconds (J
2023-12-06 13:43:56.705  INFO 9556 --- [nio-8088-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring DispatcherServlet 'dispatcherServle
2023-12-06 13:43:56.706  INFO 9556 --- [nio-8088-exec-1] o.s.web.servlet.DispatcherServlet        : Initializing Servlet 'dispatcherServlet'
2023-12-06 13:43:56.712  INFO 9556 --- [nio-8088-exec-1] o.s.web.servlet.DispatcherServlet        : Completed initialization in 4 ms
```

Рис. 3.2.4.3 Запуск проекта

Для того, щоб перейти до вводу логіну та паролю відповідає ця частина коду:

```
@GetMapping("/login")
   public String login(Model model) {
      model.addAttribute("person", new PersonDto());
      return "login";
```

}Результат показаний нижче:

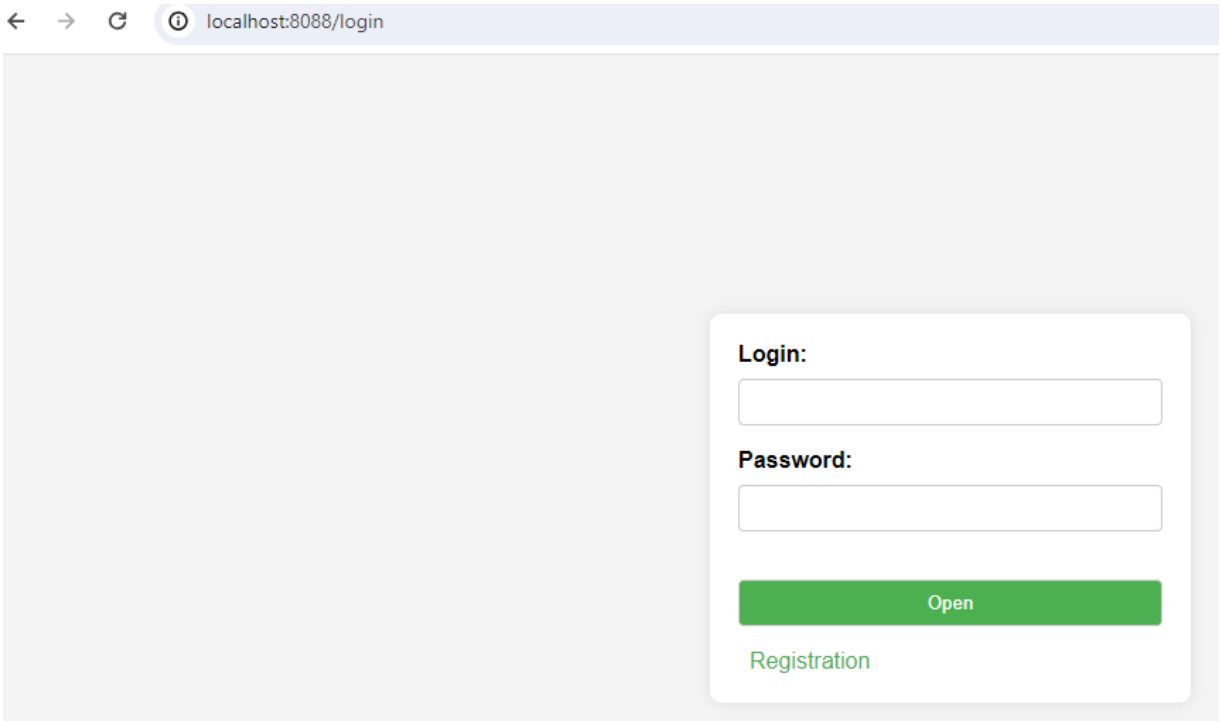

Рис. 3.2.4.4 Вхід до менеджера паролів

Для реєстрації необхідно натиснути кнопку *«Registration»* вас перенесе на сторінку реєстрації користувача. Нижче приведена частина коду, яка відповідає за дану функцію:

```
@GetMapping("/registration")
   public String registration(Model model) {
      model.addAttribute("person", new PersonDto());
      return "registration";
   }
   @PostMapping("/registration")
   public String createNewPerson(@ModelAttribute("user") PersonDto personRequest) {
      String mes = protectionComponent.encryptMessage(personRequest.getPassword());
      personRequest.setPassword(mes);
      try {
         var person = personService.savePerson(personRequest);
```

```
            return "redirect:/person/" + person.getId();
        } catch (RuntimeException ex) {
            return "emailExists";
        }
    }
```

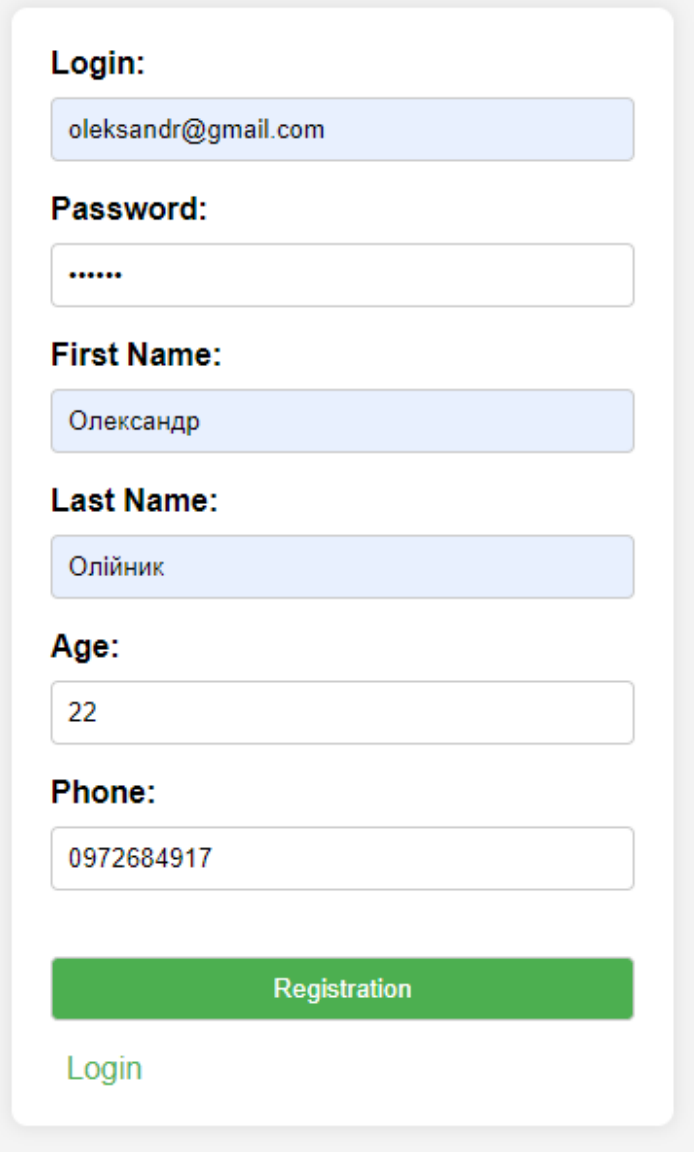

Рис. 3.2.4.5 Реєстрація користувача

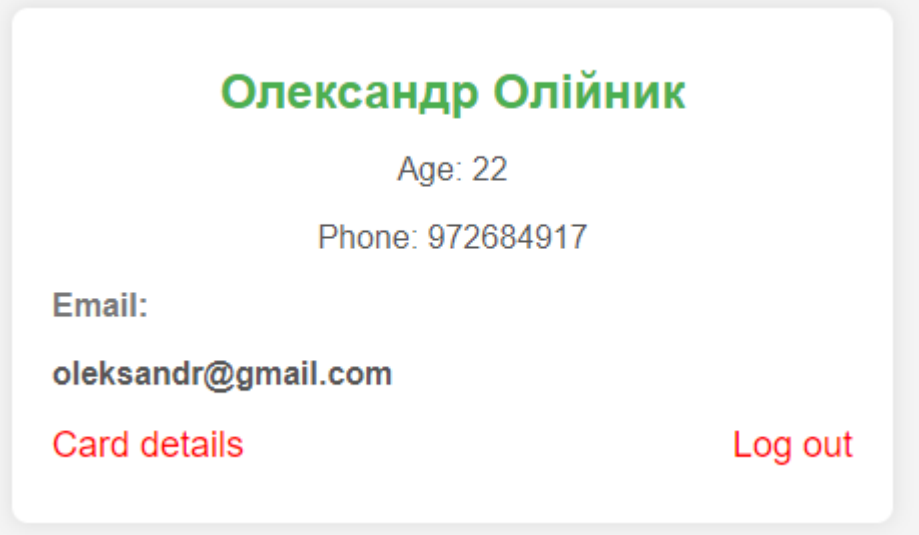

Рис. 3.2.4.6 Дані користувача при вводу вірних логіну та паролю

3.2.5 Запис даних у менеджері паролів

Для запису паролів у менеджер паролів необхідно натиснути клавішу «*Card details*», та ввести назву і пароль у відповідні рядки.

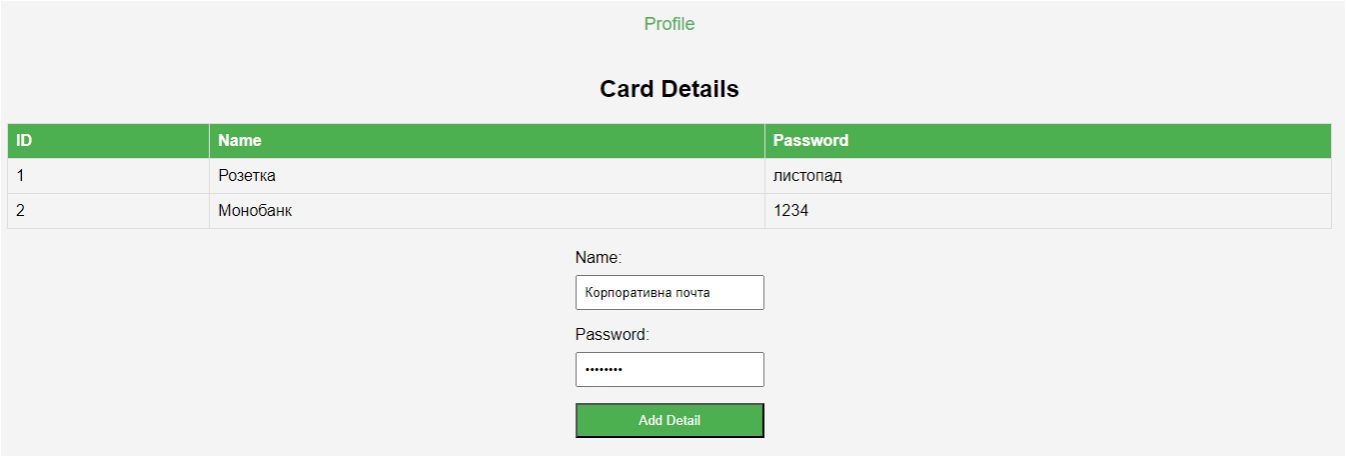

Рис. 3.2.5 Запис паролів користувача

3.2.6 Блокування профілю користувача

Для запобігання взлому профілю повним перебором була введена функція повного видалення профілю, якщо було сім спроб входу при вірному логіну та хибному паролеві.

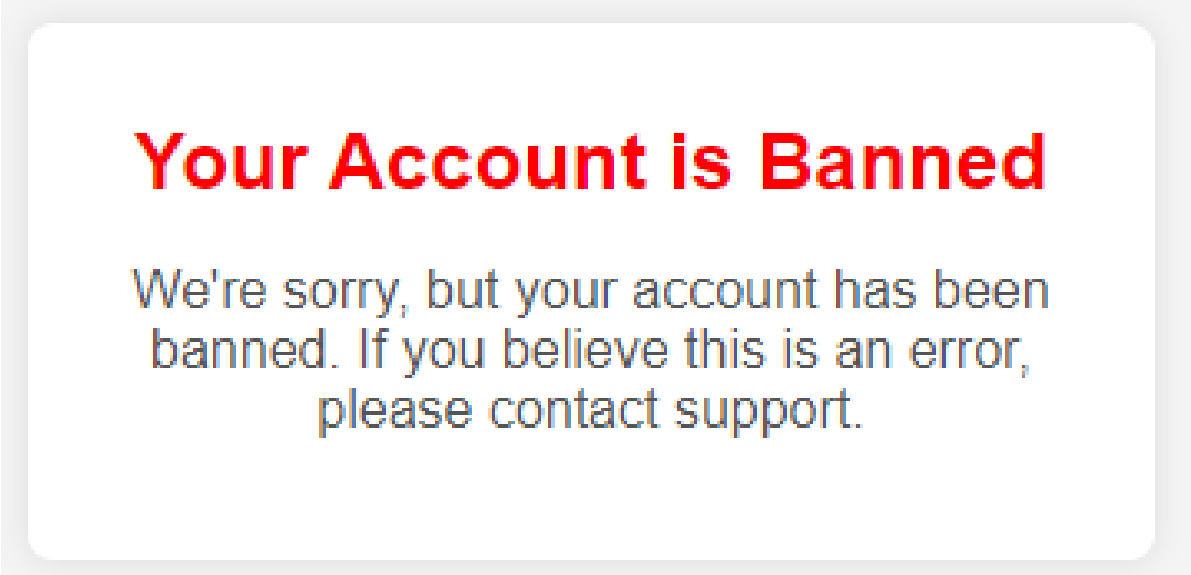

Рис. 3.2.6 Блокування профілю користувача

3.2.7 Порівняння з уже існуючим аналогом

Для простішого визначення користувача було проведене порівняння власного вебзастосунку для захисту паролів користувачів із уже існуючим аналогом Dashlane.

	Власний застосунок	Dashlane
Захист даних користувача	Криптографічний алгоритм RSA + хешування	256-бітне AES шифрування
Кількість можливих записів	Необмежена кількість	50 записів
Зручний і зрозумілий інтерфейс	+	+
Захист від атак повним перебором	Видалення акаунту користувача при правильному вводі логіна та хибному	При правильному вводі логіна та хибному вводі паролю 3 рази

	вводі паролю 7 раз поспіль	поспіль блокується можливість входу на 1 годину
Вмонтований VPN	-	+

3.3 Висновок до розділу

Під час роботи над дипломною роботою було реалізовано вебзастосунок для захисту паролів користувачів мовою програмування Java на основі криптографічного алгоритму RSA. Безпеку забезпечують база даних h2, хешування даних, та функція повного видалення профілю користувача при правильному вводі логіну та хибного вводу паролю сім разів поспіль.

РОЗДІЛ 4. ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

4.1 Антропогенний вплив на грунти

Основна діяльність людини, основана на застосуванні штучно створених засобів, приводить, в кінцевому рахунку, до різкого посилення активного втручання суспільства в хід природних процесів, яке дедалі прискорюється за темпами і зростає за силою дії. Саме тому останнім часом ми все частіше починаємо згадувати про проблеми пов’язані з екологією і безпекою нашого життя на планеті Земля.

В останні декілька років людство почало усвідомлювати величезну кількість проблем, пов’язаних з нашим впливом на навколишнє середовище. Їх можна переліковувати безкінечно; це забруднення повітря пилом та іншими частинками, забруднення води та ґрунтів пестицидами, забруднення водойм, радіоактивні відходи, парниковий ефект, озонові дірки, проблема утилізації відходів, зменшення чисельності живих організмів, демографічна криза, антропогенна зміна ландшафтів, вирубка лісів, шум, нераціональне використання природних ресурсів, хімічне та біологічне забруднення, нестача прісної води, небезпечні захворювання, епідемії тощо.

Суша – основна область зосередження життя на землі. Не дивлячись на те, що площа суходолу в чотири рази менша площі, яку займає океан, на ній зосереджено біля 57% біомаси планети. Окрім того – це головний постачальник продуктів харчування для людей і більшості тварин. Від кількості і стану придатних до використання у господарстві ґрунтів в багато чому залежить вирішення продовольчої проблеми на планеті. Тому від того настільки обережно і дбайливо ми відносимося до земельних угідь буде залежати не лише наше майбутнє, а й здоров’я та збереження багатьох видів рослин і тварин, що живуть пліч-о-пліч з нами. Але, на жаль, досвід сьогодення доводить нам зовсім протилежне.

Значні негативні зміни стану земної поверхні в цілому викликані саме нераціональними діями людей, джерело яких не лише в рівні знань, але й у погоні за економічною вигодою. Втрата плодючих земель внаслідок їх виснаження, ерозії, вилучення для промислових потреб та інших факторів призвела до загострення продовольчої проблеми на земній кулі. Потенційно оброблюваних площ на планеті є 3,2 млрд. га, з них половина вже обробляється. Причому вже 6.8% оброблюваних земель вже втрачено для сільського господарства через неправильне господарювання на них людини. Часто землі виснажуються за рахунок того, що засаджуються щорічно однією чи схожими культурами. Вже через два-три роки спостерігається збідніння таких земель на іони одних елементів і перенасичення в них інших.

Такий дисбаланс приводить до потреби внесення певних добрив і тимчасової перерви в 1-2 роки для відновлення плодючості землі. Така тактика є неправильною, бо земля в цьому випадку може втратити свою родючість. Найчастіше головним ворогом орних земель, зайнятих під сільським господарством, є внесення добрив не органічного походження, а також пестицидів (хімічні речовини, які створюються для знищення різних форм життя: бур'янів, комах та інших шкідників). Небезпечною особливістю таких речовин є те, що вони мають здатність зберігатися в природі протягом досить тривалого часу, приносячи величезну шкоду життю тварин, рослин і людині.

Вони й досі накопичуються в природі і шляхів нейтралізації їх досі не знайдено. Пестициди проходять в природі певний кругообіг, заражаючи і шкодячи всьому живому. Потрапляючи в ґрунт на значних територіях завдяки розпиленню пестициди знищують ту форму життя, для боротьби з якою їх було винайдено, потім разом з вологою та опадами вони потрапляють у ґрунт, забруднюючи його, пестициди досить швидко потрапляють у ґрунтові води, звідти у водоймища, озера, річки, криниці. Отруюючи річкову невелику річкову рибу хімікати потрапляють з прісною водою в моря і рухаються далі по харчовому ланцюжку.

Тож до людини на стіл забруднювачі потрапляють досить легко і швидко з криничною водою, з продуктами річок і морів чи з городиною, в будь-якому випадку це підриває здоров'я людини (лише за 1 рік на землі від забруднення пестицидами продуктів харчування гине до 90 млн. чоловік) та небезпечно для життя інших живих організмів. В деяких країнах вже значно обмежили використання пестицидів та інших хімічних речовин у сільському господарстві, але доки не буде винайдено альтернативні шляхи вирішення проблема залишається й досі відритою. При значних насадженнях сільськогосподарських культур єдиним поки що способом боротьби зі шкідниками є хімічні речовини, багато з яких частіше накопичуються аніж розкладаються в надрах землі або в верхньому родючому шарі.

Землю часто називають годувальницею, однак не можна стверджувати, що ставлення до неї адекватне цій назві. Якщо говорити про Україну, то за останні десятиріччя значно погіршилися показники земельного фонду. Незначний приріст продукції землеробства досягається за рахунок стійкого виснаження та деградації ґрунтів. Зростає хімічне забруднення земельних ресурсів. Продовжується значне вилучення цінних сільськогосподарських земель під промислове та інше будівництво, що становить понад 100 тис. га щорічно.

Охорону ґрунтів можна здійснювати найрізноманітнішими методами: заліснення перелогів та еродованих земель, використання раціональної агротехніки, відмова від монокультур. Технологія сільськогосподарського виробництва має базуватися на екологічно обґрунтованих раціональних нормах, виключати з обробітку землі на схилах крутістю понад 7° і інтенсифікувати використання сільськогосподарських угідь, які залишилися в обробітку. Науковці рекомендують розпочати послідовний перехід на ландшафтне землеробство, його ґрунтозахисну спрямованість, у повному обсязі виконувати протиерозійні заходи і рекультивацію земель.

ВИСНОВОК ДО ДИПЛОМНОЇ РОБОТИ

Під час виконання дипломної роботи на тему «Вебзастосунок для захисту паролів користувачів», було:

- досліджено сучасні програми захисту паролів, що надало розуміння в потрібностях користувачів;
- проаналізовано загрози менеджерам паролів, що слугувало для подальшого розроблення надійного менеджеру паролів;
- розроблено та реалізовано вебзастосунок для захисту паролів користувачів мовою програмування Java на основі криптографічного алгоритму RSA, що дозволило надійно захищати дані користувачів.

ДЖЕРЕЛА ІНФОРМАЦІЇ

1. Сторінка статті: https://web.archive.org/web/20170227140027/http://www.businessinsider.com/how-to-use-password-manager-store-protect-yourself-hackers-lastpass-1password-dashlane-2017-2
2. Стаття: The Emperor's New Password Manager: Security Analysis ofWeb-based Password Managers
3. Сайт: https://www.moyo.ua/ua/news/luchshiy_menedzher_paroley_top_10_programm_dlya_pk_i_smartfonov.html
4. Сторінка статті: https://itech.co.ua/novyny/menedzhery-paroliv-najkrashhyj-zaxyst-personalnoyi-informaciyi-v-interneti/
5. Прайс, Роб. "Менеджери паролів - це важливий спосіб захиститися від хакерів - ось як вони працюють".
6. Сайт Вікіпедія: https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D0%BD%D0%B5%D0%B4%D0%B6%D0%B5%D1%80_%D0%BF%D0%B0%D1%80%D0%BE%D0%BB%D1%96%D0%B2
7. Стаття: LessPass, менеджер паролів без громадянства https://lesspass.com
8. Стаття: Кокс, Джозеф. "Веб-сайти, припиніть блокування менеджерів паролів. Настав 2015 рік".
9. Стаття: "Менеджери паролів: атаки та захист"
10. https://web.archive.org/web/20171209221321/http://www.adambarth.com/papers/2009/adida-barth-jackson.pdf
11. Веб-сторінка: https://keeform.org/
12. Веб-сторінка: https://keepass.info/
13. Сторінка Вікіпедії: https://uk.wikipedia.org/wiki/KeePass
14. McNulty, S. Manage passwords, with 1Password. — Pearson Education.
15. Веб-сторінка: https://1password.com/ru/

16. Веб-сторінка: https://chrome.google.com/webstore/detail/dashlane-password-manager/fdjamakpfbbddfjaooikfcpapjohcfmg
17. Сайт: https://www.dashlane.com/
18. Сайт: https://it.wikiup.wiki/wiki/SQL_injection
19. Сайт: https://uk.upwiki.one/wiki/Phishing
20. Стаття: https://www.securitylab.ua/blog/company/PandaSecurity/351898.php
21. Сайт: https://uk.wikipedia.org/wiki/Keylogger
22. Стаття: https://owasp.org/www-community/attacks/csrf
23. Стаття: https://github.com/h2database/h2database
24. Стаття: https://maven.apache.org/
25. Стаття: https://uk.wikipedia.org/wiki/RSA
26. Олійник О. О. Вебзастосунок для захисту паролів користувачів – Кваліфікаційна робота здобувача освітнього рівня «Бакалавр». – Національний авіаційний університет. – Київ 2022
27. Стаття: https://osvita.ua/vnz/reports/ecology/21058/