

CO3201 Computer Science Project

Interim Report

Avoiding Shoulder Surfing via Graphical Passwords

Adam Barnes

School of Computing and Mathematical Sciences, University of Leicester

November 2021

Table of Contents

0.1 Aims and Objectives.....	1
0.2 Survey of Literature and Information Sources.....	2
0.3 Requirements, Specification and Design	4
0.4 Methodology, Planning and Timescales.....	7
0.5 References	10


DECLARATION

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s).

Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s).

I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: Adam Barnes

Signed: 

Date: 15/10/2021

0.1 Aims and Objectives

In the last decade, further authentication has become increasingly important with the risks surrounding simple, textual passwords growing. Different concepts have been designed to make authentication more secure, for example two-step verification and CAPTCHAs, which can help verify devices and even stop automated bots from brute-forcing entry. What hasn't been widely considered is the security risk that shoulder surfers can bring to unknowing users. This ability to look over the shoulder at the user input means that the password could be compromised if it is a textual password, for example pressing keys on a keyboard.

To obfuscate the password mechanism from onlookers, a part of the community is exploring the use of image and colour recognition to create "graphical passwords". There are many different concepts already in circulation, some of which focus on image recognition for a series of images, and others which use more of an abstract approach like free-drawing comparison [1], convex hulls [2] and more as mentioned in my literary review [3] [4]. However, not all concepts have shoulder-surfing at the forefront of their design there has been little comparison between concepts in terms of their security from onlookers.

My main aim for this project is to evaluate the effectiveness of different graphical password approaches, that have been documented, based on their ability to avoid shoulder surfing.

To target this aim I should research the different methods and implement these myself into a piece of software. I should apply evaluation tools to the software so the user can see testing data of their selected method, and this can be compared to the other methods that I have chosen. The evaluation should consider other aspects as well as shoulder surfing; failure to do so would allow some concepts to seem better than others even though they have not taken usability or quality into account.

My **secondary aim** is to allow users to select & modify a graphical password method and perform a trial registration and log in procedure. This means the evaluation tools will have to be flexible for modifications made and maintain accurate evaluative data.

To achieve the project aim, I need to follow a series of **objectives**. Here are the significant steps I need to take:

1. Research and choose graphical passwords to implement and evaluate
2. Create software to host these implementations and display them as intended
3. Allow the user to choose and modify a password and perform a *mock* login procedure
4. Allow the user to view an evaluation of the graphical password and login procedure
5. Evaluation and conclusion for the selected graphical passwords

0.2 Survey of Literature and Information Sources

Introduction: I decided to conduct research on the different literature surrounding graphical passwords and shoulder surfing. The aim of this was to improve my knowledge on user-authentication, the security risks and help conceptualise my own ideas based on previous efforts developing graphical passwords. I used Orna & Stevens [5] research model as inspiration by cycling through discovery whilst also evolving smaller questions I may have on niche parts in the topic. This aided my knowledge on authentication, the view and progress made on graphical passwords and the success of trials involving shoulder surfing.

Shoulder surfing is, at a personal level, not a huge issue. Passer-by's may look at you enter your password credentials, but in most cases, this is due to curiosity or boredom (68% of cases) and with no intent to cause harm [6]. However, in a few cases, patterns, PIN codes and passwords can be put at risk [6], as it is easy for an attacker to get hold of this plain data from user input. Users are not normally aware of being shoulder surfed [6], so rather than be unknowingly compromised it would be better to create a password that can be challenging for onlookers to distinguish.

Graphical Passwords have been discussed as a method for authentication for quite some time, with the basis of using images and colours. However, the concept is very broad, and many approaches are far from each other in terms of their usability process. Here are some ideas from papers I have explored.

Text and colour wheel: In the paper by S. K. Sonkar et al [3], the authors propose a graphical password to be efficient and shoulder surfing resistant. The user registers with a textual password and selects a favourite colour. To login the user is presented with a wheel, each wedge being designated a colour and 8 characters. To enter the password, the user spins the wheel, so the colour is on the wedge with the character they want to confirm. This is so any shoulder surfer cannot see which characters are being entered if they don't know the selected colour. However, shoulder surfers may eventually find out the users' colour, compromising the user's password. There should be another safety mechanism to make the users password more secure, as a favourite colour is easy to guess, especially if the attacker is usually nearby i.e., in an office environment. The usability can also be an issue, as the user must look through all letters and digits to find what they want. This will significantly increase with the length of the password, persuading users to go with a smaller, less secure password as a result.

Coin password: Another approach I found was "The Coin Passcode: A Shoulder-Surfing Proof Graphical Password Authentication Model for Mobile Devices" [7]. The authors explored different recognition and recall-based techniques. The user is provided with 10 "coins", all with a random colour, icon, and number. To enter their pre-determined password, they press on the coins that include the (one of three) element they need to enter next. I like this use of more than one element, as it creates a level

of abstraction from shoulder surfers, like imagery. However, after looking many times an attacker may be able to work out that you only need one element in the coin and can therefore workout your password; provided they shoulder surf regularly. This time would increase exponentially as the number of elements increase.

Grid method: S. N. M. Ariffin et al [4] created a grid method where the user had to select an image and “draw” into cells of an outlined grid. During login the user would be given the image and should have to select the same cells to gain access. Simple and easy-to-use, however the size of the grid and number of cells selected all factor into the difficulty for a shoulder surfer to solve the password.

I have looked at other methods, previously mentioned [1] [2]. I will not be able to implement all the approaches I find, so I will focus on the ones stated in this literary review. All the sources have provided a model of their concept, showing that it is in fact possible to implement what they have proposed. Some of the sources also provide data to show the effectiveness of the concept, however I don’t want to rely on this for my evaluation as there is possibility of bias affecting the results, and shoulder surfing is not a focus of consideration. I will be using my own evaluation tools and user testing, eliminating the need to choose sources to rely on. In conclusion when I am developing the software, I will be going back to research to help me with my implementations. Furthermore, I will need to expand my knowledge on shoulder surfing as I start designing my criterion.

0.3 Requirements, Specification and Design

Before deciding the factors for the software specification, it is important to set out some software **requirements** that will help achieve our objectives.

- Each graphical password must be fully implemented and functional, with the user able to:
 - Select the method
 - Register and provide all necessary data for the graphical password
 - Display the graphical password on the login page
 - Be able to login using the data given to the graphical password
- Graphical user interface to:
 - Perform seamless registration
 - Visualise each graphical password and demonstrate it as intended, with the functionality behind it
 - Allow the user to perform evaluation and see a summary
- A set of evaluation tools for each graphical password that can be accessed by the user
 - These will be decided prior to the completion of the graphical password implementations and documented in the final report
 - Should allow for easy comparison between methods

It is important to separate the functional requirements and non-functional requirements, those of which will be outlined as I start development:

Functional Requirements	Non-functional Requirements
<ul style="list-style-type: none">• User registration• Graphical password selection• Graphical password log in implementation• Evaluation Tools• Summary page• GUI for all pages and visualisation of graphical password	<ul style="list-style-type: none">• Scalability of each graphical password – a constraint• Performance of each graphical password• User password security in database• Data integrity and how data is handled after system is switched off

The software will include screens for an Introduction Page, Registration Page, Login Page and Summary Page. A database will be needed to hold the user log in data, chosen graphical password & parameters and any necessary data for the evaluation. The **sequence diagram** below simplifies the user **interaction** with the software and interaction between software components:

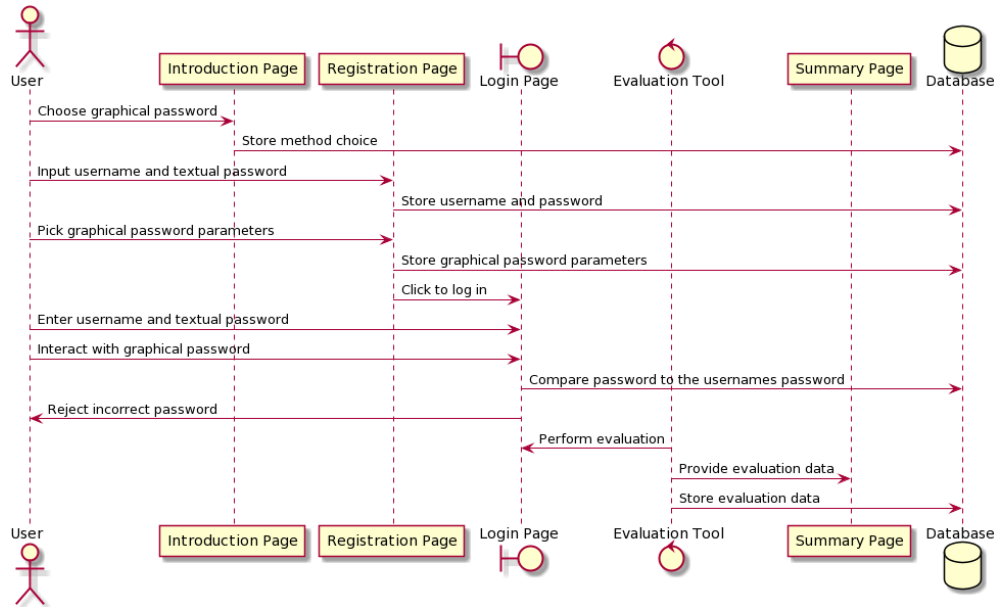


Figure 1: Sequence diagram for basic interaction that will take place between components and user.

I will be creating a desktop application to host the GUI with its graphical passwords and evaluation tools. There are many programming languages to choose, but Java will be my choice as I have had most experience with it in larger scale assignments and projects throughout my degree. There are lots of great IDEs, plugins and core packages for Java that will help make the process of designing a login program easy. The challenge starts when I introduce the implementation of graphical passwords, but my experience of Java will be key when I start running into difficulties. Python and C are alternatives; however, they provide no outstanding benefits that Java doesn't have, and I have less experience with them.

For IDEs, I needed one that could assist GUI design, and it was between NetBeans and Eclipse. While NetBeans has a powerful GUI builder, I will be using Eclipse. This is due to my experience using Eclipse, and it also contains similar GUI capabilities with a great interface for writing code. To design the interface, I will be using Swing, which has had extensive support over the years, compared to newer packages like JavaFX which is not part of the core APIs.

To store the user data such as username, textual password, and any graphical password parameters, I will connect to my project a MySQL database. Below is the architecture of how the different tables may interact with each other:

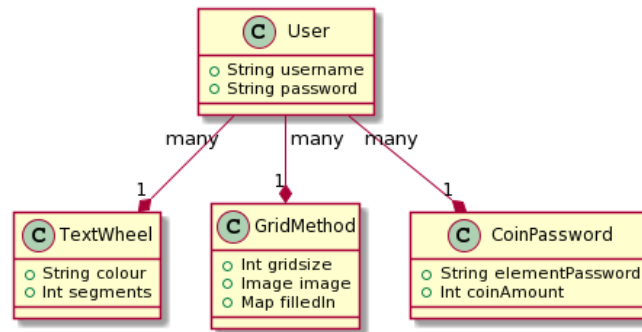


Figure 2: Simple Class Diagram

As I start to explore options for my Evaluation Tools, I will also be exploring the option of expanding the Database and the interaction between components for functions such as comparison between Graphical Passwords.

0.4 Methodology, Planning and Timescales

For this project I will be using **Scrum**, an adaptation of the Agile methodology. My reasoning for this is I will be using GitLab for my commits and project files, and this can incorporate scrum easily using issue boards, tags, milestones etc. Scrum is suitable for solo and small projects that have a degree of uncertainty, or the final destination of the project is still not entirely decided prior to development. This flexibility will help me further achieve my aim as I can make easier adjustments to my project, allowing me to better evaluate the different graphical password concepts. Using another methodology, like Waterfall, would be more rigid and at this point my requirements would already have to be set in stone. As I have no experience with designing graphical passwords or creating evaluative tools for them, Scrum will allow me to make all the little adjustments I need to reach an optimum final piece.

On GitLab I will be using Issues together with Issue boards to handle Scrum and work towards different end-user cases. On these issues I can detail exactly each step I need to take and what issues are blocking the one being viewed from being completed. To plan my deadlines in advance I have created a **Gantt Chart**. This will help me look at the timeline of the project from a glance and see how other components will be affected if another takes too long or short to complete. The task on this chart will be a short synopsis, and the details will be expanded on the corresponding GitLab issue(s). The Gantt Chart for Semester 1 and 2 can be viewed **below** and on GitLab.

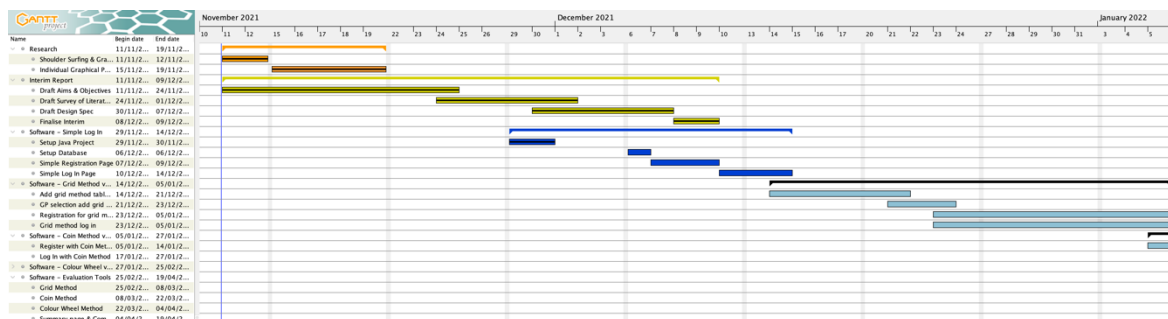


Figure 3: Semester One on Gantt Chart

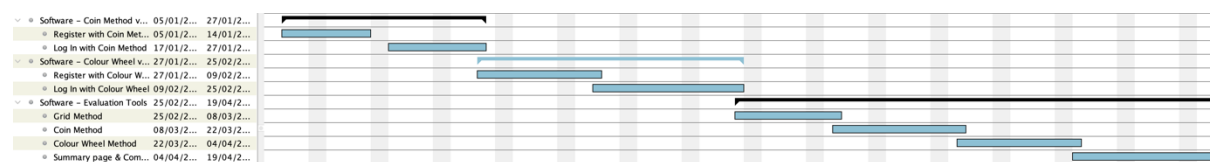


Figure 4: Semester Two on Gantt Chart

Risk Management

As with any project there are risks that can unfold, so it is important to have a process in place to help control them. C. W. Dawson [8] provides a series of steps that I can use to alleviate risks.

Identify risks: I am new to GUI development in Java Desktop applications, which is heavily important to understand as I am working with *graphical* passwords. The evaluation tools must be dependent on the graphical passwords being completed. The requirements are still indefinite and may pose risk. Apart from these technical risks, non-technical risks could include: running over time estimates, illness, or events out of my control.

Assess impact of risks: We can discover critical risks through calculating the impact of risks, using $\text{risk impact} = \text{likelihood} \times \text{consequence}$. The graphical passwords all have their own distinct nature and so we will likely have a problem with developing at least one of them. This will bring a knock-on effect to our evaluation tool dependency, so it will be important that we are prepared for this eventuality. Apart from this, Java Swing should mean we can handle the GUI easily, and the non-technical risks remain low.

Alleviate critical risks: If I have not yet started developing the graphical password (in question) and through my continued development or planning it poses a risk, I will use *avoidance* to research and select another graphical password I can implement. If I am already far in development of the graphical password and I find a problem, I will deploy a *contingency* plan. In advance I have added an extra couple days to development to allow for any delays, but in case of further delays, or an unforeseen structural error with the concept, I will make slight changes to the concept to make it easier to implement (but still work as a graphical password approach). It is likely that I will not be implementing the exact model that the sources have proposed.

Control risks: Once I complete an objective or requirement, I will analyse the risks again and see if they are any higher going forward. It is important iterate the analysis of risks, otherwise new risks could go undiscovered and cause problems further on in the project.

When creating the software, I want to specify the different working versions of the software; for example, when the user can register and log in, when each graphical password has been implemented and when the corresponding evaluation tools have been added. I will be using GitLab **iterations** to specify which issue is part of which iteration/version, and at the end of each iteration be a tested and working piece of software that can accomplish some end-user case(s). I will document each iteration up until the final iteration which be evaluated. I will be declaring **milestones** to help oversee when certain objectives will be completed.

Milestones

Completed

- **13/11/2021** – Research shoulder surfing and graphical passwords
- **20/11/2021** – Research individual graphical password approaches
- **24/11/2021** – Draft aims, objectives, and survey of literature for interim report
- **30/11/2021** – Setup Java project
- **7/12/2021** – Draft requirements, specification, and planning for interim report
- **8/12/2021** – Finalise Interim Report and complete references

Associated objectives: 1 (1, 2, 3, 5, 6), 2 (4)

Started

- **9/12/2021** – Submit Interim Report
- **14/12/2021** – Functional registration and log in using textual password with GUI

Associated objectives: 1 (7), 2 (8)

Not Started

- **5/01/2022** – Grid method [4] implementation completed
- **21/02/2022 to 25/02/2022** – Prepare for interview
- **27/01/2022** – Coin method [7] implementation completed
- **25/02/2022** – Colour wheel method [3] implementation completed
- **8/03/2022** – Grid method evaluation tools
- **22/03/2022** – Coin method evaluation tools
- **5/04/2022** – Colour wheel method evaluation tools
- **19/04/2022** – Store user evaluation and compare to pre-existing tests with same password and different method
- **4/05/2022** – Evaluation of graphical passwords and their ability to avoid shoulder surfing
- **5/05/2022** – Final software submission and dissertation report submission
- **9/05/2022** – Mini viva

0.5 References

- [1] A. J. Lin and F. Cheng, *A Free Drawing Graphical Password Scheme*, University of Kentucky, 2009.
- [2] S. Wiedenbeck, J. Waters, J. C. Birget, A. Brodskiy and N. Memon, "Design and longitudinal evaluation of a graphical password system," *International J. of Human-Computer Studies*, vol. 63, pp. 102-127, 2005.
- [3] S. K. Sonkar, R. L. Paikrao, A. Kumar and S. B. Deshmukh, "Minimizing Shoulder Surfing Attack using Text and Color based Graphical Password Scheme," *International Journal of Engineering Research & Technology (IJERT)*, vol. 3, no. 2, pp. 835-839, 2014.
- [4] S. N. M. Ariffin, M. F. A. Kadir, A. N. M. Rose, M. A. Mohamed and A. R. Mamat, "User Authentication Using Grid Based Method," *International Journal of Engineering & Technology*, vol. 7, no. 2.14, pp. 97-100, 2018.
- [5] E. Orna and G. Stevens, *Managing information for research*, Maidenhead: Open University Press, 2009.
- [6] M. Eiband, M. Khamis, E. von Zezschwitz, H. Hussmann and F. Alt, "Understanding Shoulder Surfing in the Wild: Stories from Users and Observers," in *Proceedings of the 2017 SIGCHI Conference on Human Factors in Computing Systems (CHI '17)*, 2017.
- [7] T. j. Fong, A. Abdullah, N. Jhanjhi and M. Supramaniam, "The Coin Passcode: A Shoulder-Surfing Proof Graphical Password Authentication Model for Mobile Devices," (*IJACSA*) *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 1, pp. 301-308, 2019.
- [8] C. W. Dawson, *Projects in Computing and Information Systems: A Student's Guide*, Addison Wesley, 2009.