Web-Based Evaluations

Security Document

Group Number: 3

Class CPSC 488 Section 1

4/25/2022

Tanuj Rane txr1029@sru.edu

Dalton Stenzel drs1030@sru.edu

Logan Racer lsr1006@sru.edu

Anthony Cinicola [ajc1033@sru.edu](mailto:ajc1033@sru.edu)

Duncan Lawrence [dal1017@sru.edu](mailto:dal1017@sru.edu) duncanlawrence2000@gmail.com

Contents

[Introduction 3](#_Toc101789934)

[Login 3](#_Toc101789935)

[Passwords 3](#_Toc101789936)

[Role Division 4](#_Toc101789937)

[Exception Catching 4](#_Toc101789938)

[Flaws 4](#_Toc101789939)

## Introduction

The security manual for this specific software is designed to cover the key points of how the system either is or should be protected from situations such as errors and hacking. Each section below will cover a particular topic of security threats pertaining to the software and how they’re resolved. The very last section will cover flaws in our system that need to be taken into consideration for the next set of hands that are determined to pick up where we left off.

## Login

First topic to be discussed in the action of logging in. When a new user is created, a temporary password is created on their behalf from a user administrator. Once the user is informed of their temporary password, they will be prompted to reset their password upon logging in for the very first time. In order to prevent mistakes, the “First Login Screen” will provide two password boxes that, by default, hide the characters being typed into the boxes. The boxes must contain exactly matching passwords.

This “First Time Login” section is integrated into the system as a means to protect newly created users from having a password created by someone else. In the case of a “user upload file”, a file used to create new users upon being uploaded, being leaked, the passwords are partially protected as those files exclusively contain hashed versions of the temporary passwords used for users. As long as the file doesn’t use the same hashed password for all the users, then the users will be protected from having their accounts stolen in such a way.

## Activity Logging

The ability to obtain log files relating to information done from within the system is accessible through the “ADMIN” role from any page in the navigation bar. The logging capabilities of the program include, but aren’t limited to: users who log in, user account changes, rows not added from a user add file, group creations, group edits, when an evaluation is started/attempted/finished, uploading an evaluation file with or without warn/issues, etc.

## Passwords

Continuing the topic of login details, passwords are protected via the use of the BCrypt hashing function. A hashing function can be described as a function used to produce coded value used as an index from an arbitrary value. The index value serves to be the value checked when a password is entered. Upon entering a password to login, the password is checked with the password stored for the user email that is being used to login in. If the user-entered password correctly points to the same index where the hashed password is, then it is a match and the user is logged in, otherwise a “login failed”-esque message displays. When editing, resetting, or manually adding a user, any password submitted must be five or more characters, where spaces aren’t allowed nor counted. Failure to make the password at least five characters long with no spaces will result in feedback mentioning the failure to submit the information. The only exception to adding small passwords is in when a user is added via the “upload users” ability where there isn’t currently a way to check the number of characters provided as the password in the file is a hashed variant of the password so retrieving the original password’s length is impractical as of now. This measure is preferred as a smaller temporary password that is protected via a hash function is a lot safer than having a user’s raw temporary password located somewhere in an unprotected format.

## Role Division

When it comes to security, you don’t want to put all your eggs in a single basket. So instead of relying on the “ADMIN” role exclusively for every sort of task relating to users, groups, and evaluation forms, there is a separate role for handling the groups and evaluation forms while the “ADMIN” group handles users. The role for handling groups and evaluation forms is the “EVAL\_ADMIN” role. Having this disconnect allows for less users to have the ability to do significant damage to the system. An “EVAL\_ADMIN” with compromised credentials won’t allow for the users to be altered much other than their groups and forms. The opposite can’t be said of an “ADMIN” account that becomes compromised as they are capable of creating users of any type of role, including an “EVAL\_ADMIN” role with the current setup.

## Exception Catching

As with any piece of software, errors may and likely will occur. Having such a problem can lead to unintentional backdoors, data leaking, or something of the sort. Regardless, taking action to prevent errors, faults, and failures are necessary to avoid costly and irreparable damage. Exception catching is a tool used to help prevent crashes for a system. While it hasn’t been implemented for every single method in each class there are some instances of its use in methods that require computation. One such example is in the pageCalc() method in the AdminMethodsService class in order to catch any weird errors that might occur if something like an index is amiss.

## Flaws

Time constraints and limited knowledge has led to some shortcuts and unfinished business left unattended. Notable design flaws with our program include a lack of an “ADMIN-LOGS” role in order to view and control the logging being done, SQL injection tests, strong password requirements, and power control with “ADMIN” users since they can create any type of users.