

**Софийски университет „Св. Кл. Охридски”**

Факултет по математика и информатика

специалност : „Защита на информацията в компютърните системи и мрежи“

**Дисциплина: Моделиране на защитени взаимодействия в компютърните системи**

**Курсова работа на тема “Описание на система за съхранение, проверка и генериране на сигурни пароли”**

*Автор:*

Иван Ивов Чучулски, фак. номер: 4MI3400043

*Ръководител:*

проф. д.т.н. Веселин Целков

летен семестър, 2021/2022г.

Съдържание

[**1.** **Въведение** 3](#_Toc106743936)

[**2.** **Описание на системата** 3](#_Toc106743937)

[**3.** **Описание на основните алгоритми в системата** 5](#_Toc106743938)

[**3.1** **Основни функционални възможности на потребителите** 5](#_Toc106743939)

[**3.2** **Регистрация и влизане в системата** 6](#_Toc106743940)

[**3.3** **Процедура по добавяне на данни за автентикация** 6](#_Toc106743941)

[**3.4** **Процедура по генериране данни за автентикация** 7](#_Toc106743942)

[**3.5** **Процедура по проверяване сигурността на данни за автентикация** 7](#_Toc106743943)

[**3.6** **Процедура по извличане на данни за автентикация** 7](#_Toc106743944)

[**3.7** **Процедура по премахване на данни за автентикация** 7](#_Toc106743945)

# **Въведение**

Целта на проекта е да бъдат формално описани взаимодействията в система за съхранение, проверка и генериране на сигурни пароли. С помощта на граф-схеми, е-мрежи ще бъдат представени основните функционални характеристики на системата и ще бъдат проверени условията за пълнота и непротиворечивост на алгоритмите.

Системата представлява клиент-сървър приложение, което позволява на потребителите да записват пароли за използвани от тях уеб сайтове или други услуги. Приложението дава възможност за генериране на сигурна парола за конкретна услуга, ако потребителите не желаят да измислят такава.

Клиентската част на системата ще поддържа два варианта – десктоп приложение с графичен потребителски интерфейс, което може да се изпълнява на всички основни операционни системи за персонални компютри, а вторият вариант на системата е с конзолен интерфейс, който изисква от потребителя въвеждане на текстови команди.

# **Описание на системата**

При съхранението на чувствителна информация като пароли за достъп е необходимо да се предприемат мерки за защита от неоторизиран достъп, модификация или кражба. За целта ще използваме концепциите за криптографска хеш функция и симетрично криптиране на данните.

Паролите, които потребителите подават при регистрация в системата ще преобразуваме с помощта на криптографска хеш функция и ще пазим в базата от данни резултата от пресмятането, който е поредица от байтовете на техния хеш. При опит за влизане в системата се пресмята хеша на въведената от потребителя парола и за успешен вход е необходимо той да съвпада с този в базата данни. Допълнителна сигурност ще ни донесе използването на т.нар. salt – това е поредица от байтове, която произволно генерираме при регистрация, долепяме към подадената парола и така пресмятаме хеша. По този начин се предпазваме от варианта, в който паролите за достъп на потребителите съвпадат и резултата от хеш функцията ще е еднакъв, а също така и на практика обезсмисляме много от brute-force атаките и атаките с rainbow таблици. Ще използваме 16-байтов salt и криптографската хеш функцията SHA-512.



Фигура 1 Хеширане на парола със salt

Тъй като приложението цели да предостави възможността на потребителите да могат да записват пароли за използвани от тях услуги е нужно записаната информация да бъде записана достатъчно сигурно, но да може да възстановена в явен вид. Това се постига с използването на симетрична криптографска функция, която използва един частен ключ, който трябва да остане таен за криптиране и декриптиране на информацията. За генерирането на този частен ключ ще изискваме от потребителя т.нар. “master” парола, която той трябва да въведе при регистрация. При нужда от добавяне или генериране на парола е необходимо потребителят да въведе своята “master” парола отново, като това добавя още едно ниво на сигурност и автентикация след влизането в системата.



Фигура 2. Симетрично криптиране

За образуването на частния ключ ще използваме алгоритъма “PBKDF2”, който дава възможност да добавим и случайно генерирана последователност от байтове(salt на ключа), тъй като отново съществува възможността “master” паролите на някои потребителите да съвпадат, както и правим по-трудни атака за отгатване на ключа, знаейки само “master” паролата. За самото криптиране използваме алгоритъма AES в режим CBC, като той изисква освен текста за криптиране да бъде подадена и случайно генерирана поредица от байтове, наречена инициализационен вектор(IV), която служи за подравняване на данните и увеличава силата на криптирането. В базата данни записваме криптирания текст, байтовете на salt-та за ключа и байтовете на инициализационен вектор.

При декриптиране потребителят въвежда своята “master” парола, която заедно с байтовете на salt-та за ключа формират тайния ключ. После ключа и инициализационен вектор участват в декриптирането на получаването на паролата в явен вид.

# **Описание на основните алгоритми в системата**

## **Основни функционални възможности на потребителите**

Потребителите в системата могат да изпълняват следните основни действия:

* Вписване(регистрация) в системата
* Заявяване на достъп до системата(Логин)
* Получаване на отговор от системата
* Анализ и визуализация на отговора
* Проверка на наличните функционалности на приложението(Помощ)
* Въвеждане на данни за автентикация в услуга без проверка на тяхната сигурност
* Въвеждане на данни за автентикация в услуга с проверка на тяхната сигурност
* Въвеждане на данни за автентикация в услуга и генерация на парола
* Проверка сигурността на данни за автентикация
* Изход от системата

## **Регистрация и влизане в системата**

## **Процедура по добавяне на данни за автентикация**

Алгоритъмът за добавяне на парола е последователен.

Основните състояния на заявка по добавяне на парола са:

* s1: постъпила е нова заявка
* s2: заявката е за добавяне на данни за автентикация
* s3: съобщение за грешка
* s4: потребителят има валидна сесия
* s4: master паролата e успешно извлечена от базата данни
* s5: въведената парола е криптирането успешно
* s6: успешно добавяне на новите данни в базата данни
* s7: заявката е обработена

Основните събития, извършващи преходи и промяна състоянието на заявка по добавяне на парола са:

* a1: постъпва заявка за добавяне на парола
* a2: сесията на потребителя е изтекла
* a3: параметрите на заявката са невалидни
* a4: извличане на записаната master парола
* a5: извършване на криптиране на паролата
* a6: добавяне на новите данни за автентикация в базата данни
* a7: записване на потребителското действие в системния дневник
* a8: връщане на отговор и преминаване към следваща заявка

Таблица на състоянията и преходите при процедура за добавяне на данни за автентикация

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a1 | a2 | a3 | a4 | a5 | a6 | a7 | a8 |
| s1 | s2 |  |  |  |  |  |  |  |
| s2 |  | s3 |  |  |  |  |  |  |
| s3 |  |  |  |  |  |  | s7 |  |
| s4 |  |  |  |  | s5 |  |  |  |
| s5 |  |  |  |  |  | s6 |  |  |
| s6 |  |  |  |  |  |  |  |  |
| s7 |  |  |  |  |  |  |  |  |

## **Процедура по генериране данни за автентикация**

## **Процедура по проверяване сигурността на данни за автентикация**

## **Процедура по извличане на данни за автентикация**

## **Процедура по премахване на данни за автентикация**