Let's Get Cracking: Leveraging Gameplay from an Adversarial Perspective to Teach Password Security Concepts

Dylan Lins Brasiliense Drucker

4th Year Project Report
Computer Science
School of Informatics
University of Edinburgh

2024

Abstract

Text-based passwords are the most commonly used method of authentication used on the internet. Despite their extensive use, a significant gap exists between the general public's understanding of password security and the realities posed by adversarial threats. This dissertation introduces "Let's Get Cracking", a serious game designed to enhance users' understanding of password security by placing them in the role of an adversary attempting to crack passwords. The game simulates real-world password-cracking techniques, including brute-force attacks, dictionary attacks, and the exploitation of common password patterns, thereby providing players with insight into what makes a password secure. Through a comprehensive literature review, game design documentation, and empirical evaluation involving user gameplay data and questionnaires, this study demonstrates the effectiveness of adversarial gameplay in raising awareness and understanding of password security. The findings suggest that players overwhelmingly found the game to be fun and engaging. It effectively helped them identify weaknesses in their own password practices and greatly improved their understanding of various aspects involved in password security.

Research Ethics Approval

This project obtained approval from the Informatics Research Ethics committee.
Ethics application number: 214977
Date when approval was obtained: 2024-01-30
The participants' information sheet and a consent form are included in the appendix.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Dylan Lins Brasiliense Drucker*)

Acknowledgements

I would like to thank all of the participants of the study for their help and enthusiasm. Thank you for volunteering your time.

I would also like to thank all of the game's initial play-testers, who helped me identify various typos, bugs, and problems in the game's structure.

Thank you to Ana for helping me proofread this dissertation with me at 3 in the morning, and for all your help.

Lastly, I would like to thank my supervisors, Borislav Ikomonov and Jingjie Li, for their amazing support and guidance over the course of this year. I greatly enjoyed working on this project with you, and it has played no small part in my decision to pursue a Master's degree. I look forward to working together next year.

Table of Contents

Chapter 1

Introduction

1.1 Password Security

Passwords play a fundamental role in digital security, acting as the primary line of defense in user authentication. Despite the emergence of alternative authentication systems, text-based passwords remain the most common user authentication system in computer and web systems (Kelley et al., 2012). Systems for cracking passwords have advanced, leading to the implementation of stricter password requirements. Despite the increasing use of password managers that generate complex passwords, a significant portion of users still opt to create their own (Pearman et al., 2019). Unfortunately, these user-generated passwords tend to be less secure, as they often prioritise memorability over strength.

Password data breaches are frequent and have potentially severe consequences, yet, multiple studies have identified a fundamental gap between the average person's understanding of password security and the practices that are exploited by the attacker model, leading to unsafe password practices (Ur et al., 2016) (Pearman et al., 2019). The current most widely used method of measuring password strength is password-strength meters. These usually employ LUDS (lowercase, uppercase, digit, symbols) which measures password strength as a product of length and diversity of character types. This system is considered highly cumbersome and ineffective as it can mislead users into creating passwords that, while considered strong by these criteria, are, in reality, highly susceptible to being cracked (e.g. P@$$w0rd123) (Wheeler, 2016). It also fails to capture heuristics such as the use of common keyboard patterns, character substitutions, dictionary words, and predictable positioning of characters. The problem introduced by ineffective password strength estimators is exacerbated by the gap in knowledge regarding password storage practices and the effectiveness of brute-force attacks.

1.2 Limitations of Existing Serious Games

The use of serious games - games that do not have entertainment as their primary purpose - has been proven to be an effective teaching method. By immersing players

in interactive and engaging environments, they facilitate the retention of information, enabling players to acquire and apply new knowledge and skills in practical situations (Backlund and Hendrix, 2013).

While serious games aimed at teaching important password security practices exist, they often adopt an instructional approach. These often explicitly state which password characteristics players should avoid, without proper reference to the foundational principles behind these practices, including the mechanisms of brute-force attacks, hashing, and dictionary attacks. However, by focusing on enhancing players' understanding of the attacker model, these games could empower players to come to these conclusions independently.

1.3 Proposal and Rationale for "Let's Get Cracking"

These insights led to the creation of the serious game detailed in this project, "Let's Get Cracking". Designed to place players in the role of an adversary, the game challenges them to crack as many passwords from a leaked database as possible within a set time limit. By adopting this unique adversarial perspective, players are encouraged to think creatively about exploiting vulnerabilities in password security to maximise their score. As a result, by analysing the cracked passwords, they are able to recognise the characteristics of weak passwords intuitively. This approach aims to teach the underlying mechanics behind password storage, expose players to the tools used by attackers, and ultimately help players intuitively understand the characteristics of weak passwords.

1.4 Aims and Organisation

This dissertation aims to understand how engaging in a serious game about simulating password cracking from an adversarial perspective improves players' understanding of secure password practices. The effectiveness of "Let's Get Cracking" was evaluated in a study of 22 participants. Analysis was completed on their recorded gameplay and their responses to a questionnaire before and after playing the game.

The dissertation is organised as follows: Chapter 2 reviews the existing literature on serious games and password security. Chapter 3 details the application of these concepts throughout the design and implementation of "Let's Get Cracking". Chapter 4 presents the methodology used in designing the questionnaire used to evaluate the game. The results from the questionnaire and the recorded gameplay are reviewed in Chapter 5. Finally, Chapter 6 concludes the dissertation by synthesising the results and proposing potential future extensions to improve the game based on results from the study.

Chapter 2

Background

This chapter serves as a background to understanding passwords in digital security and serious games. The chapter introduces the necessary terminology for understanding password storage, hashing, and the strategies employed by attackers. It also examines the gap in knowledge between public perception and the actual principles of secure password practices. The concept of serious games is introduced, and examples of existing serious games about password security are analysed and critiqued. These games usually bring attention to a very limited number of weak password characteristics and do not explain the underlying reasons for their vulnerabilities, particularly in relation to the attack model. These insights were used in developing ”Let's Get Cracking” with the aim of reducing the disparity in public perception in ways that previous work has not fully achieved.

2.1 Passwords in Digital Security

2.1.1 Password Storage

As passwords remain the most commonly used method of user authentication, organisations must carefully consider how they store them within their databases. Storing passwords in plaintext, as exemplified by notorious leaks such as the RockYou data breach discussed in section 2.1.3, is a critical vulnerability that should be avoided. Instead, the industry-standard employs a combination of hashing and salting.

2.1.1.1 Hashing and Salting

Hashing involves using a hash function to convert plaintext passwords into a unique fixed-length string of characters. Hashing is a one-way function designed to be easy to compute but difficult to reverse. Hashing functions are deterministic in that the resulting hash value will always be the same for a given input. This predictability is necessary for authentication processes, as it allows systems to verify passwords by comparing the hash of the inputted password with the stored hash in the database.

Hashing can be combined with salting, which involves adding a unique random value,

known as a salt, to each password before hashing. This ensures that two identical passwords will have different hashes because each has been hashed with a unique salt.

2.1.2 Types of Attacks

When a password storage database is compromised, attackers typically gain access to a list of hashed passwords. They then employ computational resources to generate a roster of potential passwords and subsequently hash them using the same algorithm implemented by the password storage system. The attacker then matches the newly generated hashes with those stored in the compromised database. The detection of a match indicates a successful guess of the original password.

While attackers could potentially resort to brute-force attacks, systematically generating hashes for every conceivable password, this method is inefficient. Instead, they often exploit patterns used to create memorable passwords. These patterns include the incorporation of easily recognisable keyboard sequences (e.g., ”qwerty” and ”1234”), the substitution of characters with similar symbols (e.g., using ”@” instead of ”a”), and exploiting predictable character placements (e.g. numbers and special characters tend to appear at the end of passwords). Attackers use what are called dictionary attacks, which involve making guesses using common words, phrases, keyboard patterns, and cracked passwords (Summers and Bosworth, 2004). Each of these techniques is aimed at exploiting the predictability of human password creation.

In scenarios where salts are not implemented, attackers can potentially exploit precomputed tables of hashes known as rainbow tables. These tables contain pairs of plaintext passwords and their corresponding hash values. These allow password crackers to find matches without the extensive computational effort of generating these hashes themselves.

Once a password is obtained, attackers may unlock the compromised account and potentially other accounts across the internet that employ the same username and password combination. This is primarily due to the common practice of password reuse among individuals. A study by Poornachandran et al. (2016) found that 59% of regular internet users reuse passwords for multiple accounts and that 57% use slightly modified versions of existing passwords.

2.1.3 The RockYou Data Breach

RockYou, an advertising network best known for distributing mobile games to third-party platforms, such as Facebook, suffered a significant security breach in December 2009 due to a SQL vulnerability. 32 million user accounts were exposed, including many from third-party websites. It was revealed that RockYou was storing passwords in plaintext rather than securely hashing them. To this day, it remains the largest single data breach of plaintext passwords[1].

[1]Understanding RockYou.txt, https://www.keepersecurity.com/blog/2023/08/04/understanding-rockyou-txt-a-tool-for-security-and-a-weapon-for-hackers/

Due to its size and ease of access, the leaked database has been used extensively in password security research, providing data regarding password selection practices and patterns (Ur et al., 2016) (Kelley et al., 2012).

2.1.4 Public Perception of Password Security

Public perception of password security often deviates from established best practices. A study by Ur et al. (2016) revealed that many individuals underestimate the vulnerabilities associated with constructing passwords around common keyboard patterns and familiar phrases. Users frequently employ predictable characteristics in their passwords, such as incorporating words, easily recognizable keyboard sequences, and substituting characters with similar-looking symbols (also known as ”common l33t”).

It has also been observed that users tend to base passwords on easily guessable words, phrases, or concepts, including names, dates, and song lyrics. This stems from a lack of knowledge regarding the details of large-scale database attacks. Although 73% of participants in the Ur et al. (2016) study identified large-scale guessing attacks as a threat, there was a wide variance in the estimated number of adversarial guesses. 34% of participants believed a password to be secure if it could resist up to 50 guesses, while 67% believed that withstanding 50,000 guesses was a sign of security. In reality, a good password should be able to withstand between 10^{14} - 10^{20} guesses to be considered secure, depending on the type of hash function used (Ur et al., 2016).

2.2 Serious Games

Serious games are commonly defined as games that do not have entertainment, enjoyment, or fun as their primary purpose. The field of serious games has seen rapid growth in the past decade, with a market value in the billions (Laamarti et al., 2014). The field has also seen rapid growth in research due to its potential in various industries. Serious games leverage the interactive, immersive, and addictive nature of games to engage and motivate users to learn and improve skills (Laamarti et al., 2014). The research conducted by Backlund and Hendrix (2013) yielded positive outcomes for serious games and their impact on learning. Out of the 40 selected studies, 29 demonstrated a positive effect, while only 7 showed neutral results.

2.2.1 Key Elements of Effective Serious Games

2.2.1.1 Enjoyment

Although serious games do not prioritise entertainment, player engagement is highly correlated with enjoyment and fun. To do this, the game should contain varied and interesting gameplay that embeds the educational goal. A balance between the educational and game parts must be carefully considered by ensuring all learning tasks are connected to in-game elements (Caserman et al., 2020).

2.2.1.2 Appropriate Rewards, Feedback, and a Clear Objective

The game's goal should be aligned with its educational purpose and encouraged through positive reinforcement. For example, a very common form of in-game reward is through a point or score system. This can be further improved with a high-score table, allowing players to compare their performance against other players, further encouraging them to perform better (Caserman et al., 2020). The player's score and/or status must always be visible, and they should receive immediate feedback as a result of their actions. This is necessary not to break player concentration and to maintain a flow state (Sweetser and Wyeth, 2005).

2.2.1.3 Player Control

Players should possess a sense of autonomy over their decisions in-game rather than simply employing strategies intended by the developer. Through this, players are able to experiment and feel an increased sense of control. A game with too many constraints on player freedom can give the perception the player must follow a predetermined path, frustrating players and negatively impacting their experience. Player control must also be maintained in the case of player error. The game should continue to function if the player makes an error and help them recognise and fix the error through, for example, a warning message (Sweetser and Wyeth, 2005).

2.2.1.4 Adaptive Level of Difficulty

The level of challenge should match the skill level of the player. As the player naturally improves at the game, the difficulty should also increase in tandem. As a large discrepancy between challenge and skill level can negatively affect players' enjoyment of the game, care should be put into ensuring a minimal gap. The player should also be taught to play the game through an interactive tutorial. The tutorial should not feel very different from what actually playing the game is like (Sweetser and Wyeth, 2005).

2.2.1.5 Appropriate Graphics and Sound

Player engagement can be improved by stimulating different human senses through the use of appropriate audiovisual elements (Caserman et al., 2020). These include appropriately thematic visuals, sound effects, and background music. Care should be taken to ensure these game elements do not distract players from the goal and interfere with learning.

2.2.2 Examples of Serious Games and Related Work

2.2.2.1 Re-mission

The Re-Mission series of games are serious games designed for young cancer patients. In the game, you play as a nanobot designed to fight cancer and related infections in the human body (Figure 2.1). The goal of the game is to teach players about cancer treatments and how they work to promote adherence to self-care during treatment. A study conducted by Beale et al. (2007) found a significantly larger increase in the

retained knowledge of re-mission players over a 3-month period compared to the control group.

Figure 2.1: Gameplay of re-mission (Beale et al., 2007)

2.2.3 GamePass

GamePass is a mechanism to gamify the creation of graphical user authentication (GUA) passwords. GUA passwords are an alternative to text-based passwords, where users draw passwords on background images. GamePass encourages users to create more unpredictable passwords by guiding user attention to non-salient areas of authentication images. This behaviour is further encouraged with a reward system which gives higher scores based on how secure the password is (Figure 2.2). The research showed an improvement in the GUA passwords created by GamePass players. Although GUA passwords are much rarer than traditional text-based passwords, GamePass demonstrates how an in-game reward system can encourage users to generate better passwords (Raptis et al., 2021).

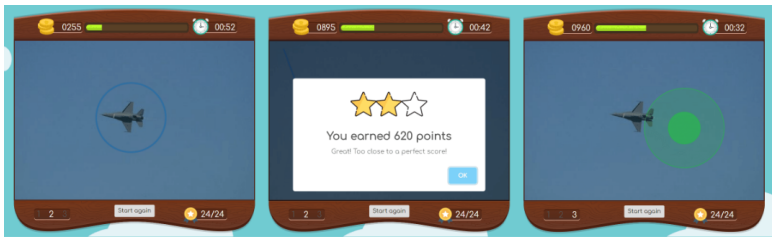


Figure 2.2: Gameplay of GamePass (Raptis et al., 2021)

2.2.4 PASDJO

PASDJO is an online game where players rate the strength of text-based passwords under a time limit (Figure 2.3). Players are then scored based on how closely their perceived score is to the expected score. A results screen provides very brief justifications for the expected strength ratings of passwords. This helps users learn about the characteristics of weak passwords they may not have been familiar with. The study observed how users underestimated passphrases by an average of 1.4 points on a 5-point strength. Although multiple playthroughs of the game were shown to improve password strength perception, only 27% of the participants chose to play more than one round. The game

only used 4 types of passwords to test a few selected characteristics: random passwords, common passphrases, common alterations, and ”common l33t” (the substitution of letters for similar-looking characters). PASDJO falls short in explaining the reasons why these characteristics are considered weak, omitting any explanation of the attack model and the tools used to exploit these vulnerabilities (Seitz and Hussmann, 2017).

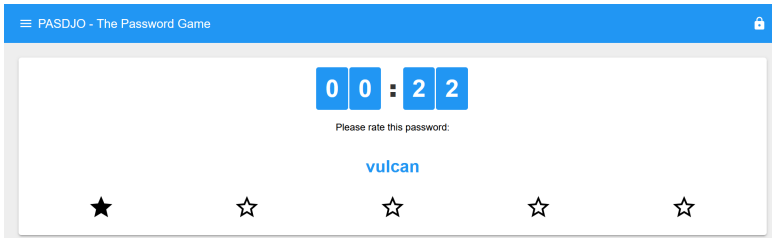


Figure 2.3: Gameplay of PASDJO (Seitz and Hussmann, 2017)

2.2.5 GAP

GAP is a serious game created with the goal of educating users about common and unsafe password habits. Players play as a tank that must navigate a maze filled with barriers representing different passwords (Figure 2.4). The player must shoot these barriers down by identifying the weak characteristics of the passwords they represent (e.g., the password contains an uppercase letter at the beginning, which over 70% of users do; hence, it is not safe practice). The study found that after an average of 3 and a half minutes of playing, participants identification of the tested password characteristics improved from an average of 60.65% to 82.96%. However, the study only tested six characteristics, focusing on the identification of password strength through the placement of uppercase letters, digits, and special characters at either the beginning or end of passwords. This omits password characteristics such as their susceptibility to dictionary attacks. The game also does little to explain why these characteristics are insecure beyond the identification they are found in a large percentage of users' passwords (Tupsamudre et al., 2018).

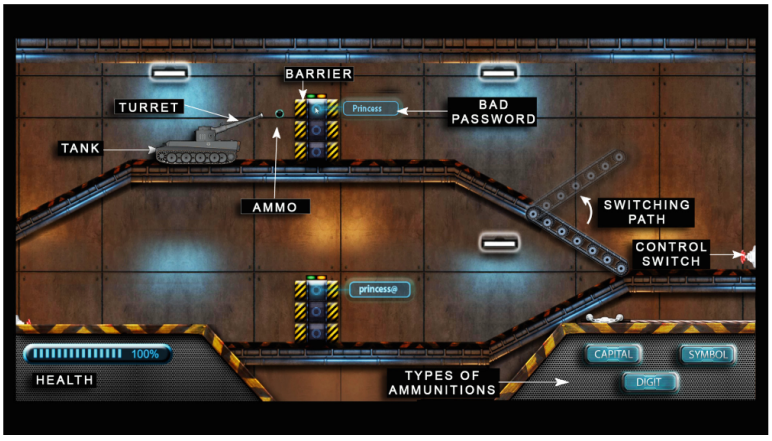


Figure 2.4: Gameplay of GAP (Tupsamudre et al., 2018)

Chapter 3

Design and Implementation

This chapter outlines the conceptualisation, development, and implementation of ”Let’s Get Cracking”. The process of creating the game was informed using the insights from Chapter 2. In particular, focus was placed on the key elements of serious games and addressing the gaps identified in related work.

3.1 Initial Concept

Passwords are a ubiquitous aspect of modern digital life, and the average person is frequently reminded to create secure passwords, often guided by heuristics such as password strength meters. However, these meters usually calculate strength based on password length and the variety of character types used (LUDS). They often fail to consider risks such as the predictability of placing numbers, special characters, and uppercase letters in common positions, as well as the usage of words and predictable patterns. These render passwords susceptible to brute-force attacks and dictionary attacks, respectively.

While the serious games discussed in section 2.2.2 identify many of these risks, they do not cover the full range of these characteristics and do not explain why they are insecure with reference to the attack model. ”Let’s Get Cracking” addresses this by positioning players in the role of the adversary. From this unique perspective, players better understand the attack model and the tools used for cracking passwords. Players are encouraged to think creatively about how to maximise the number of accounts they can crack. This process naturally leads to an implicit learning experience, where players discern the characteristics of weak passwords and gain insights on how to strengthen their own passwords.

3.2 Gamification

The game simulates the adversarial position by giving players simplified versions of tools used by real password crackers. This includes a straightforward interface that allows players to input a character and/or word sequence and then simulate the brute-

force generation and matching of hashes with those in a leaked password database. This input sequence can be constructed using numbers, letters, special characters, and a large variety of words for use in dictionary attacks.

The gamification of the system was created by using the principles discussed in section 2.2.1. As recommended in section 2.2.1.2, the game has a clear goal: crack as many accounts as possible within the allotted time limit. Time counts down in real time and is also consumed upon generating hashes. The time consumed during hash generation is 10^{10} hashes per second, based on an estimate of the speed of a fast hashing algorithm used in the zxcvbn library discussed in section 4.3 (Wheeler, 2016). In reality, the true quantity can vary greatly based on the hashing function, but the amount in the game was kept constant so as not to confuse players. The player repeats this gameplay loop of inputting, generating hashes, and cracking passwords until no more time remains. This encompasses a singular level of the game.

This system encourages player freedom and experimentation, as recommended in section 2.2.1.3. It has a large number of gameplay systems, providing constantly varied gameplay to engage the player. In experimenting with these different systems, players receive immediate feedback on the number of accounts they have cracked with the hashes they have generated. The learning outcome of the game is thus embedded in the game's goal, which is an important part of balancing the educational and entertainment parts of the game, as outlined in section 2.2.1.1. By observing how different input sequences affect the player's score and time, the player can learn the patterns which crack the most accounts in the least amount of time and, therefore, the characteristics of these weak passwords.

3.3 Technologies Used

3.3.1 Godot

The game engine used for this project was Godot 4.0, a free and open-source 2D and 3D game engine. It was chosen over other game engines as it is a lightweight engine with the ability to export to Windows, Mac, Linux, Android, iOS, and the web (HTML5). Although Godot is not as feature-rich as other engines such as Unity and Unreal, it contains everything necessary for the development of "Let's Get Cracking". The lack of unnecessary features allowed for a much more efficient workflow, as the time between implementing and testing a gameplay feature is much faster than other engines. The open-source license also presents fewer restrictions in future distributions of the game.

3.3.2 GDScript

Although Godot also supports C++ and C, GDScript was chosen as the primary programming language for the game. GDScript is a high-level object-oriented programming language designed specifically for use in Godot. Its syntax is very similar to that of Python. As it was designed specifically for Godot, it is tightly integrated and optimised for the engine.

3.3.3 Python

Python was used to process and format word and password data within Godot. Relevant words and passwords were processed to remove invalid characters and to set all uppercase characters to lowercase. These processed words would then be inserted into hash sets, and converted to JSON for use in Godot.

3.4 User Interface

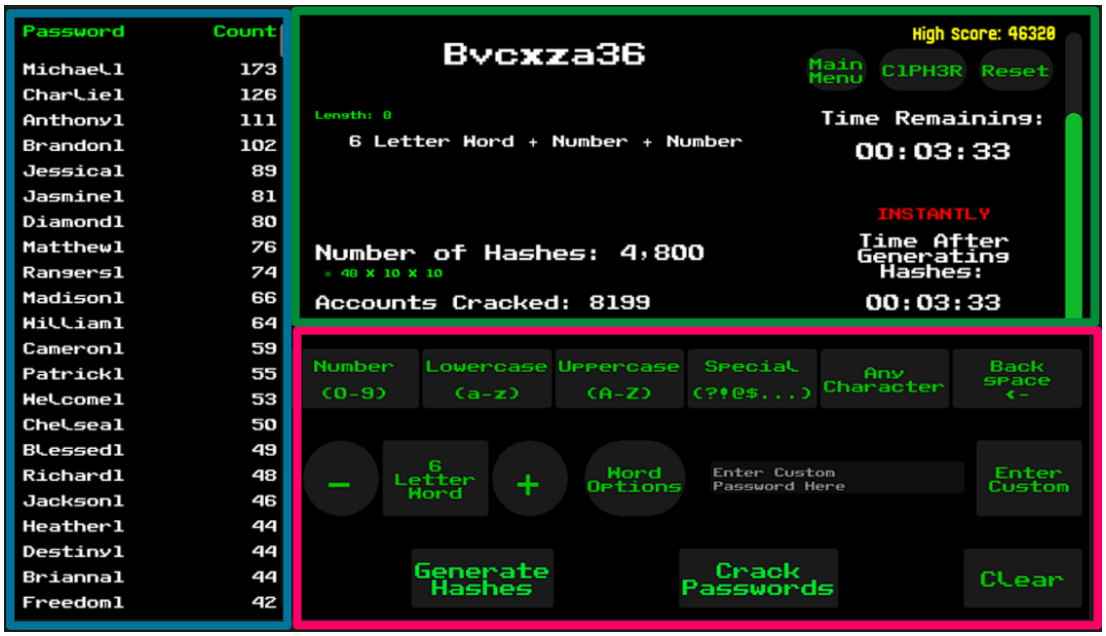


Figure 3.1: Screenshot of Gameplay: Level 4. Coloured boxes added for clarity.

Figure 3.1 shows a screenshot of level 4 of the game, where all UI elements and password-cracking tools have been unlocked. The interface has been intuitively designed with a clear separation between the buttons the player can interact with (the pink box) and the informational data (the green box).

The bottom buttons (pink box) are organised into rows. The top row contains the buttons for inputting specific character types, including numbers, lowercase letters, uppercase letters, special characters, and any character (a combination of all four character types). The second row contains buttons for inputting dictionary attacks, including the word-input button (labelled ”6 Letter Word” in the above figure), plus and minus buttons for adjusting word length, and the ”Word Options” button. The bottom row contains the buttons for generating hashes and cracking passwords. These were kept as separate buttons despite the fact they are always used sequentially. This was a design decision to reinforce the understanding of the process of password cracking for players: first generating the hashes and then cracking passwords by matching them. Lastly, the right-most column includes a backspace button and a clear button for removing values from the input.

The top half of the interface (the green box) contains information about the player’s

current input. When the player has inputted something, examples of passwords generated from the input sequence appear at the very top of the screen. A new example is generated and displayed every 0.3 seconds. The exact input sequence, along with the length of the input, is displayed below.

Below this, the number of hashes needed to generate all combinations of the input sequence is displayed. This is accompanied by the combinational equation used to calculate it, which is displayed just underneath it. This allows players to compare how different character types influence the number of hashes.

The top-right of the interface is dedicated to displaying the time remaining. The timer counts down in real-time and is represented with a progress bar to show how much time remaining there is in relation to how much time the player started with. When the player inputs something, the UI updates to display the time required to generate all hashes and the time remaining afterwards. The progress bar has also been updated to represent the remaining time after generating hashes. If the time to generate the hashes is less than one second, the time to generate is replaced with ”Instantly”, as can be seen above.

The left column of the interface (the blue box) is primarily dedicated to displaying cracked passwords and the number of accounts found using the same password. It is also used during hash generation to display a short ”digital rain” animation.

Lastly, above the timers are additional quality-of-life buttons for returning to the main menu, replaying the level's opening dialogue (labelled ”CiPH3R”), and resetting the level. Above these, highlighted in yellow, is the high score of the current level. This gives players an attainable goal to reach and provides a reference to the number of cracked accounts they should aim to reach before their time runs out.

3.5 Level Progression

To avoid overwhelming the player with the quantity of password-cracking tools at their disposal, the game introduces these tools gradually over multiple levels. These tools make password cracking more efficient, and thus, an increasing difficulty level balances this. This also ensures players effectively use and experiment with these newly introduced tools.

The game features an adaptive level of difficulty as discussed in section 2.2.1.4. The difficulty of the game is raised by increasing the complexity of the password requirements for the passwords that can be found within a level. For example, level 5 is the most difficult level as the passwords in the database must contain at least 8 characters, one number, one lowercase letter, one uppercase letter, and a special character. This increased complexity renders traditional brute-force attacks infeasible, therefore dictionary attacks must be employed. This ensures gameplay remains varied as players will need to adapt their strategy for different password requirements, as discussed in section 2.2.1.1.

These password constraints also match the password requirements that modern websites increasingly ask users to conform to, raising the likelihood that players will find a weakness in their current password practices.

3.5.1 Tutorial

The game includes a tutorial to familiarise players with cyber-security concepts such as hashes, their properties, how they are used in password storage, and the process involved in cracking them. Following this foundational knowledge, the tutorial transitions to familiarising players with the game's user interface and the main gameplay loop. By ensuring players properly understand the game's mechanics and the cyber-security concepts being emulated, their mental model of the attacker is reinforced, providing context and deeper meaning to their actions within the game.

The in-game tutorial was implemented in two distinct sections. The first section provides a simple overview of password database storage, hashes, their properties, and how passwords can be cracked. The second section familiarises players with the UI and the gameplay loop. All in-game explanations are delivered through dialogue from the game's singular NPC (non-playable character) CiPH3R. The choice to use an NPC (non-playable character) to explain these concepts through dialogue was chosen to better incorporate the educational elements within the thematic context of the game, as discussed in section 2.2.1.1.

In the first stage of the tutorial, CiPH3R introduces themself as a member of the same cybercriminal team as the player. Upon learning that the player does not know how to complete a brute-force attack on the hashed password database, CiPH3R provides a basic overview of password security concepts. CiPH3R's explanations are aided using visualisations and animations. In the first part of the explanation, CiPH3R explains how websites must store passwords in a database for authentication and the risks associated with storing passwords as plaintext (Figure 3.2). Then, CiPH3R explains what a hash is and its properties. Figure 3.3 shows the animation used during this explanation, displaying different examples of passwords and their respective unique hashes. With this context, CiPH3R then provides a basic overview of how hashed passwords are stored in a database (Figure 3.4). CiPH3R then explains how the only way to crack a hash would be to brute-force generate hashes and find one that matches, and how this can be done by exploiting common patterns in passwords. This acts as a natural segue into the first main level of the game.

Once in the first level, CiPH3R briefly explains the different UI elements the player can see. CiPH3R then instructs the player through two examples of password cracking. CiPH3R first requests that the player inputs the sequence necessary to crack all 5-digit passwords. Then, CiPH3R explains the process of generating hashes, and then matching those hashes with "Crack Passwords". This results in the player cracking thousands of accounts. To show the player how different character types can be combined, this is repeated once more with the input sequence for all passwords with the pattern of 5 lowercase letters followed by a number on the end (Figure 3.5). Afterwards, CiPH3R lets the player complete the rest of the level on their own with the goal of cracking as many passwords as possible within the time remaining.

CiPH3R's dialogue for the tutorial and all levels can be read in full in Appendix D.2.

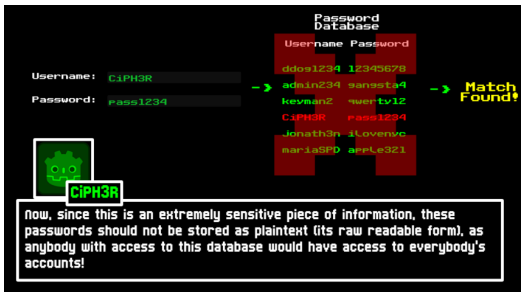


Figure 3.2: Tutorial: Passwords Stored in a Plaintext Database

Figure 3.3: Tutorial: Hash Definition

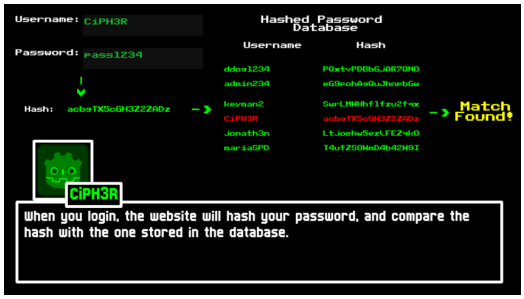


Figure 3.4: Tutorial: Storing Hashed Passwords

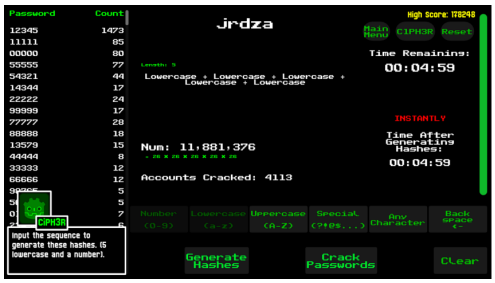


Figure 3.5: Tutorial: Level 1

3.5.2 Main Levels

The game consists of a tutorial and six levels of which one is a bonus free-play level. As outlined in section 3.5, most levels introduce the player to a new password-cracking tool. Table D.1 in Appendix D shows what tool is unlocked in each level, the minimum password requirements of that level, and the starting time limit.

The free-play level is the final level of the game and serves as both a reward and an experimental playground for the player. It contains all unlocked UI elements, no password restrictions, and infinite time. This promotes experimentation with input combinations that the player may have otherwise not been able to use under the previous time limits and password requirements.

3.5.2.1 Gameplay Loop

As most levels introduce a new password-cracking tool, each level starts with a brief overview. CiPH3R's dialogue introduces the new tool, how to access it in the UI, and the rationale behind it. For example, when introducing dictionary attacks, CiPH3R says "So you may have noticed that a lot of passwords containing letters aren't just random, they are usually words...so to make your cracking more efficient, I've given you the ability to input dictionary words". CiPH3R additionally describes the level's minimum password requirements and a hint on how to best use the new tool. For example, on level 4, passwords must contain an uppercase letter, therefore, CiPH3R says, "I bet you'll find capitalising the first letter of words quite useful.", suggesting the player should use the "Capitalise First" option within the "Word Options" menu. CiPH3R also gives hints

about unsafe password characteristics the player should exploit. For example, CiPH3R says, ”Think about where people like to put numbers in their passwords”, guiding players to think critically about the placement of digits in passwords, which frequently appear at the end. This helps players intuitively come to their own conclusions about unsafe password habits.

After reading the dialogue, players are able to input their character/word sequence. If the sequence matches the password requirements, the player is able to generate hashes by pressing the ”Generate Hashes” button. This plays a short animation on the left-hand column of the interface showing randomly generated ”hashes” streaming downwards. Whilst this animation is playing, the time remaining and the progress bar gradually decrease to their new updated values. This illusion of time being sped up and consumed is reinforced by the increase in tempo and pitch of the in-game music. Once the animation has finished, the left column will display ”Finished Generating: Ready to Crack Passwords”, and the ”Crack Passwords” button will begin flashing. Once pressed, matched passwords will be displayed as they are found in the left column. For each found password, the ”Accounts Cracked” quantity is updated, accompanied by a satisfying sound effect. Once the matching has been completed, the loop has finished, and players are able to input a new sequence. If there is no more time remaining, a large indicator displaying ”Out of Time!” will appear, and the ”Clear” button will be replaced by a flashing ”See Results” button. Pressing this fades the level out and displays the results screen. From there, the player can continue to the next level.

This simple gameplay loop encourages player experimentation. In each loop, players can input a new sequence and observe how successful it is at cracking passwords. Input sequences that exploit weak password characteristics naturally reward the player with more cracked accounts and, therefore, a higher score. Through this adversarial perspective, players are encouraged to learn and exploit weak password characteristics.

3.5.3 Results Screen

After each level, a results screen displays various statistics regarding the passwords cracked by the user in each level. This allows players to reflect on the input sequences that were most effective.

At the end of each level, players are shown a screen detailing various statistics about the passwords they have cracked. The leftmost column of the interface shows the passwords cracked sorted by highest count. The stats shown in the centre of the interface show the total number of cracked accounts, the percentage of all cracked accounts, the input sequence that resulted in the highest number of cracked accounts, and the input sequence with the highest ratio of cracked accounts to the number of hashes generated. The purpose of these statistics is to allow players to reflect and analyse the relative effectiveness of the different attack strategies and input sequences they used.

3.5.3.1 Leaderboard

The rightmost column contains the leaderboard. This allows the player to compare their scores with others. This provides a frame of reference as to how well they have

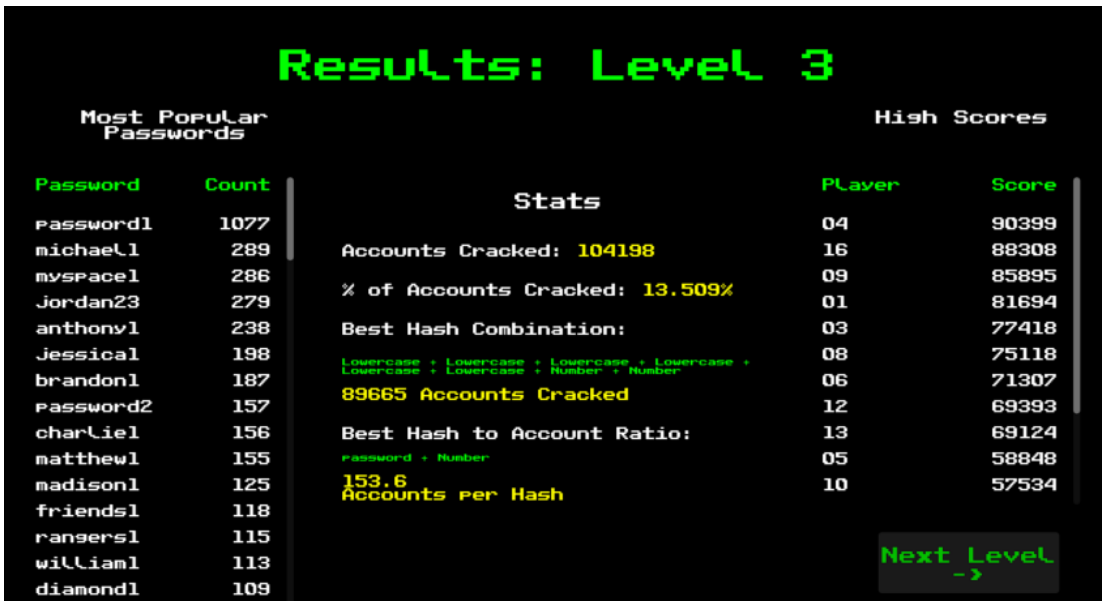


Figure 3.6: The results screen

performed, adding a competitive motivation and encouraging them to do better, as outlined in section 2.2.1.2.

3.6 Password Cracking Tools

3.6.1 Custom String Input

Introduced in the second level, the custom string input box allows players to input their own strings into the input sequence. CiPH3R provides examples of how to best use this tool, including inputting common passphrases such as ”password” and using ”1” rather than using the ”Number” button, as the player may have observed that it is a commonly used number in passwords. As a precaution, CiPH3R reminds players not to input their own passwords into the custom string box.

Custom strings serve as a natural precursor to dictionary attacks. It is expected that players will find custom string inputs rather inefficient at cracking large quantities of passwords and wish for a more streamlined method of inputting multiple words at once. This nicely leads the player to understand the rationale for dictionary attacks, which are introduced in the next level.

3.6.2 Dictionary Attacks

Dictionary attacks are introduced in the third level. CiPH3R instructs the player how to input a word and how to adjust the word length using the plus and minus buttons. While CiPH3R briefly explains the rationale behind using dictionary attacks rather than traditional brute-force, players can intuitively come to this conclusion themselves by observing how much more efficient dictionary attacks are in terms of hash computation time.

Figure 3.7: Level 2: Custom String Input

3.6.3 Word Options

The fourth level of the game introduces the ability to further customise dictionary attacks using the ”Word Options” button. Pressing it brings down a menu, shown in Figure 3.9. The menu allows the customisation of the ”Word Types” used in the brute-force dictionary attacks, and the ability to modify the words using ”Common Substitutions”.

The different ”Word Types” include Dictionary Words (default), Common Names (e.g. ’michael’, ’francesca’), Chemical Elements (e.g. ’boron’, ’hydrogen’), Well-Known Places (e.g. ’london’, ’spain’), Keyboard Patterns (e.g. ’qwerty’, ’12345678’), and Previously Found Passwords, which includes the passwords cracked by the player thus far. Purposefully, not all of these word categories are particularly useful in dictionary attacks. This aims to deter players from mindlessly toggling all categories and encourages critical thinking about which ones would be most beneficial to include.

”Common Substitutions” provides options which allow players to modify the character types within the words. ”Capitalise First” capitalises the first letter of a word, ”Capitalise All” capitalises all letters of a word, and ”Common L33T” replaces certain letters with similar-looking special characters and/or numbers.

CiPH3R explains these details in the opening dialogue. He additionally hints at the utility of ”Capitalise First,” given the level 4 and 5 password requirements, which require both uppercase and lowercase characters. This teaches players to exploit the fact that uppercase characters frequently appear at the start of passwords, allowing them to learn intuitively that it is an unsafe practice.

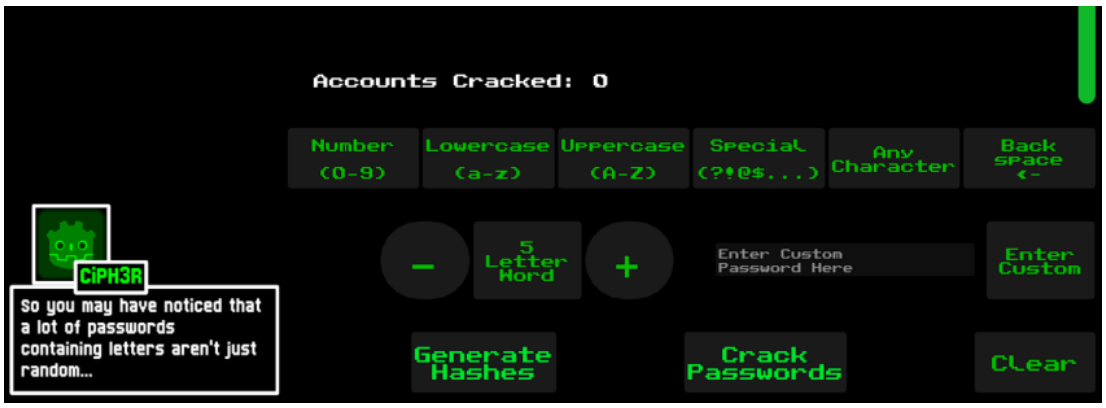


Figure 3.8: Level 3: Dictionary Attacks Unlocked

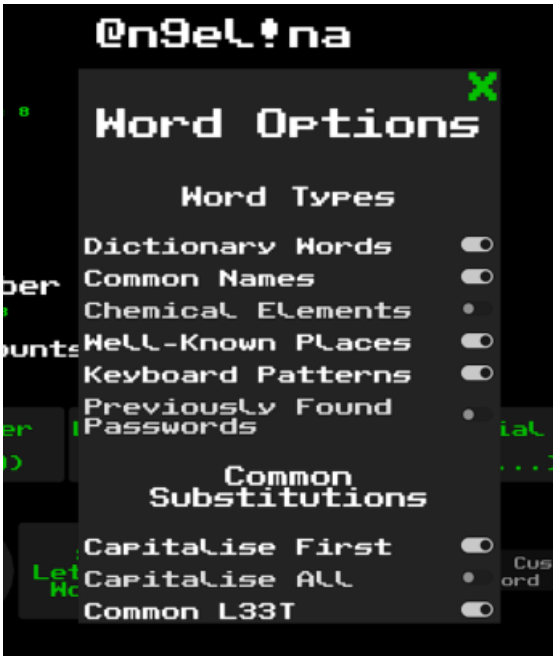


Figure 3.9: Level 4: Word Options Menu

3.7 Warning Messages

An important feature of successful serious games, as described in section 2.2.1.3, is ensuring the game continues to function despite an error in the player's input. The game features warning messages to help players diagnose and fix their errors. When the player makes an error, the appropriate warning message appears for a few seconds in the top left corner of the UI before fading out.

Figure 3.10 shows examples of these warning messages. From left to right, these warnings appear if the input sequence does not meet the level's password requirements, "Generate Hashes" has been pressed when nothing has been inputted, "Crack Passwords" has been pressed before hashes have been generated, and if the time required to generate the hashes for the inputted sequence is longer than the time remaining, resulting in the partial generation of the hashes.

Figure 3.10: Examples of in-game warning messages.

3.8 Graphics and Sound

As mentioned in section 2.2.1.5, appropriate graphics and sound are crucial in creating an immersive and engaging game, thus improving the player's capacity to learn. Care was therefore taken to include thematically appropriate graphics, music, and sound effects.

3.8.1 Graphics

The game's aesthetics are very simple, yet thematically consistent. The game's colour palette consists of mostly green and black, loosely emulating a green and black terminal window emblematic of a 90s media interpretation of "hacking". For example, the animation that plays during hash generation, shown in Figure C.1, is loosely inspired by the "digital rain" effect used in *The Matrix*[1]. This aesthetic choice not only engages the player but also allows the game's graphics to be simple and mostly text-based, ensuring a lightweight game that can be played on any hardware.

3.8.2 Sound effects and Music

The game's sound effects and background music were created to increase immersion and provide feedback for the player's interactions and inputs. The soundtrack consists of two tracks composed by myself using an electric keyboard. The first of these tracks plays during the tutorial and results screen. It is a moody minimalist piece designed not to distract players as they focus on the game's content. This tone was also chosen to contrast the higher energy of the second track, which plays during each level. Instead, the second track features a much faster tempo and two distinct musical sections. It is a mostly chord-based piece, similar to the tutorial track, to not distract players from the gameplay. The track is approximately three minutes long and contains improvised variations of the two musical sections so as not to get too repetitive when listened to

[1]Matrix Digital Rain, https://en.wikipedia.org/wiki/Matrix_digital_rain

throughout the full game. Both tracks were designed to loop endlessly to be seamlessly integrated into gameplay.

The in-game sound effects were obtained online from Pixabay[2], a royalty-free stock media website. These were used to give the player feedback for their inputs and changes in their score. A sound effect was added to every button in the interface to give players a sense of tactile feedback. Different sounds were utilised for buttons used for modifying the input sequence, and other buttons such as ”Generate Hashes” and ”Crack Passwords, to subtly distinguish their functionality. Sound effects were also used to alert players of a change in their scores. When cracking passwords, each time a new match is found, a satisfying ”ding” sound effect is played. If no passwords are found, a ”fail” jingle plays once matching has ended.

These sound effects support the need for immediate feedback outlined in section 2.2.1.2.

3.9 Changes Made During Play-Testing

Before starting the official evaluation of the game, four colleagues were asked to informally play-test it. Below are some of the changes made to the game which address the issues observed from their gameplay.

3.9.1 Leaderboard

Play-testers voiced their concern that the game’s levels lacked a concrete goal or any frame of reference for what a good final score should look like. This problem was exacerbated by the fact that as the levels progress and password complexity increases, the expected final score of each level decreases. This could potentially discourage players who may assume the decrease in their scores throughout the game was attributed to their worsening skill.

This was addressed with the leaderboard system shown in the results screen. With its addition, players are motivated to achieve higher scores. The highest score on the leaderboard is displayed at the top right of the interface during gameplay to give players a reference of the scores they should aim for.

3.9.2 Shortened In-Game Time Limits

At this stage of the game development process, each level contained a significantly longer time limit than in the final version of the game. The expectation was that players would input hash sequences that would use up large quantities of in-game time. However, observations of play-tester gameplay showed that they would often tend to input simple sequences which would use up very little time. This problem was exacerbated by the introduction of more efficient password-cracking tools such as dictionary attacks. This resulted in play-testers growing frustrated and feeling levels were too lengthy.

[2]Pixabay, https://pixabay.com/

This was addressed by lowering the time limit of most levels to 5 minutes, providing a balance between allowing players to experiment with more complex input sequences while limiting their maximum time on a single level. The only level with a different time limit is level 2, with a limit of one hour instead. This was justified as dictionary attacks are not yet unlocked at this stage, and there is a minimum password requirement of passwords with a length greater than 8. Even simple input sequences, such as 8 lowercase letters, could use up over 30 minutes of in-game time. It also encourages player experimentation at an early point in the game and provides the opportunity to observe how an increased time limit affects gameplay on a larger evaluation group.

3.10 Emulating Password Cracking

As real hash generation would be computationally infeasible within the game's setting, it is instead emulated by pattern matching the player's input sequence with plaintext passwords. In-game, while the game appears to generate hashes, it is actually pattern matching each plaintext password with the input sequence. When a match is found, the player's score is increased by the password's frequency count.

While regex was originally used for password pattern matching, it did not perform optimally, especially when combined with dictionary attack inputs. Instead, a custom pattern-matching algorithm was implemented.

The algorithm's first step is to check if the plaintext password's length matches the expected length derived from the input sequence. If it does not, the algorithm can immediately return a false output. Otherwise, the password is checked in the next stage of the algorithm.

The next step is to check if the input sequence pattern matches the characters in the plaintext password. A hash set is initialised for each character type, including a set containing the valid dictionary words for dictionary attacks. This set dynamically changes depending on which word types the player has enabled. Each character in the password is matched with the corresponding character type in the input sequence. If the character or word is not found to be contained in the corresponding set, the algorithm returns false. Otherwise, if every input in the input sequence matches with the plaintext password, it is considered a match and the algorithm returns true.

By utilising hash sets, the algorithmic runtime for pattern matching each password is at most $O(m * n)$, where n is the length of the input sequence and m is the number of passwords. In practice, the runtime constant c is low due to the algorithm's ability to return false as soon as a mismatch within a single password is found. These optimisations are necessary as each use of "Generate Hashes" requires checking each one of the hundreds of thousands of passwords within each level.

3.11 Data

3.11.1 Passwords

The plaintext passwords used in the game were obtained from the RockYou data breach, which was detailed in section 2.1.3. While there may be ethical considerations with the usage of this data, the passwords in the game are not associated with any usernames, login details, or any other identifiable personal information. As the database is easily accessible and more than a decade old, the risks associated with using these passwords for the original users are negligible. Within the game, only a subset of the RockYou database is used to reduce the length of time for pattern matching. Each level contains at most 800,000 unique passwords, as opposed to the 13 million within the original dataset.

3.11.2 Dictionary Attacks

Players can customise their dictionary attacks by including different ”Word Types”. The final word types used in-game are ”Dictionary Words”, ”Common Names”, ”Common Places”, ”Chemical Elements”, ”Keyboard Patterns”, and ”Previously Found Passwords”. The latter is generated based on the passwords cracked by the player in-game.

”Dictionary Words” and ”Common Names” were both obtained from Python’s nltk library[3]. For the ”Dictionary Words” set, the 10,000 most common words from nltk.corpus were used. The names were similarly obtained from the 1000 most frequent names within the ”names” corpus.

”Common Places” is a set containing continents, countries, regions, and cities. Lists of these were obtained from Python’s ”geotext” library[4].

”Chemical Elements” includes the elements on the periodic table, as well as various chemical compounds, as otherwise the list would have been too small. These were obtained by extracting unique tokens from Wikipedia’s glossary of chemical formulae[5].

”Keyboard Patterns” were obtained by prompting ChatGPT 3.5, as it was not possible to find a comprehensive list of keyboard patterns. For each length n from 3 to 12 (the valid word lengths in-game), ChatGPT was prompted to ”Generate a list 100 keyboard patterns of length n that could be used in passwords”.

Each of these word lists was processed in Python into a dictionary data structure. The dictionary was constructed using the length of the word as the key and a set of all words of that length as the value. This would enable efficient dictionary-attack pattern matching and merging of dictionaries in-game. Each word-type dictionary would be processed into a JSON file, where it could be interpreted in the Godot engine.

[3]NLTK, https://www.nltk.org/

[4]Geotext, https://pypi.org/project/geotext/

[5]Glossary of Chemical Formulae, https://en.wikipedia.org/wiki/Glossary_of_chemical_formulae

Chapter 4

Evaluation

This chapter reviews the methods used in evaluating ”Let’s Get Cracking”. The findings are presented in the subsequent chapter. The evaluation of the game involved a questionnaire and the recording of participants’ gameplay for observation. Participants’ understanding of password security was assessed through a questionnaire completed in two parts: one before and one after playing the game. Many of the questions asked in the first part of the questionnaire were asked again in the second part, to observe how participants’ responses were changed as a result of playing the game. No demographic information was collected during the experiment, and participants were exclusively identified using a randomly assigned two-digit participant number.

4.1 Opinion-Based Questions

The following opinion-based questions were asked before and after playing the game:

- How confident do you feel you can identify the characteristics of a weak password?
- In the event of a large-scale password data breach, would the strength of your password play a role in its ability to remain secure?
- Approximately how many guesses do you think your password needs to be able to withstand for it to be considered secure?

These questions were chosen to observe how the game would affect participants’ understanding of the attack model and password characteristics.

4.2 Knowledge-Based Questions

Similarly, the following knowledge-based questions were asked before and after playing the game (with the exception of the first question which was not expected to change).

- Have you heard about password data breaches in the news or through other sources?

- Briefly describe how you think a website securely stores passwords for user authentication.
- Briefly describe how you think attackers guess passwords.
- What is a hash in the context of computer security and cryptography?

These questions were asked to compare participants' baseline understanding of password storage protocols and password cracking with their understanding after playing the game. They also serve as a method to identify participants who already have pre-existing knowledge of computer security.

4.3 Password Strength Identification

The questionnaire includes two sections dedicated to rating password strength to corroborate participants' identification of password strength characteristics. The first of these asks participants to rate the strength of 12 passwords on a five point scale from "Very weak" to "Very strong". The second section asks participants to compare the strength of two passwords on a seven-point scale from "Password 1 is much more secure" to "Password 2 is much more secure".

The passwords and password pairs were chosen to compare how user perception of various password characteristics changed before and after playing the game. These include length, character type variety, the use of dictionary words, keyboard patterns, predictable character placement, and "common l33t". The tables in Appendix C.2 indicate which characteristic each password is specifically testing. The order of the passwords was shuffled for each section to ensure this did not affect participants' responses.

The rating of password strength is the primary gameplay element of the serious game PASDJO, discussed in section 2.2.4. As was the case in their study, participants' strength ratings would be compared with the strength ratings for the same passwords from the Python library zxcvbn.

zxcvbn is a password strength estimator that returns strength estimations based on techniques used by password crackers. These techniques include taking characteristics into account that other password strength estimators which employ LUDS, do not. This includes identifying predictable character positions, the use of dictionary words, keyboard patterns, and similarity to existing cracked passwords.

zxcvbn rates passwords by estimating the number of guesses required to crack them. If the guess estimate is greater than 10^{10}, it is considered "Very strong". If it is in between 10^8 and 10^{10}, it is considered "Somewhat strong". If it is in between 10^6 and 10^8, it is considered "Neither weak nor strong". If it is in between 10^3 and 10^6, it is considered "Somewhat weak". Otherwise, if it is guessable within 10^3 guesses it is considered "Very weak".

The use of zxcvbn's strength ratings as an objective measure aims to evaluate the game's performance through a quantitative result which can be compared directly with the findings from the Seitz and Hussmann (2017) study.

Password pair strength comparison was also conducted in the Ur et al. (2016) study, of which many password pairs were taken or inspired by the selection used there.

A similar metric was used to generate expected ratings for password pair strength estimation. Unfortunately, utilising zxcvbn to provide a definitive objective rating for password pair strength comparison did not always yield accurate results. The attempted method involved comparing zxcvbn's estimation of the number of ”guesses” required to crack each password within the pair. However, zxcvbn was not designed for this comparative analysis as it does not account for subtle variations in password complexity that can significantly impact security. zxcvbn is much better suited to give a rough approximation of the order of magnitude of guesses required to crack a password.

For instance, zxcvbn assigned the same guess count to password pairs such as ”appleton16” and ”appletonqy” or ”scotland1” and ”1scotland.” According to more nuanced security assessments, this is a misrepresentation. This is most likely due to the fact that zxcvbn leans heavily on data from previously cracked passwords, rather than analysing password complexity factors.

Therefore, some zxcvbn ratings were adjusted. These include ”appleton16 vs appletonqy”, ”scotland1 vs 1scotland”, and ”Broccolihead1 vs Brocolliheadw”, which were all adjusted from ”Both passwords are equally secure” to ”P2 is slightly more secure”. The rest of the ratings were obtained using the following metric inspired by the metric used for rating singular password strength. If the number of guesses of a password is 10 times greater than another, it is considered ”slightly more secure”. If it is 10^3 times greater, it is considered ”more secure”. If it is 10^6 times greater, it is considered ”much more secure”.

Ultimately, the inaccuracies in zxcvbn's approximations do not negatively affect the study, as the most important factor being observed through these questions is how password characteristic perception changes, regardless of an objectively correct rating.

4.4 Reflection on the Game

These questions were asked in the second section of the questionnaire after participants had finished playing the game.

- How fun did you find the game?
- Did the game help you identify any weaknesses in your current password practices?
- What aspects of password security surprised you the most while playing the game?
- How much did assuming the role of the adversary contribute to your understanding of secure password practices?
- How likely are you to implement any of the practices you learned from the game in your daily life?

- How likely would you be to recommend this game as a tool for teaching password security?
- What aspects of the game, if any, do you think could be improved to enhance its effectiveness in teaching password security concepts?
- Is there anything else you would like to share about your experience with the game?

These questions are designed to provide insight into the game's effectiveness in terms of its educational impact, its encouragement of long-term secure password practice adoption, player engagement and enjoyment.

Chapter 5

Results

This chapter analyses the results obtained from observed gameplay and the questionnaire. The study concluded with 22 participants in total. As outlined in the previous chapter, participants were asked to complete two sections of a questionnaire: one before and one after playing the game. Each participant was anonymised and asked to identify themselves using a designated participant number, which was assigned upon reading the participant information sheet and signing the participant consent form (Appendix A and B).

No demographic information was collected during this study, however as a result of my own position as an undergraduate student within the school of informatics, many participants were colleagues with similar backgrounds. This could potentially have biased the results, as these participants may know more about the relevant security subjects than the average person.

5.1 Observations from Gameplay

Overall, reactions to the game were extremely positive. Participants spent an average of approximately 24 minutes playing the game, all of which reached the final free-play level. This is a strong indication that participants were highly engaged while playing the game.

In particular, the game's leaderboard system was highly successful in motivating participants. Within each level, the high score value indicated at the top right of the interface provided players with an attainable goal to reach. Players would often celebrate having beaten the high score or reflect on potential strategies they may have missed when scoring relatively poorly.

The game's technical performance was good. Participants played it on both Windows and Linux platforms. In both cases, the game never crashed, and participants did not encounter any game-breaking bugs. This is an impressive feat considering the number of gameplay systems in place and indicates the quantity of care and polish that went into development.

Participants were also very impressed by the game's presentation. The music, sound effects, visual aesthetics, and animations were complimented throughout the evaluation process and were never described as distracting.

While these observations from gameplay were encouraging, the evaluation also revealed unexpected observations which negatively impacted participants' experience with the game.

5.1.1 Unexpected Observations from Recorded Gameplay

5.1.1.1 Overlooking Simpler Strategies

Many participants opted for complex input sequences without first exploring simpler, single-character inputs. This behaviour may have arisen from modern password creation policies that employ LUDS, which influenced participants' inputs even when such complexity was not necessary. The game's encouragement of experimentation, with the assumption that players are familiar with common password patterns, may have contributed to this.

5.1.1.2 Overlooking Hash Quantities

Players would often not pay attention to the number of hashes an input sequence would generate, or the amount of in-game time it would take to generate. This may be attributed to the game's UI density and colour choices that made the hash counts less noticeable. The hash count is also rendered somewhat obsolete by the timer UI elements, as time is a much more digestible interpretation. Due to the game's encouragement of input sequences of minimal combinational complexity, players rarely inputted sequences with a hash count greater than 10^{10}, or one second of in-game time. All sequences that would take less than a second of in-game time are said to be generated "Instantly" in the timer UI element, as is shown in Figure 3.9. This may have given some players the impression that all passwords found through input sequences of this complexity were equally secure.

This could be addressed by playing an animation which scales the hash count each time it is updated, drawing players' attention. The colour of the hash count could also change on a gradient based on the complexity of the input sequence.

5.1.1.3 Underutilisation of Word Options

Players did not engage with the word options menu as much as intended. This can be attributed to the game failing to describe the disadvantage of simply enabling all word types, as the menu provides no indication of how many words would be added for each word and, therefore, how hash complexity will increase. As there was little to discourage doing so, participants would often enable all word type categories. This made it difficult to observe and analyse how common the use of each individual word type category is and, therefore, could not come to conclusions about the security risks using them in password poses.

This could be addressed by discouraging this behaviour more substantially and introducing each word type gradually over the course of the game.

5.1.1.4 Underutilisation of ”Any Character” Button

Players did not use the ”Any Character” button as often as expected. This could be due to its placement to the far right of the crowded interface. This highlights a need for better UI clarity. This could be addressed by separating the interface into distinct sections with background elements and employing additional colours that do not thematically ruin the game's aesthetic.

5.1.1.5 Dialogue Interaction Challenges

Players would occasionally skip over parts of the game's dialogue, missing important information about how to use new password-cracking tools and the level's password requirements. This problem was exacerbated by the fact that there was no back button for the dialogue. The only way to read something they had missed was to press the ”C1PH3R” button at the top right of the screen and restart the level's dialogue from scratch. This highlights a need for improved control within the game's dialogue system.

5.2 Questionnaire Responses

5.2.1 Opinion-Based Questions

5.2.1.1 How confident do you feel you can identify the characteristics of a weak password?

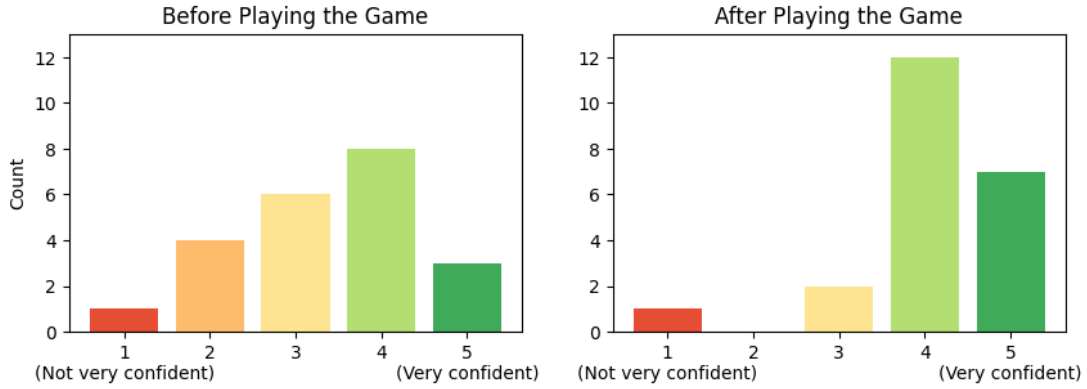


Figure 5.1: Change in Participants' Confidence Regarding Characteristics of Weak Passwords

Figure 5.1 shows how participants' confidence in identifying characteristics of weak passwords increased after playing the game. The average confidence rating increased from 3.36 to 4.09. This suggests that the players found the adversarial perspective useful in disambiguating gaps in their knowledge.

5.2.1.2 In the event of a large-scale password data breach, would the strength of your password play a role in its ability to remain secure?

Results show that participants were more likely to correctly identify that password strength does play a role in remaining secure during an offline password database attack after playing the game.

Before playing the game, 50.0% of participants selected ”Yes”, 9.1% of participants selected ”No”, and 40.9% of participants selected ”Not Sure”. After playing the game, 95.5% of participants selected ”Yes”, and a single participant, representing 4.5% of all participants, selected ”No” (Figure C.2).

This substantial increase in the number of participants who understood the importance of strong passwords suggests that the game was effective in educating and informing players about this concept. The game likely helped participants appreciate how much more difficult it was to crack more complex passwords and how much more effective and rewarding it was to exploit weaker password patterns instead.

5.2.1.3 Approximately how many guesses do you think your password needs to be able to withstand for it to be considered secure?

According to Ur et al. (2016), secure passwords should be able to withstand approximately 10^{14} to 10^{20} guesses to be considered secure, based on the speed of the hashing function utilised.

When asked this question, participants provided a large range of answers. Figure 5.2 categorises participants' responses to the closest order of magnitude.

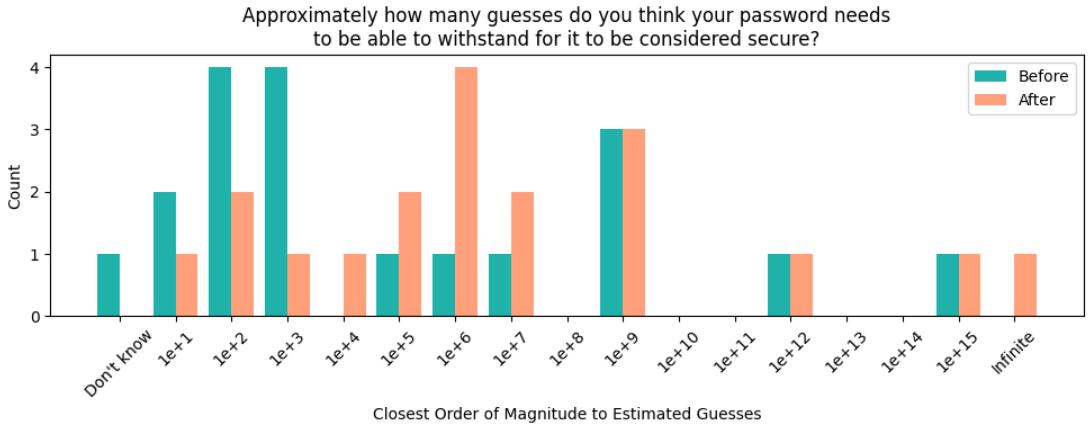


Figure 5.2: Change in participant perception of the number of guesses a secure password should be able to withstand.

This same question was also explored in the Ur et al. (2016) study with 165 participants. Their findings showed that 34% of participants estimated a quantity less than 50, and 67% of participants estimated a quantity less than 50,000. In comparison, before playing the game 23% of participants estimated less than 50, and 55% of participants estimated less than 50,000. However, after playing the game, only a single participant (4.5%) chose less than 50, and just 18% of participants chose less than 50,000.

While a significant increase in response values is notable, only two participants identified a value within the range specified by Ur et al. (2016) after playing the game. This may be attributed to the game's short time limits which encourage the player to input sequences of low complexity. This is exacerbated by the observation made in section 5.1.1.2, where players did not pay very much attention to the "Number of Hashes" counter.

This result may also be due to the question's phrasing. It may have been unclear to participants that a guess is equivalent to the generation of a single hash. It may have been assumed that a guess instead referred to a single in-game input sequence, which would generate a large number of hashes.

5.2.2 Knowledge-Based Questions

5.2.2.1 Have you heard about password data breaches in the news or through other sources?

90.5% of participants selected "Yes", indicating they had previously heard of password data breaches. This indicates a clear awareness among the majority of participants about the existence of this security risk. The finding suggests the gap in public understanding is likely less about a lack of general awareness of password breaches, and more a result of misunderstandings about the attacker model and password storage practices, which is exactly what "Let's Get Cracking" addresses.

5.2.2.2 Briefly describe how you think a website securely stores passwords for user authentication

The expected answer for this question would have been one that mentions a database storing hashed passwords. Before playing the game, participants answered with a variety of answers. 40.9% of participants correctly mentioned storing hashes, including 18.2% of participants who correctly mentioned salting, which is not taught in the game. This is an indication of how some participants' background in informatics may have affected results.

27.3% of participants identified encryption in general without specifically identifying hashes. The rest of the participants, 2 participants (9.1%), responded with a variation of a "Password-Protected Database", showing the prevalence of the misconception that a leaked password database would hold plaintext passwords and, therefore, password strength would not play a role in its security. After playing the game, all participants' responses included a reference to hashing or the storage of hashed passwords. This indicates the game's tutorial and mechanics were successful in educating participants on this topic, and that the game's gameplay mechanics and UI elements further supported this learning.

5.2.2.3 Briefly describe how you think attackers guess passwords.

The expected answer to this question would mention the matching of generated hashes created through brute-force attacks that target common patterns in passwords, including

dictionary attacks. In reality, attackers also use tools such as rainbow tables and existing cracked passwords, but as they are not included in the game, it was not expected participants would be able to identify these.

Before playing the game, 54.5% of participants correctly identified brute-force attacks, including 27.3% which further identified dictionary attacks, rainbow tables, and the exploitation of common password patterns. 22.7% answered that attackers also exploit personal information in their brute-force attacks. This shows a slight bias of perception towards more targeted attacks, rather than that of large-scale data breaches. After playing the game, 90.1% of participants correctly identified either brute-force attacks, dictionary attacks, or the exploitation of common password patterns. The other 9.1% simply stated a response about the generation of hashes.

This substantial improvement demonstrates the game was highly successful in enhancing participants' understanding of the attack model. This can be largely attributed to the game's tutorial content and the player's role as the adversary.

5.2.2.4 What is a hash in the context of computer security and cryptography?

A correct response to this question would identify a hash as a unique string generated from input data, in this case, a password, through an irreversible hash function.

Before playing the game, only 50% of participants identified a hash as an encrypted string, including 22.7% who identified the irreversible property of hashes. After playing the game, 95.5% of participants identified a hash as an encrypted string, including 27.7% who mentioned their irreversible property.

This significant increase in the correct identification of hashes shows the game's tutorial was successful in teaching this information through CiPH3R's dialogue and the helpful visuals and animations.

5.2.3 Password Strength Identification

5.2.3.1 Single Password Strength Rating

Figure 5.3 shows how participants evaluated the strength of 12 passwords before and after playing the game. The "Expected" password strengths were obtained from the zxcvbn Python library discussed in section 4.3. Figure 5.4 shows how the average distance from participants' responses to the expected values changed.

The largest deviations from the outcome, such as "6859147" and "Timothy1", can be attributed to a significant gap in strength estimation between the participants' consensus and that of zxcvbn. In this case, it appears that the error may lie with how zxcvbn calculates password strength, as described in section 4.3. Regardless of the validity of the objective value provided by zxcvbn, participants' perception of the strength of these passwords correlates with what is taught in the game.

Participants did appear to overestimate the strength of "n3!%Z" despite its very short length due to the variety of character types. This can be attributed to the lack of the use of the "Any Character" button in-game discussed in section 5.1.1.4, and the

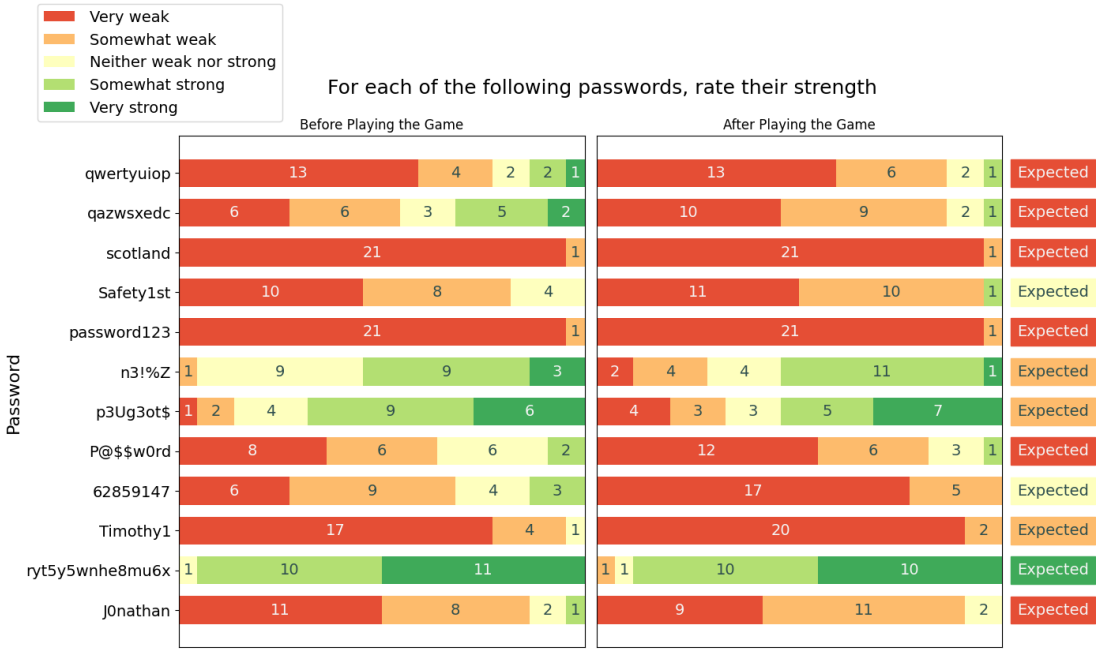


Figure 5.3: Impact of gameplay on participants' perception of password strength

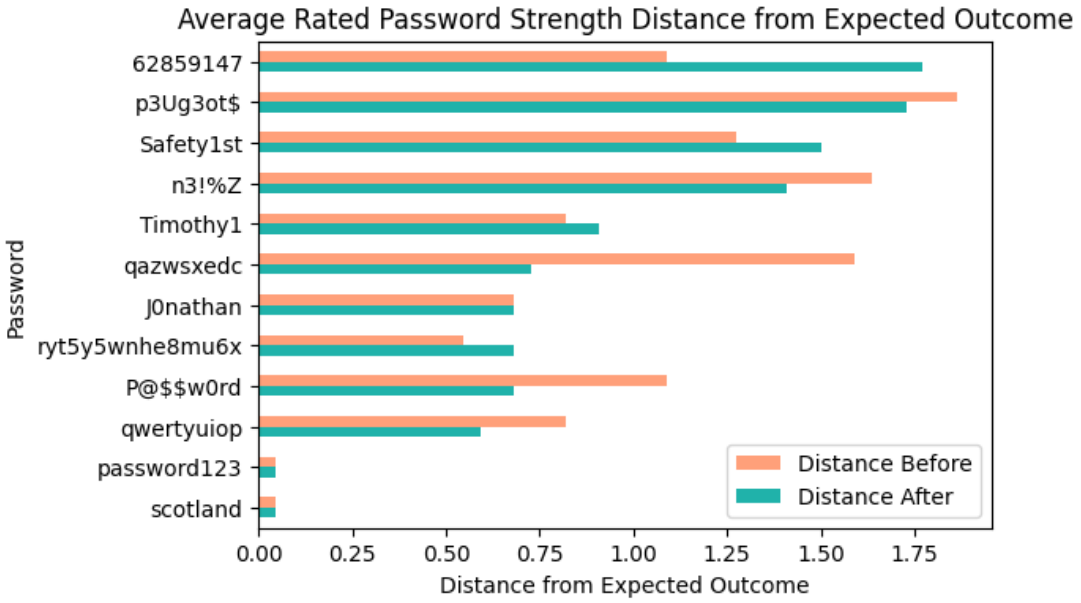


Figure 5.4: Average distance of participant responses to expected values before and after playing the game.

lack of levels where one could input a five-character sequence due to the password requirements.

Overall, the average distance for all passwords decreased from 0.96 to 0.90, but by adjusting the expected values of ”6859147” and ”Timothy1” to ”Very weak”, the average distance decreased from 0.92 to 0.70. This is a great result compared to the PASDJO study. Their study found that, from an original average distance of 1.4, the distance only decreased below 0.90 after 3 rounds of the game and below 0.7 after 9 rounds, which very few participants were willing to play to. (Seitz and Hussmann, 2017).

The results show the game was successful in decreasing participants' perception of the strength of using keyboard patterns (”qwertyuiop”, ”qazwsxedc”), common l33t (”P@$$w0rd”), and predictable character placement (”Timothy1”).

5.2.3.2 Password Pair Strength Comparison Ratings

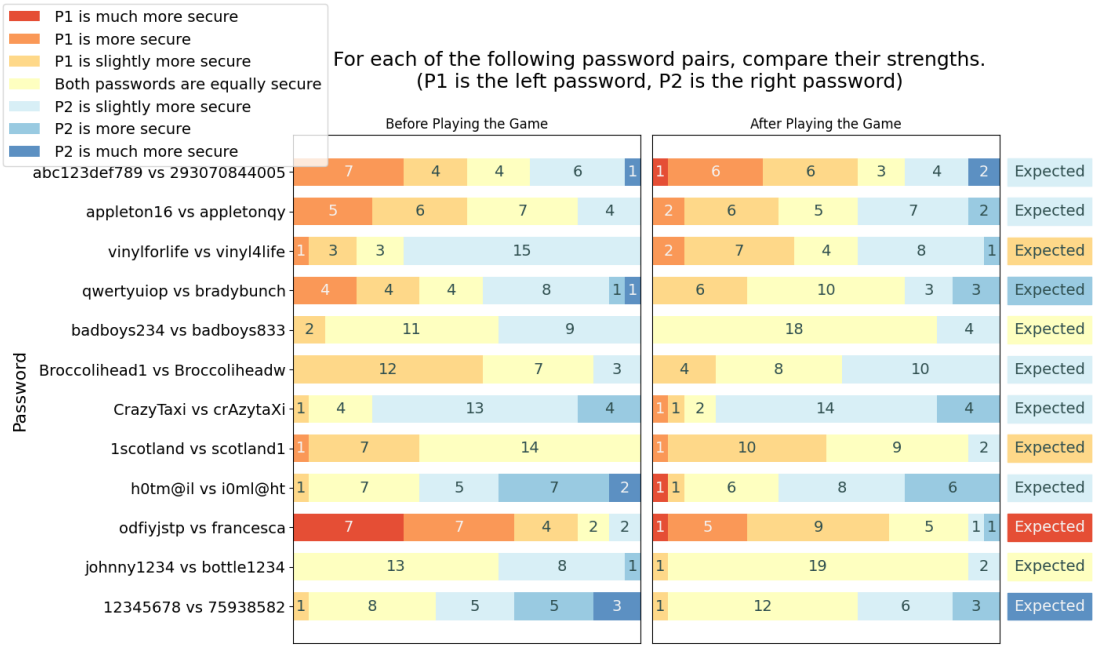


Figure 5.5: Impact of gameplay on particpants' perception of the comparison of password pair strength

Figure 5.5 shows how participants compared the strength of 12 password pairs on a 7-point scale from ”Password 1 is significantly stronger” to ”Password 2 is significantly stronger”. Figure 5.6 shows how the distance of participants' average rating compared to the ”Expected Values” changed.

The results show that the game helped participants identify the weakness of predictable character placement (”1scotland vs scotland1”) and of the use of numbers instead of letters (”appleton16 vs appletonqy”, ”vinylforlife vs vinyl4life”, and ”Broccolihead1 vs Broccoliheadw”). Players struggled to identify the relative weakness of keyboard patterns in ”12345678 vs 75938582”, ”abc123def789 vs 293070844005” and ”qwertyuiop vs bradybunch”, and the weakness of using a common name over random letters

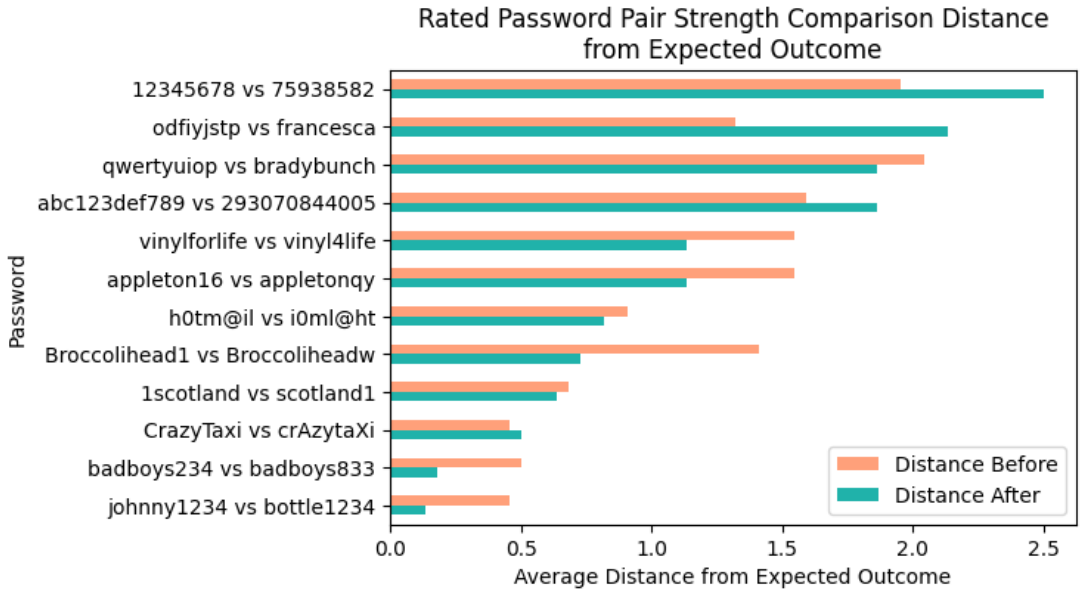


Figure 5.6: Change in average distance of participants' pair strength comparison ratings

in "odfiyjstp vs francesca". This can be attributed to the lack of acknowledgement of the "Hash Number" counter (section 5.1.1.2), and the lack of experimentation with the word options (section 5.1.1.3).

Overall, the average rating distance for all passwords decreased from 1.20 to 1.14. While this is a positive result, it indicates areas where the game could improve.

5.2.4 Reflection on the Game

5.2.4.1 How fun did you find the game?

Participants were asked to rate how fun they found the game on a five-point scale from "Not fun" to "Very fun". 45.5% selected the highest rating of 5, 50.0% of participants selected 4, and one participant (4.5%) selected 3. Overall, the average rating of fun was 4.41. This enthusiastic response shows the game was successful in teaching players in an entertaining and engaging manner.

5.2.4.2 Did the game help you identify any weaknesses in your current password practices?

86.4% of participants selected "Yes", indicating the game was successful at helping them identify weaknesses in their understanding of weak password characteristics.

5.2.4.3 What aspects of password security surprised you the most while playing the game?

Participants remarked on their disbelief at the ease and effectiveness of password cracking. This included responses about the rapid speed of hash generation, the large quantity of weak and predictable passwords, and the effectiveness of dictionary attacks

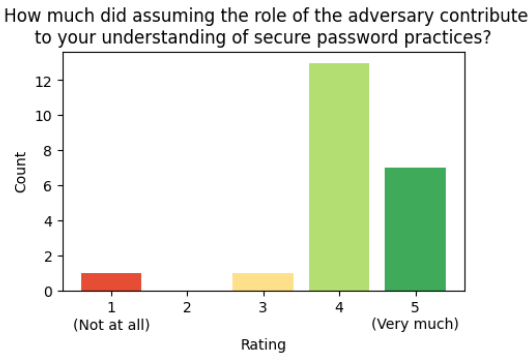


Figure 5.7: Participants' perception of the importance of the adversarial role in "Let's Get Cracking"

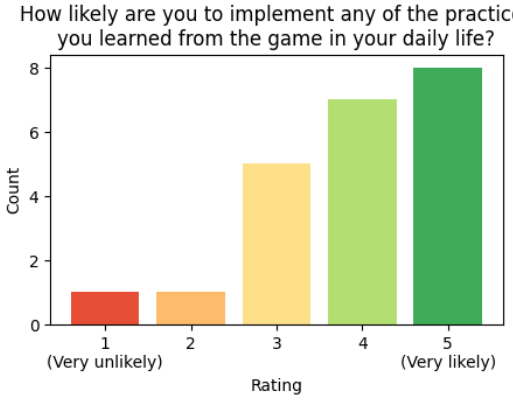


Figure 5.8: Likelihood of future implementation of learned practices

and common l33t. These insights are a direct result of the participants' role as the adversary in-game.

5.2.4.4 How much did assuming the role of the adversary contribute to your understanding of secure password practices?

Participants were asked this question on a 5-point scale from "Not at all" to "Very much". Figure 5.7 shows the result of this. It shows a very strong response with an average rating of 4.14.

This feedback overwhelmingly suggests the adversarial perspective employed in "Let's Get Cracking" was highly effective in educating participants about the real-world attack model used in compromised password databases. By directly experiencing this perspective, players were able to learn about various password cracking tools, and how to best use them to exploit common patterns in passwords, allowing them to understand the importance of password security measures.

5.2.4.5 How likely are you to implement any of the practices you learned from the game in your daily life?

Figure 5.8 shows participants would be highly likely to implement the practices learned from the game in their daily lives, with an average rating of 3.91 out of 5, and with 36% of participants selecting the highest rating. This is a very positive result, indicating how the game's encouragement of player freedom and intuitive learning helped the participants acquire an actionable understanding of password security.

5.2.4.6 How likely would you be to recommend this game as a tool for teaching password security?

Participants were asked to answer this question on a five-point scale ranging from "Very Unlikely" to "Very Likely". 77% of participants selected the highest rating of 5, and the other 23% selected a rating of 4, resulting in an average rating of 4.81. This is an extremely strong indicator that participants were extremely satisfied with the game in teaching password security concepts in an enjoyable and entertaining fashion.

5.2.4.7 What aspects of the game, if any, do you think could be improved to enhance its effectiveness in teaching password security concepts?

Of the 12 participants who replied to this optional question, a variety of improvements were suggested. These included quality-of-life improvements such as the ability to go backwards in the dialogue sequence, and an input sequence history page to keep track of what sequences had been inputted already. Suggestions were made regarding the game's accessibility. In particular, it was suggested that players should be able to adjust the size of the dialogue text and include a voice-over of the content. This could be included through text-to-speech which could be enabled in an options menu. Participants' also expressed they wished the game did more to suggest how to create stronger passwords. It should have been discussed how best to generate memorable passwords which do not conform to patterns that could be exploited by password crackers.

5.2.4.8 Is there anything else you would like to share about your experience with the game?

Of the 11 participants who responded to this optional question, they all reiterated how much they enjoyed the game. 6 participants expressed how much they loved the game and how enjoyable they found it. 3 participants complimented the game's graphics, music, and sound effects. The other 2 participants expressed how great of a teaching tool they found it to be, and strongly suggested it should be distributed online. While these results may be biased due to the fact many participants were close colleagues, the responses indicate extremely high enthusiasm for the game.

Chapter 6

Conclusions

6.1 Overview

The development and evaluation of ”Let’s Get Cracking” resulted in several key insights. The goal of this project was to observe if a serious game which simulates password cracking from an adversarial perspective could help reduce the gap between the public understanding and the reality of password security in the context of data breaches. The development of the game was conducted successfully by addressing criticisms about password security identified in existing serious games. The evaluation process showed that the game left an extremely positive impression on participants, successfully educating them on password security misconceptions in an entertaining and engaging way. The evaluation process also identified areas the game could improve on, both in terms of gameplay, and in terms of educational content.

6.2 Results Achieved

The results of the evaluation process were extremely positive. The game successfully taught important password security concepts, such as hashes and brute-force attacks, and cleared common misconceptions about password security, such as the importance of strength in a data breach.

Overall, the results showed participants had a much better understanding of the attack model after playing the game than they did before. This can be attributed to the adversarial perspective used in the game, which helped participants understand the human element behind password cracking, and how password-cracking tools can exploit the patterns in weak passwords. This had a great impact on participants, who were surprised by the ease and speed with which passwords can be cracked. The overwhelming majority of participants acknowledged that the game helped them identify weaknesses in their current password practices and that they would employ what they learned from the game in their daily lives.

The evaluation process did help identify areas of the game that can be improved. These include improvements to the gameplay experience and improvements to reinforce as-

pects of password security, where a noticeable gap still existed between the participants' understanding and reality.

6.3 Future Extensions and Improvements

6.3.1 Adjustment of Time Limits

While the game's time limits successfully ensured participants would not spend inordinate amounts of time on a single level, there are still flaws in the system. For one, it highly discouraged experimentation with input sequences of high complexity, resulting in the vast majority of input sequences being extremely simple.

To encourage player experimentation with longer time limits, whilst still limiting their time to one level, a different level goal may be implemented. For example, future versions of the game may experiment with a score threshold that must be reached, or a limit on the number of times players can "Generate Hashes". Adjustments to the time system could also be addressed by adjusting the speed of hashing through new gameplay mechanics. These could include systems which mimic real defensive procedures such as the use of slow hash functions and salting.

6.3.2 Slower Introduction of Concepts

Observations from gameplay, such as the minimal use of word type customisation and the "Any Character" button, can be attributed to the fact the game did not give enough attention to these elements. In the case of "Any Character", it is introduced at the very beginning of the game while the player is still processing the basic gameplay mechanics. More players would have used it if it was introduced and explained at a later point in the game. The same applies for the various word types, as they are all introduced together, little emphasis is put on a single one. As the game is already rather lengthy, the game could introduce these during a level, after reaching a certain threshold or by unlocking them.

For example, the different word types used in dictionary attacks could be unlocked or purchased using in-game currency. This would encourage players to think critically about the utility of each individual word type before purchasing and utilising them. This would further deepen the gameplay mechanics and increase the value of the rewards given to the player.

6.3.3 Help Players that Overlooked Simpler Strategies

As described in section 5.1.1.1, players would often overlook simpler password input sequences that would have achieved very high scores. This could be addressed with an adaptive difficulty feature where the game could identify if the player is struggling and have CiPH3R suggest patterns the player may have missed.

Another way to address this could be to show the most popular passwords the player did not crack in each level's results screen. This would help players critically evaluate

the common patterns in these passwords and identify how they could exploit them in future levels.

This problem could also be addressed more organically with an extension which would introduce players to rainbow tables.

6.3.3.1 Rainbow Tables

Rainbow tables are a very important part of password cracking and are among the largest security concept omissions in the game. In this proposed system, players would be able to purchase rainbow tables. Each rainbow table would represent a different precomputed set of hashes of a particular input sequence (for example, a rainbow table containing the hashes for all passwords consisting of eight lowercase letters). The advantage of purchasing rainbow tables rather than generating the hashes oneself would be that rainbow tables do not consume time. Players would be able to choose from a selection of rainbow tables, therefore exposing them to input sequence patterns they may not have originally considered, allowing them to think critically about which patterns best exploit common password vulnerabilities. This would successfully deepen the game's mechanics, improve the game's reward system, and effectively introduce a new password security concept to players.

6.3.4 Password Creation Suggestions

As "Let's Get Cracking" is played from the adversarial perspective, it does little to tell users how best to defend their own passwords. Some participants expressed they wished the game offered more advice and guidance in this respect. For one, the game could suggest the best ways to create memorable but secure passwords. It could also suggest other secure password creation methods, such as the use of password managers and the advantages and disadvantages associated with them.

The game should also discuss how one can keep their data safe in the event their details are exposed in a data breach. This is briefly mentioned in the free-play level, but could be further expanded. One should immediately change their password and ensure they are not reusing the same password elsewhere on the internet. They should also ensure passwords are not the only line of defense for their digital accounts, and should employ the use of measures such as two-factor authentication. While this could be explicitly stated in the game's dialogue, it could also be incorporated through gameplay. For example, the results screen could indicate how many of the cracked accounts found by the player are inaccessible due to these measures.

Bibliography

Per Backlund and Maurice Hendrix. Educational games - are they worth the effort? a literature survey of the effectiveness of serious games. In *2013 5th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, pages 1–8, 2013. doi: 10.1109/VS-GAMES.2013.6624226.

Ivan L. Beale, Pamela M. Kato, Veronica M. Marin-Bowling, Nicole Guthrie, and Steve W. Cole. Improvement in cancer-related knowledge following use of a psychoeducational video game for adolescents and young adults with cancer. *Journal of Adolescent Health*, 41(3):263–270, 2007. ISSN 1054-139X. doi: https://doi.org/10.1016/j.jadohealth.2007.04.006. URL https://www.sciencedirect.com/science/article/pii/S1054139X07001759.

Polona Caserman, Katrin Hoffmann, Philipp Müller, Marcel Schaub, Katharina Straßburg, Josef Wiemeyer, Regina Bruder, and Stefan Göbel. Quality criteria for serious games: Serious part, game part, and balance. *JMIR Serious Games*, 8(3):e19037, Jul 2020. ISSN 2291-9279. doi: 10.2196/19037. URL http://games.jmir.org/2020/3/e19037/.

Patrick Gage Kelley, Saranga Komanduri, Michelle L. Mazurek, Richard Shay, Timothy Vidas, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Julio Lopez. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *2012 IEEE Symposium on Security and Privacy*, pages 523–537, 2012. doi: 10.1109/SP.2012.38.

Fedwa Laamarti, Mohamad Eid, and Abdulmotaleb El Saddik. An overview of serious games. *Int. J. Comput. Games Technol.*, 2014, jan 2014. ISSN 1687-7047. doi: 10.1155/2014/358152. URL https://doi.org/10.1155/2014/358152.

Sarah Pearman, Shikun Aerin Zhang, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Why people (don't) use password managers effectively. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*, pages 319–338, Santa Clara, CA, August 2019. USENIX Association. ISBN 978-1-939133-05-2. URL https://www.usenix.org/conference/soups2019/presentation/pearman.

Prabaharan Poornachandran, M. Nithun, Soumajit Pal, Aravind Ashok, and Aravind Ajayan. Password reuse behavior: How massive online data breaches impacts personal data in web. In H. S. Saini, Rishi Sayal, and Sandeep Singh Rawat, editors, *Innovations in Computer Science and Engineering*, pages 199–210, Singapore, 2016. Springer Singapore. ISBN 978-981-10-0419-3.

George E. Raptis, Christina Katsini, Andrew Jian-lan Cen, Nalin Asanka Gamagedara Arachchilage, and Lennart E. Nacke. Better, funner, stronger: A gameful approach to nudge people into making less predictable graphical password choices. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445658. URL https://doi.org/10.1145/3411764.3445658.

Tobias Seitz and Heinrich Hussmann. Pasdjo: Quantifying password strength perceptions with an online game. In *Proceedings of the 29th Australian Conference on Computer-Human Interaction*, OzCHI '17, page 117–125, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450353793. doi: 10.1145/3152771.3152784. URL https://doi.org/10.1145/3152771.3152784.

Wayne Summers and Edward Bosworth. Password policy: The good, the bad, and the ugly. pages 1–6, 01 2004.

Penelope Sweetser and Peta Wyeth. Gameflow: A model for evaluating player enjoyment in games. *Computers in Entertainment*, 3:3, 07 2005. doi: 10.1145/1077246.1077253.

Harshal Tupsamudre, Rahul Wasnik, Shubhankar Biswas, Sankalp Pandit, Sukanya Vaddepalli, Aishwarya Shinde, C. J. Gokul, Vijayanand Banahatti, and Sachin Lodha. Gap: A game for improving awareness about passwords. In Stefan Göbel, Augusto Garcia-Agundez, Thomas Tregel, Minhua Ma, Jannicke Baalsrud Hauge, Manuel Oliveira, Tim Marsh, and Polona Caserman, editors, *Serious Games*, pages 66–78, Cham, 2018. Springer International Publishing. ISBN 978-3-030-02762-9.

Blase Ur, Jonathan Bees, Sean M. Segreti, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. Do users' perceptions of password security match reality? In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, page 3748–3760, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450333627. doi: 10.1145/2858036.2858546. URL https://doi.org/10.1145/2858036.2858546.

Daniel Lowe Wheeler. zxcvbn: Low-Budget password strength estimation. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 157–173, Austin, TX, August 2016. USENIX Association. ISBN 978-1-931971-32-4. URL https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentatio

Appendix A

Participants' Information Sheet

Participant Information Sheet

Project title:	Cybersecurity Game – Teaching password strength
Principal investigator:	Borislav Ikonomov
Researcher collecting data:	Dylan Drucker
Funder (if applicable):	

This study was certified according to the Informatics Research Ethics Process, reference number **214977**. Please take time to read the following information carefully. You should keep this page for your records.

Who are the researchers?

For this undergraduate research project, the student leading the research is Dylan Drucker. The principal supervisor for this project is Borislav Ikonomov. Jingjie Li is an additional supervisor on the project.

What is the purpose of the study?

The purpose of this study is to research how a game can educate users about offline password database attacks and the importance of strong passwords. Participants will be asked to play a short game and answer two questionnaires, one before and one after the game. The study will observe how playing the game improved participant's knowledge in this area.

Why have I been asked to take part?

The target group of participants for this research project are participants with varying degrees of familiarity regarding offline password database attacks.

Do I have to take part?

No – participation in this study is entirely up to you. You can withdraw from the study at any time, up until you finish the second questionnaire without giving a reason. After this point, personal data will be deleted and anonymised data will be combined such that it is impossible to remove individual information from the analysis. Your rights will not be affected. If you wish to withdraw, contact the PI. We will keep copies of your original consent, and of your withdrawal request.

What will happen if I decide to take part?

You will be asked to play the game for up to 20 minutes and answer two questionnaires (one before and one after playing the game) which should take no more than 10 minutes per questionnaire. Your gameplay will be recorded. The questionnaire will include questions regarding your existing knowledge of offline password database attacks, your perceived strength of different passwords, and your thoughts on the game (including how it could be improved and if it improved your understanding of the concepts).

Are there any risks associated with taking part?

There are no significant risks associated with participation.

Are there any benefits associated with taking part?

No.

What will happen to the results of this study?

The results of this study may be summarised in published articles, reports and presentations. Quotes or key findings will be anonymized: We will remove any information that could, in our assessment, allow anyone to identify you. With your consent, information can also be used for future research. Your data may be archived for a maximum of 4 years. All potentially identifiable data will be deleted within this timeframe if it has not already been deleted as part of anonymization.

Data protection and confidentiality.

Your data will be processed in accordance with Data Protection Law. All information collected about you will be kept strictly confidential. Your data will be referred to by a unique participant number rather than by name. Your data will only be viewed by the researcher Dylan Drucker.

All electronic data will be stored on a password-protected encrypted computer, on the School of Informatics' secure file servers, or on the University's secure encrypted cloud storage services (DataShare, ownCloud, or Sharepoint) and all paper records

will be stored in a locked filing cabinet in the PI's office. Your consent information will be kept separately from your responses in order to minimise risk.

What are my data protection rights?

The University of Edinburgh is a Data Controller for the information you provide. You have the right to access information held about you. Your right of access can be exercised in accordance Data Protection Law. You also have other rights including rights of correction, erasure and objection. For more details, including the right to lodge a complaint with the Information Commissioner's Office, please visit www.ico.org.uk. Questions, comments and requests about your personal data can also be sent to the University Data Protection Officer at dpo@ed.ac.uk.

Who can I contact?

If you have any further questions about the study, please contact the lead researcher, Dylan Drucker (s2077148@ed.ac.uk), or the principal supervisor Borislav Ikonomov (borislav.ikonomov@ed.ac.uk).

If you wish to make a complaint about the study, please contact inf-ethics@inf.ed.ac.uk. When you contact us, please provide the study title and detail the nature of your complaint.

Updated information.

If the research project changes in any way, an updated Participant Information Sheet will be made available on http://web.inf.ed.ac.uk/infweb/research/study-updates.

Alternative formats.

To request this document in an alternative format, such as large print or on coloured paper, please contact Dylan Drucker (s2077148@ed.ac.uk).

General information.

For general information about how we use your data, go to: edin.ac/privacy-research

Appendix B

Participants' Consent Form

Participant number:_____

Participant Consent Form

Project title:	Cybersecurity Game – Teaching Password Strength
Principal investigator (PI):	Borislav Ikonomov
Researcher:	Dylan Drucker
PI contact details:	borislav.ikonomov@ed.ac.uk

By participating in the study, you agree that:

I have read and understood the Participant Information Sheet for the above study, that I have had the opportunity to ask questions, and that any questions I had were answered to my satisfaction.

- My participation is voluntary, and that I can withdraw at any time without giving a reason. Withdrawing will not affect any of my rights.
- I consent to my anonymised data being used in academic publications and presentations.
- I understand that my anonymised data will be stored for the duration outlined in the Participant Information Sheet.

Please tick yes or no for each of these statements.

1.	I agree to having my gameplay recorded.	**Yes** ☐	**No** ☐
2.	I allow my data to be used in future ethically approved research.	**Yes** ☐	**No** ☐
3.	I agree to take part in this study.	**Yes** ☐	**No** ☐

Name of person giving consent	Date dd/mm/yy	Signature

Name of person taking consent	Date dd/mm/yy	Signature

THE UNIVERSITY of EDINBURGH informatics

Appendix C

Questionnaire

Password Security Game Questionnaire

To be answered before playing the game

* Indicates required question

1. Please provide your designated participant number *

2. How confident do you feel you can identify the characteristics of a weak password? *

Mark only one oval.

	1	2	3	4	5	
Not	◯	◯	◯	◯	◯	Very confident

3. Have you heard about password data breaches in the news or through other sources? *

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

4. Briefly describe how you think a website securely stores passwords for user authentication *

5. In the event of a large-scale password data breach, would the strength of your password play a role in its ability to remain secure? *

Mark only one oval.

- Yes
- No
- Not sure

6. Briefly describe how you think attackers guess passwords *

7. What is a hash in the context of computer security and cryptography? (It is ok if you do not know) *

8. Approximately how many guesses do you think your password needs to be able to withstand for it to be considered secure? *

9. For each of the following passwords, rate their strength *

Mark only one oval per row.

	Very weak	Somewhat weak	Neither weak nor strong	Somewhat strong	Very Strong
qwertyuiop					
qazwsxedc					
scotland					
Safety1st					
password123					
n3!%Z					
p3Ug3ot$					
P@$$w0rd					
62859147					
Timothy1					
ryt5y5wnhe8mu6x					
J0nathan					

10. For each of the following password pairs, compare their strengths. (P1 is the left password, P2 is the right password) *

Mark only one oval per row.

	P1 is much more secure	P1 is more secure	P1 is slightly more secure	Both passwords are equally secure	P2 is slightly more secure	P2 is more secure	P2 is much more secure
abc123def789 vs. 293070844005	◯	◯	◯	◯	◯	◯	◯
appleton16 vs. appletonqy	◯	◯	◯	◯	◯	◯	◯
vinylforlife vs. vinyl4life	◯	◯	◯	◯	◯	◯	◯
qwertyuiop vs. bradybunch	◯	◯	◯	◯	◯	◯	◯
badboys234 vs. badboys833	◯	◯	◯	◯	◯	◯	◯
Broccolihead1 vs. Broccoliheadw	◯	◯	◯	◯	◯	◯	◯
CrazyTaxi vs crAzytaXi	◯	◯	◯	◯	◯	◯	◯
1scotland vs scotland1	◯	◯	◯	◯	◯	◯	◯
h0tm@il vs i0ml@ht	◯	◯	◯	◯	◯	◯	◯
odfiyjstp vs. francesca	◯	◯	◯	◯	◯	◯	◯
johnny1234 vs. bottle1234	◯	◯	◯	◯	◯	◯	◯
12345678 vs. 75938582	◯	◯	◯	◯	◯	◯	◯

Password Security Game Questionnaire Part 2

To be answered after playing the game

11. How confident do you feel you can identify the characteristics of a weak password? *

Mark only one oval.

	1	2	3	4	5	
Not	◯	◯	◯	◯	◯	Very confident

12. Briefly describe how you think attackers guess passwords *

13. What is a hash in the context of computer security and cryptography? *

14. Briefly describe how you think a website securely stores passwords for user authentication *

15. In the event of a large-scale password data breach, would the strength of your password play a role in its ability to remain secure? *

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

16. Approximately how many guesses do you think your password needs to be able to withstand for it to be considered secure? *

_____

17. How fun did you find the game? *

Mark only one oval.

	1	2	3	4	5	
Not	◯	◯	◯	◯	◯	Extremely fun

18. Did the game help you identify any weaknesses in your current password practices? *

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Other: _____

19. What aspects of password security surprised you the most while playing the game? *

20. How much did assuming the role of the adversary contribute to your understanding of secure password practices? *

Mark only one oval.

	1	2	3	4	5	
It di	◯	◯	◯	◯	◯	It contributed significantly

21. How likely are you to implement any of the practices you learned from the game in your daily life? *

Mark only one oval.

	1	2	3	4	5	
Extr	◯	◯	◯	◯	◯	Extremely likely

22. How likely would you be to recommend this game as a tool for teaching password security?

Mark only one oval.

	1	2	3	4	5	
Extr	◯	◯	◯	◯	◯	Extremely likely

23. What aspects of the game, if any, do you think could be improved to enhance its effectiveness in teaching password security concepts?

24. For each of the following passwords, rate their strength *

Mark only one oval per row.

	Very weak	Somewhat weak	Neither weak nor strong	Somewhat strong	Very Strong
qwertyuiop					
qazwsxedc					
scotland					
Safety1st					
password123					
n3!%Z					
p3Ug3ot$					
P@$$w0rd					
62859147					
Timothy1					
ryt5y5wnhe8mu6x					
J0nathan					

25. For each of the following password pairs, compare their strengths. (P1 is the left password, P2 is the right password) *

Mark only one oval per row.

	P1 is much more secure	P1 is more secure	P1 is slightly more secure	Both passwords are equally secure	P2 is slightly more secure	P2 is more secure	P2 is much more secure
abc123def789 vs. 293070844005	◯	◯	◯	◯	◯	◯	◯
appleton16 vs. appletonqy	◯	◯	◯	◯	◯	◯	◯
vinylforlife vs. vinyl4life	◯	◯	◯	◯	◯	◯	◯
qwertyuiop vs. bradybunch	◯	◯	◯	◯	◯	◯	◯
badboys234 vs. badboys833	◯	◯	◯	◯	◯	◯	◯
Broccolihead1 vs. Broccoliheadw	◯	◯	◯	◯	◯	◯	◯
CrazyTaxi vs crAzytaXi	◯	◯	◯	◯	◯	◯	◯
1scotland vs scotland1	◯	◯	◯	◯	◯	◯	◯
h0tm@il vs i0ml@ht	◯	◯	◯	◯	◯	◯	◯
odfiyjstp vs. francesca	◯	◯	◯	◯	◯	◯	◯
johnny1234 vs. bottle1234	◯	◯	◯	◯	◯	◯	◯
12345678 vs. 75938582	◯	◯	◯	◯	◯	◯	◯

26. Is there anything else you would like to share about your experience with the game?

C.1 Additional Figures

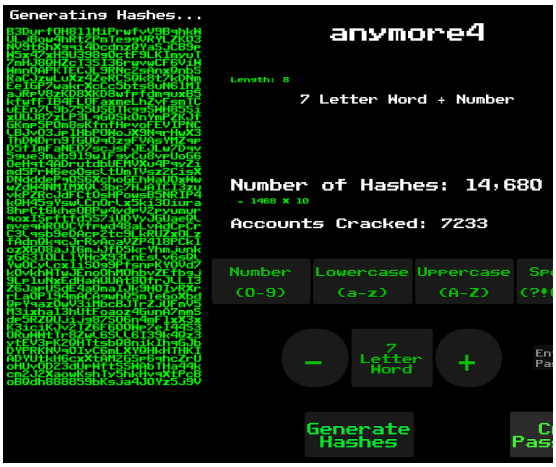


Figure C.1: Hash generation animation

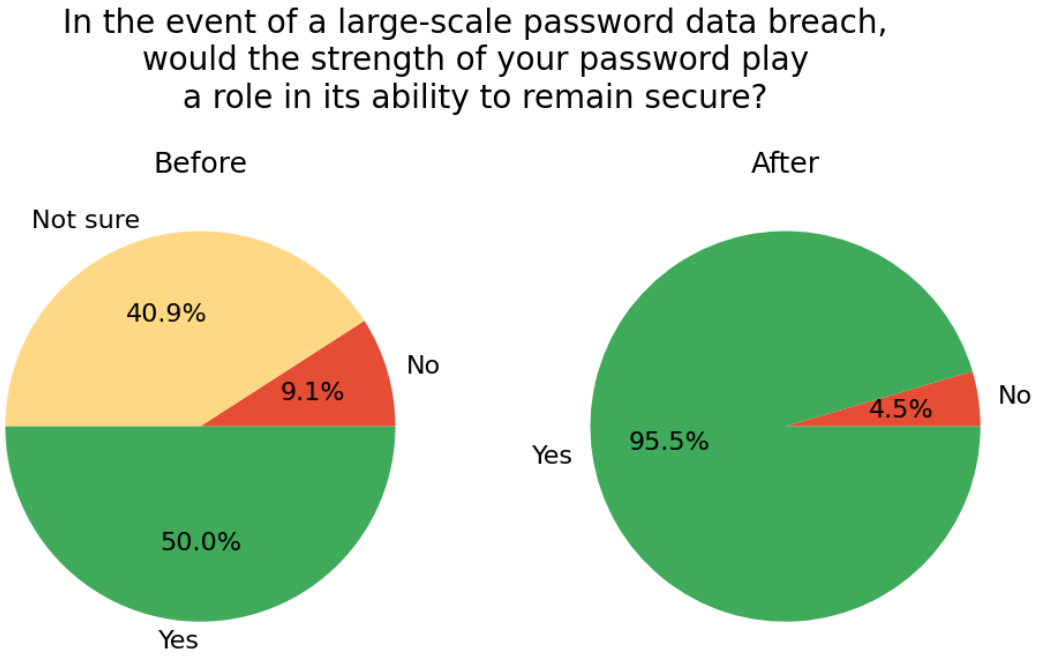
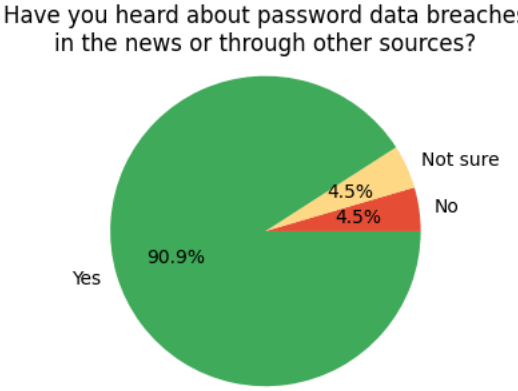
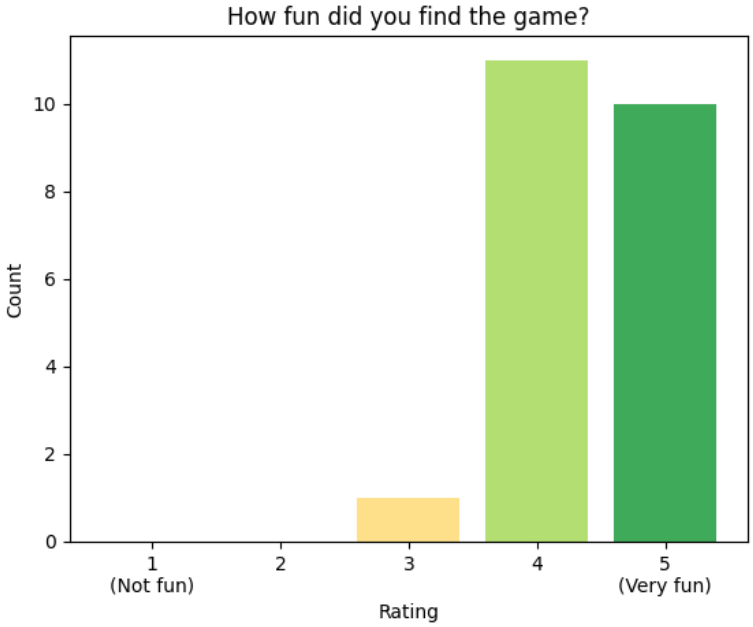
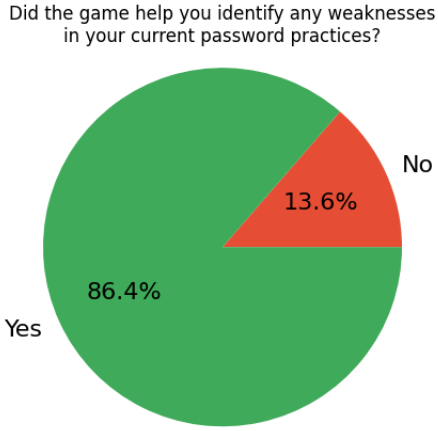


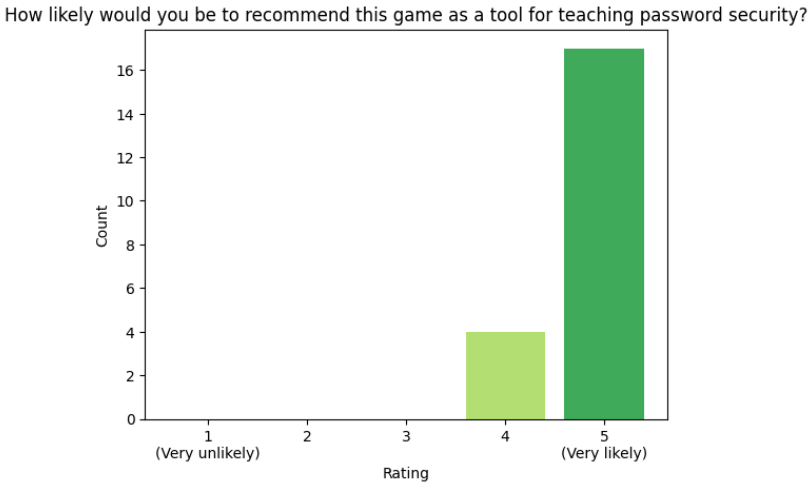
Figure C.2: Participant perception of password strength's role in offline data breaches

C.2 Password Strength Rating


Have you heard about password data breaches
in the news or through other sources?
Not sure
4.5%
No
4.5%
90.9%
Yes


How fun did you find the game?
Count
0
2
4
6
8
10
1
(Not fun)
2
3
4
5
(Very fun)
Rating


Did the game help you identify any weaknesses
in your current password practices?
No
13.6%
86.4%
Yes



Password	**Characteristic Tested**
qwertyuiop	Common Keyboard Pattern
qazwsxedc	Common Keyboard Pattern
scotland	Common word/place
Safety1st	Mix of Uppercase, Lowercase, and Numbers. Number in Unpredictable Location.
password123	Common Password
n3!%Z	Short Length despite mix of character types in unpredictable locations
p3Ug3ot$	Common L33t
P@$$w0rd	Common L33t of a very common word
62859147	Random, but single character type
Timothy1	Mix of character types, but in predictable locations and form a common name.
ryt5y5w-8mu6x	Long and contains a mix of alphanumeric characters
J0nathan	Common name + Common L33t

Table C.1: Password Strength Evaluation

Password Pair	**Characteristic Tested**
abc123def789 vs 293070844005	Usage of keyboard pattern vs random numbers
appleton16 vs appletonqy	Usage of random numbers vs letters at the end
vinylforlife vs vinyl4life	Usage of letters vs number
qwertyuiop vs bradybunch	Keyboard pattern vs two words
badboys234 vs badboys833	Different numbers at the end
Broccolihead1 vs Broccoliheadw	Number vs letter at the end
CrazyTaxi vs crAzytaXi	Placement of Uppercase characters
1scotland vs scotland1	Number at the beginning vs end
h0tm@il vs i0ml@ht	Common L33t vs random letters
odfiyjstp vs francesca	Random letters vs common name
johnny1234 vs bottle1234	Different base words
12345678 vs 75938582	Keyboard pattern vs random numbers

Table C.2: Password Strength Comparison

Appendix D

Let's Get Cracking Levels

D.1 Level Overview

Level	New Password Cracking Tool	Minimum Password Requirement	Time Limit (minutes)
Level 1	Number, Lowercase, Uppercase, Special, Any	None	5
Level 2	Custom String Input	Password Length ≥ 8	60
Level 3	Word Input (Dictionary Attack)	Same as Level 2 + must contain letters and numbers	5
Level 4	Word Options	Same as Level 3 + contains uppercase	5
Level 5	None	Same as Level 4 + contains special	5
Free Play	None	None	Infinite

Table D.1: Let's Get Cracking Level Progression

D.2 CiPH3R's Dialogue

D.2.1 Tutorial

Hey newbie, welcome to the team. (Press Enter to advance dialogue)

My name is CiPH3R. Together we are a group of cybercriminals that specialise in password cracking.

A fresh database has just leaked onto the web, so we are trying to crack as many of these passwords as we can so we can be the first to sell the account details on the dark web.

Uhhh you look a bit lost. You do know how to do a brute-force hashing attack on the database right??

Hmmm, alright let me try to explain...

Basically, to login to an account on the internet, you usually need a username and a password. These websites need some way to verify that the password you have typed in is a match with your actual password, so they need to store your password in their database somewhere.

Now, since this is an extremely sensitive piece of information, these passwords should not be stored as plaintext (its raw readable form), as anybody with access to this database would have access to everybody's accounts!

So instead these passwords are [b]hashed[/b] before being stored.

What is hashing? Hashing is a cryptographic technique which turns your nice readable password into an unreadable series of characters.

While it may look random, hashing the same password will always result in the same hash and no two passwords will have the same hash, each hash is unique.

The thing about hashing though? It is impossible to reverse. No computer in the world can reverse a hashed password in any reasonable amount of time.

The only way to "reverse" a hash would be to generate hashes from random passwords until you happen to find one that matches.

So, because of these nifty properties, website databases will store your hashed password rather than your plaintext password.

When you login, the website will hash your password, and compare the hash with the one stored in the database.

Since the hash is unique and can only be obtained by applying a hash function onto your password, the website is able to authenticate you.

This way, if the database is leaked, no one has access to the passwords themselves, only the hashes!

Which, if you remember, are basically impossible to reverse.

Ever notice when you click on "Forget your password?" the website always asks you to make a new one?

That's because even they don't actually know your password. All they have is the hash of the password.

So problem solved right? The only way to crack a hash would be to generate hashes from every possible password (of which there are infinitely many) and find the one that matches. It's impossible!

Well yes, this would be an impossible task, if humans weren't so predictable.

Most people don't make long and unpredictable passwords. Instead they usually make short predictable ones.

So instead of generating hashes for ALL POSSIBLE passwords, we can focus on generating hashes for the most common ones.

For example, you'd be surprised by the number of people who have passwords made up of 5 digits. (E.g. 12345, 00000, 99999, 67284, etc.)

You would only need to generate 100,000 hashes to be able to match all 5 digit passwords. That would take a computer no time at all!

Here let me show you!

D.2.2 Level 1

This is your password cracking interface!

Here you can generate hashes and match them with ones from our leaked database.

The buttons to the right allow you to specify what type of passwords you want to generate hashes for.

Currently, you can generate passwords with numbers, lowercase, uppercase, and special characters.

The "Generate Hashes" Button starts the process of generating hashes.

Doing so will use up your Time Remaining, which can be seen on the top-right.

Cracking longer passwords takes exponentially more time than shorter ones.

Then "Crack Passwords" allows you to match the generated hashes with the ones in the leaked database.

So let's say we want to crack all 5 digit passwords in the database.

Press the "Number" button 5 times. This will specify the type of password.

This will generate 100,000 hashes which any modern computer can do instantly.

Now press "Generate Hashes".

This has generated 100,000 hashes, one for each number from 00000 to 99999.

Now if anybody used a 5 digit number as their password...

The hash will be the same as one of the ones we have generated.

Press "Crack Passwords" to find these passwords.

On the left, you can see the password, and the number of accounts using it.

Told you this was easy!

Now let's try to crack all passwords made up of 5 lowercase letters and a number on the end.

Input the sequence to generate these hashes. (5 lowercase and a number).

Great! Now Generate Hashes...

And Crack Passwords!

Nice! Well done! You're a natural!

I think you're ready now, I'll let you do the rest on your own.

Try and crack as many passwords as you can before time runs out.

The clock's ticking.

Ok good luck!

D.2.3 Level 2

Welcome Back!

Hopefully you're a bit more comfortable with the interface now.

This new leaked password dataset only contains passwords containing AT LEAST 8 characters.

I've also given you the ability to add in your own custom strings.

Just type into the box and press "Enter Custom" to add it to your hash generation.

Of course, you can add other characters to it.

For example...

You could see how many accounts have the password "password" followed by a number.

Or maybe you've noticed passwords tend to use the number 1 more often than other numbers...

So instead of inputting "Number" you may wish to input 1 as a custom string...

That would reduce the time to generate hashes by Ninety Percent!

Just some ideas

Whatever you do, DO NOT INPUT YOUR REAL PASSWORD INTO THE CUSTOM STRING BOX!

I know you're smart enough to not do that right?

Still, if you want to maximise the number of accounts you crack...

You should still primarily use the letter and number buttons.

I've given you an hour for this level.

So don't be afraid to make some time consuming hashes.

Great! See you later

D.2.4 Level 3

Good to see you again!

So you may have noticed that a lot of passwords containing letters aren't just random...

They are usually words, like the ones you'd find in the dictionary.

So to make your cracking more efficient, I've given you the ability to input dictionary words.

Just use the plus and minus buttons to adjust the word length...

and then press the middle button (labelled X letter word) to input it.

We call this a dictionary attack! I'm sure you'll find it very useful.

Oh one more thing!

This database contains passwords of size 8+ containing at least one number and one letter.

Try to use what you know about password patterns.

Think about where people like to put numbers in their passwords.

Good luck!

D.2.5 Level 4

Welcome back once again!

I see you've been using dictionary attacks quite a lot!

So I've given you the ability to customise your words even further.

Try pressing the "Word Options" button.

Under "Word Types" you can change what types of words to add to your dictionary attacks.

They're pretty self explanatory, just remember that they may not all be worth including.

"Common Substitutions" allows you to change the format of the words.

You can capitalise the first letter, capitalise the whole word...

Or use "Common L33T" which replaces some letters in the words with other symbols...

L!k3 th1$!

You'll also have noticed that you can use your previously found passwords as words too!

But keep in mind this is a different database, with different requirements.

The passwords we're trying to hash come from a website which required users to...

Make passwords of size 8+, containing at least one uppercase, lowercase, and number character.

So keep that in mind! I bet you'll find capitalising the first letter of words quite useful.

See ya!

D.2.6 Level 5

Hey!

You've been doing real great so far!

This is the hardest challenge yet!

These passwords come from a website with very strict password specifications.

Passwords must be of size 8+ containing at least 1 of each character type.

That is one lowercase, uppercase, number, and special character.

It won't be easy, but give it your all. Use everything you've learned so far!

D.2.7 Free Play

Congrats, that was the final challenge!

You have been amazing. That was a real stellar job.

Of course, not every account you cracked was useful.

Some users changed their passwords immediately upon learning about the database breach.

And some users were smart enough to not reuse their passwords on other websites.

Which we absolutely do check.

As a reward for all your hard work, here is a free-play level with no time limit and no constraints

That's all from me

Have fun!