LETTERKENNY INSTITUTE OF TECHNOLOGY

ASSIGNMENT COVER SHEET

To Be Completed by the Student

Lecturer’s Name: Maria Griffin

Assessment Title: Lab Books

Submission Date: 00/00/0000

Student’s Name: \_\_Aidan dullaghan\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Course / Stage \_\_\_\_3rd year\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Subject/Module: Secure Programming

Word Limit: N/A Actual Word Count: N/A

I confirm that the work submitted has been produced solely through my own efforts.

Student’s signature: Aidan dullaghan Date: \_\_10/05/2022\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |
| --- |
| **Note**: **PENALTIES**   * The total marks available for an assessment will be reduced by 15% for work submitted up to one week late. The total marks available are reduced by 30% for work up to two weeks late. * Assessment work received more than two weeks late, without prior approval by the lecturer will receive a mark of zero. * Marks awarded will be reduced by 10 % if submitted work is greater than 10% above or below the assigned word limit. * A further hard or electronic copy of your submitted work may be requested, and therefore you must keep a copy on disc. * Incidents of alleged plagiarism and cheating are dealt with in accordance with the Institute’s Assessment Regulations   **Plagiarism:** Presenting the ideas, words of someone else without proper acknowledgement. Refer to the Institutes’ procedures and guidelines for the assessment of learners. |

Salt Lab Report

Aidan Dullaghan

L00155144

Submitted 13/05/2022

Contents

[Introduction 2](#_Toc103362791)

[Aims/Objectives 2](#_Toc103362792)

[Method 2](#_Toc103362793)

[Results and Testing 3](#_Toc103362794)

[Conclusion 4](#_Toc103362795)

[Appendices 5](#_Toc103362796)

[References 5](#_Toc103362797)

## Introduction

We'll look at salting in this lab, which is a method that may be used by any programming language. This exercise will demonstrate how salting is performed in Java code to add more security to a user's password, making it more difficult for a hacker to find the password. Before each password is hashed, a unique, random string of characters known only to the site is added to it. Typically, this "salt" is placed in front of each password.

# Aims/Objectives

* Complete the code in generate salt method
* Complete the code in the main () method
* Use salt for a securer password

The aim of this lab is to create a java project that uses salt to protect user’s passwords from potential hackers.

## Method

1. Creating the project

So, to start this lab I had to create a project called PasswordEncryptionService.java in eclipse editor how I done this was I opened eclipse and clicked file in the top right hand of the screen then clicked new and went for java project once in here this was where I named the project. Everything you can see below.

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

1. Writing the code

Most of the code was provided again for me on blackboard so I just copied and pasted it into my project PasswordEncryptionService.java there were two methods that needed to be coded for the salt to work they were the generate salt () method and generate main () method. The code used here was to Generate an 8-byte (64 bit) salt as recommended by RSA PKCS5. I then had to create strings for password and attempted password and be able to print out the password, the attempted password, the salt and whether they matched or not. I will show this in my results and testing section.

## Results and Testing

This part will describe the testing portion of the program, as well as the findings that will be provided. I'll go over the proven facts and provide screenshots to back up my statements.

My first job was to code the generate salt method and the main methods of the program for the program to work. For the generate salt method it was important to use Secure Random instead of just Random because a random class has only 48 bits whereas secure Random can have up to 128 bits so the probability of repeating in secure Random are smaller. As you can see below this is my generate salt method using Secure Random.

A screenshot of a computer

Description automatically generated

My second task was to complete the main method by creating strings for password and attempted password put in methods for salt and encrypted password the idea of this method is to create a salted password from a random string of passwords.

Graphical user interface, text

Description automatically generated

As you can see above this is the code that create the salted password also it encrypts it as well, I will show the results below when I run the following code.

Graphical user interface, text

Description automatically generateds

As you can see from the results it prints out password value, Attempted password, salt, the salted password and if the passwords match as you can see the password under salt is generated from randoms strings. Then once the password is salted it is changed to keep from hackers accessing it on the database.

## Conclusion

In my conclusion I felt that the lab went well and performed the task that it was meant to perform I learned a lot about salting regards to password security protecting it from hackers that I never knew before. Also, I learned how to code the salt into the programme, which is a good skill to have learnt, and we seen when the programme was run how the salt worked and how secure it was so I enjoyed doing this lab and working through it.

# References

https://www.geeksforgeeks.org/random-vs-secure-random-numbers-java/#:~:text=Size%3A%20A%20Random%20class%20has,which%20the%20seed%20was%20generated.