Bachelor Degree Project

Multi-factor Authentication Mechanism Based on Browser Fingerprinting and Graphical HoneyTokens

Authors: Amin Marteni & Dillon Jonsson
Supervisor: Ola Flygt
Semester: HT 2022
Subject: Computer Science

Abstract

Multi-factor authentication (MFA) offers a wide range of methods and techniques available today. The security benefits of using MFA are almost indisputable, however, users are reluctant to adopt the technology. While many new MFA solutions are being proposed, there is a lack of consideration for user sentiment in the early stages of development. In an attempt to balance security and usability, this report investigates the feasibility of a new authentication mechanism that uses browser fingerprinting, graphical passwords, and honeytokens. This was evaluated by conducting a limited literature review, producing a prototype, interviews with test users, and security experts, as well as ensuring feasibility through a requirements checklist. The results of this research provides evidence that this mechanism is feasible, and appealing to end users. However, more investigation is required in order to ensure the mechanism's viability in a real-world deployment.

Keywords: Multi-factor Authentication, Browser Fingerprinting, Graphical Passwords, Honeytokens.

Contents

1 Introduction

Digital security has become more and more of a societal focus due to the rapid advancements of new technology and the threats that come with them. It is an unavoidable topic seen in modern media, with daily reports of attacks and data breaches from large companies. While there are many aspects related to security, one is user authentication over the internet. Remote authentication has proposed a constant challenge for IT professionals due to end users' lack of in-depth knowledge and failure to comply with security policies. This concept can be seen from a business and personal use perspective in relation to the adoption rates of (MFA) Multi-factor Authentication. MFA offers a wide range of methods and techniques available today. The security benefits of using MFA are almost indisputable. Despite this, people are reluctant to adopt the technology [1, 2, 3].

This is a bachelor's thesis for the Network Security program at Linnaeus University. The focus of this project is to investigate the feasibility of a new authentication mechanism, with a focus on user perspective. The proposed mechanism uses browser fingerprinting technology and graphical honeytokens. The mechanism evaluated usability by conducting interviews with test users and security experts together with ensuring feasibility through a requirements checklist.

1.1 Background

MFA is the process of combining different authentication factors to identify a system's user. These authentication factors are typically something one knows (text password), something one is (biometric ex: fingerprint), and something one has (Smart card, authentication token). The commonly known Two Factor Authentication (2FA) typically includes two factors from separate categories, while MFA can include several factors from the same category [4].

MFA schemes are typically used to authenticate users over the Internet. A client-server architecture is the basic building unit of web applications. A client tries to connect to a remote server to request the desired service from the web server. Web browsers are the dominant Internet client software used by the majority of users to interface with the internet. Web browsers provide web servers with access to different Application Programming Interfaces APIs and other system information via the HTTP protocol used by the client-server architecture to exchange information. Due to the increasing complexity of modern web applications, web browsers expose many vectors of information that can be used by web browsers to offer users the desired services and functionality. These vectors of information can be specific and they can help in distinguishing a certain system. The identification process of a device/browser via information vectors exposed by a web browser is called browser fingerprinting [5]. A browser fingerprint represents a compelling authentication factor, depending on the degree to which it is unique. There are two ways to compute a browser's fingerprint: The first, the fingerprint can be collected via publicly accessible information from the browser. HTTP headers, timezone, IP address, etc. The second method of collecting a browser's fingerprint is done actively by the web server. A web server sends crafted queries and monitors the bhaviour of the web browser when answering the queries. The difference in behaviour between browsers comes from the differences in software and hardware configurations and can be used in a browser fingerprint. A combination of information vectors can be used to create a more unique browser fingerprint [6]. Browser fingerprints are typically used for tracking users that why research in this area is focused on privacy and fingerprinting prevention

[7]. Browser fingerprints can also be used in security for users, and device authentication [6, 7]. The proposed MFA mechanism utilizes a fingerprinting mechanism to collect the browser's fingerprint to authenticate a user's device acting as the possession factor. Fingerprinting mechanisms should be considered carefully. A big challenge for browser fingerprinting is the stability of a given fingerprint because data used for fingerprinting is susceptible to change, thus, changing the entire fingerprint [6, 5].

Graphical passwords provide a different form for users to enter a knowledge factor during authentication. These passwords can be more memorable for users due to the concept that humans can remember images more accurately than text [8]. The most well-known graphical password implementation is drawing a pattern on a grid popularized by mobile phones. Various graphical passwords have been implemented to varying degrees of success [8]. However, they inherently also introduce more expansive vulnerabilities to some specific attacks such as shoulder surfing [8].

The word "Honey" in the cybersecurity field is used for techniques designed to attract or deceive an intruder. These techniques help detect if there has been a security incident and gives security teams a canary in a coal mine-like detection method to get one step ahead of an intruder [9]. This technique was initially used for systems like honeypots. Now the technique has been adopted for password storage and is called honeywords or honeytokens [4, 10] (see Section 3.4).

The combination of browser fingerprinting and graphical honeytokens in the proposed MFA scheme aims to overcome the challenge of users' negative attitudes towards MFA. Fingerprinting is passive and does not require user input. Graphical honeytokens are sent over the same channel in the browser, and they are easier to remember and input than other authentication factors.

1.2 Related work

The need for a convenient and secure way to authenticate users has caused an abundance of research. This need may be correlated to why a majority of these academic papers propose new MFA tools [1]. Since the proposed method relates to browser fingerprinting, graphical passwords, honeytokens, MFA, and user perspective, there is a large amount of related research.

Device and browser fingerprinting is a popular research subject. The use of fingerprinting mechanisms in authentication is being explored frequently, specifically in the area of IoT device authentication [11]. Alaca et al. [6] explored the different attributes that a device's fingerprint should have to be considered a viable option for authentication. The authors looked into how a device's fingerprint can be used for authentication, but they have not evaluated the usability and feasibility of their proposed methods. Threat models and their attack vectors were presented in this study to help in understanding the vulnerabilities of using a device's fingerprint for authentication [6]. In this study from 2021 [7], Durey et al. investigated browser fingerprinting adoption in web security. They found that browser fingerprinting research is aimed toward tracking, privacy, and bot detection. A Morellian Analysis of browsers is a new authentication protocol that uses Canvas fingerprinting to identify users to a web server [12]. This authentication protocol uses dynamic queries to the Canvas API to ensure the stability of the fingerprint. Laperdrix et al. [12] collected the Canvas fingerprints of more than one million devices; 99.9% of these fingerprints were unique. The authors did not evaluate the protocol's effectiveness in changing users' MFA adoption.

A study by Papaspirou et al. 2021 was crucial in developing this mechanism. It implemented a 2FA mechanism using honeytokens, sent through a One Time Password (OTP), as well as giving an overview of current MFA methods [4]. The honeytoken aspect of their proposed method used QR codes or URL links sent through email or SMS. This study was the most similar to this project as it also proposed a new mechanism that uses honeytokens. This concept was very similar to the proposed mechanism in this project.

Vaddeti 2020 presented a study that discussed the challenges faced by graphical passwords, namely balancing performance and security. This study was aimed at graphical passwords and related to this project solely because of the graphical aspect of the proposed mechanism. The study presented a method that keeps the benefits of graphical passwords without sacrificing either security or performance [13].

1.3 Problem formulation

Security is and always will be a societal focus. Cybersecurity is becoming increasingly so. The need for a convenient and secure way to authenticate users is a topic that is currently being researched. Users' attitude towards MFA is negative, which in turn decreases the adoption of MFA tools [2]. Ackerman [14], and Colnago et al. [15] showed in their studies that the lack of adoption of two-factor authentication methods is mainly due to: 1) it being time-consuming. 2) users were unaware of the dangers of cybercrime. 3) more than half of the users thought 2FA was a significant inconvenience and prevented them from doing important tasks. A survey done by Duo in 2017 showed that over half of the participants were unaware of what 2FA was [16]. With these studies in mind, MFA and 2FA are not depicted as favored in the eyes of the public. In a literature review by Das et al., a majority of academic papers proposed new MFA tools, but only around 9.1% performed any user evaluation [1]. This lack of user evaluation is a area of significance, because the adoption of MFA is inherently tied into the appeal of the solution to end users.

Browser fingerprinting and graphical honeytokens are decently researched. However, to the best of our knowledge, these two tools have not been used in conjunction to provide an MFA solution [12, 7, 4]. The knowledge gap for this research was tied both to the feasibility of this new mechanism and user evaluation. Bridging this knowledge gap was done by answering the following Research Questions (RQs).

RQ1 What is the state of the art of MFA solutions that use Browser Fingerprinting, Honeytokens, and Graphical Passwords? How does this solution build on existing solutions?

RQ2 What are current threats to MFA, and how do they impact this solution?

RQ3 What would be the necessary requirements for this solution?

RQ4 Would end-users be likely to adopt this mechanism?

1.4 Motivation

An authentication mechanism that is secure and easy to use is the desired product for many companies and industries to ensure accurate access control. This thesis provides insight into users' perspectives on a new MFA mechanism. The feasibility and security of this mechanism are also presented. This investigation advances the research for the scientific community and provides additional resources that assist in considering new MFA mechanisms. These findings could help improve MFA adoption rates and provide a more secure authentication mechanism. These results are desirable for ensuring security for society and industry.

1.5 Results

In this paper, a new MFA mechanism was produced. It investigated the feasibility of this solution for remote authentication. The implemented prototype was a resulting artifact. A Limited Literature Review (LLR) was conducted to assist in creating the prototype and a requirements checklist. Then the prototype was evaluated externally via open interviews with security experts with a focus on the security aspects of the mechanism. As well as closed interviews with end-users focused on the user evaluation. These results allowed drawing conclusions to the research questions.

1.6 Scope/Limitation

This report focuses on the proposed MFA mechanism that includes fingerprinting and graphical honeytokens; thus, other MFA mechanisms and their components were not evaluated. The users involved in the evaluation interviews mainly were university students, and a large percentage of them were computer science students. Another limitation of this thesis is the focus on remote user authentication; other forms such as local user authentication or device authentication were not included. Due to time and resource constraints, a *Limited* Literature Review was conducted.

1.7 Target group

Due to the recent attraction in research of fingerprinting, graphical passwords, and honeytokens, and the extensive list of challenges that come with remote authentication, this research may catch the eyes of a variety of people. However, because of the focus on feasibility and user evaluation, the target group for this thesis is software developers considering implementing new authentication mechanisms and security professionals considering furthering research in this area.

1.8 Outline

The paper is laid out as follows. Chapter 2 describes the methodology followed throughout this study to obtain, analyze, and evaluate results. Chapter 3 gives an in-depth description of the research area of MFA in general, with an emphasis on browser fingerprinting and graphical honeytokens. In Chapter 4, the prototype implementation is presented. Chapter 5 introduces the results obtained from the users' and security experts' interviews. Chapter 5 includes the security checklist conformance results. Chapter 6 is where the results were analyzed and discussed. Chapter 7 provides a full summary of the findings and discusses future research possibilities.

2 Method

This section describes the steps and processes conducted to fulfill the objectives of this research. A multi-method research approach was used to organize these processes. The section starts by describing the concept of design science and the motivation for its use (Section 2.1), then an in-depth description of each method with motivation (Section 2.2), reliability and validity (Section 2.3), and finally ethical considerations (Section 2.4).

2.1 Research Project

The research method for this project was based on the Design Science (DS) methodology. Design science is an area of multi-method research that is used when a project's focus is creating a new artifact, technique, or algorithm [17]. Although the main focus of this project was not on the development of the artifact itself and more on the feasibility of this new mechanism, the DS research method was the most applicable for this project. In a paper published in the Journal of Management Information Systems in 2007 the author Peffers described and evaluated the DS research methodology. DS methodology is divided into 5 Activities [17].

1. *Problem Identification and Motivation:* Defining the problem and justifying the solution.
2. *Define the Objectives for a Solution:* Defining objectives that can be quantitative in order to compare to existing solutions or qualitative in how it is expected to perform as a solution. These should be rationally connected to the problem at hand.
3. *Design and Development:* Create the artifact itself.
4. *Demonstration:* Do experiments in order to gain knowledge on the effectiveness of this solution.
5. *Evaluation:* Decide if this solution solves the problem. This activity is the culmination of the previous steps and determines the feasibility. That could improve the artifact itself or describe the research to relate to future investigation.

After reading these activities, it is likely clearer why this is the multi-method research approach related to this project. The DS method provides the general structure of the research and provides the path to reach the final goal of determining the feasibility of the proposed solution. At this point in the report, activity 1 (Problem Identification and Motivation) was already completed in the Section 1.3. The next step is to Define the Objectives for a Solution.

2.2 Research Methods

This section goes through each of the methods in more detail, describing how they were done and the logic for their use. Figure 2.1 shows a diagram of the methods used in this study and what Research Questions RQs they answered.

2.2.1 Limited Literature Review

Systematic Literature reviews are generally done in order to; Summarize the existing state of an area, identify research gaps, or get a background for new research [18]. The reason literature reviews are done so commonly in research is that they assist in combating several forms of bias. Mainly selection bias, performance bias, measurement bias, and attrition bias [18]. With this in mind, systematic literature reviews are resource-consuming, so a limited literature review was chosen. Other projects are exclusively systematic literature reviews, while the LLR method is only one of the methods for this project. Other than being less resource-consuming, the LLR method was selected for several reasons. It provided a background on the research areas and the current state of MFA systems, assisted in developing the requirements checklist for the mechanism, and raised possible attack vectors for similar implementations.

The methodology for doing the limited literature review was first to find existing systematic literature reviews on relevant research areas and snowball from there. Relevant search terms such as *Browser Fingerprinting, Honeytoken, and Graphical Password* were used to find the systematic literature reviews across several databases. Primarily Google Scholar, IEEE Xplore, and ACM. In order to obtain more in-depth research and existing summaries of topics on related topics. (See appendix for the exact search terms used).

The proposed mechanism had never been implemented, and the combination of authentication factors had never been used in conjunction. However, there is existing research on the use of the factors separately. In this way, research on existing implementations was used in designing and implementing this mechanism.

2.2.2 Requirements Checklist

In order to assist in ascertaining if the solution was sufficiently viable, a requirements checklist was created. This checklist was made by drawing elements from research found in the LLR essential to other authentication mechanisms and products. Additional requirements may be added after interviews with security experts. This method served to improve the internal validity of the feasibility of the mechanism. The requirements checklist was created to be used as a tool to evaluate the solution. In this way, the checklist allowed making the claim that the mechanism was or was not feasible with these features. This method also helped answer RQ 3. Further information on the implementation and the evaluation stage from the requirement's checklist can be found in section 5.1.3.

2.2.3 Development

The development of the prototype was a continually iterative process that was only limited by the allocated resources. The most limited being time. Once the prototype had been implemented to a sufficient stage, it was used for interviews with end-users. The prototype served as a proof of concept that helped legitimize the feasibility of the mechanism. Using the prototype for end-user testing helped answer RQ4 and fulfill activities 3, 4, and 5 from the DS methodology.

The purpose of developing the prototype in an iterative process was to have the opportunity to adapt the prototype to newly discovered information. The interviews with the security experts were done after end-user interviews, if there was time their input was used to improve the prototype. If there was insufficient time, these changes will be mentioned in the results Section 5 and future work Section 7.1. The development process is described in more detail in Section 4.1.

2.2.4 Interviews with End Users

The interviews with end-users were an essential aspect of evaluating this solution. The interviews were a combination of open and closed style interview questions. A closed interview is guided with prepared questions that have been carefully crafted to obtain as clear data as possible. Since closed interviews may limit the participants, an open-style interview took place at the end of the closed interviews. This mix of open and closed style interviews with end-users improved this thesis's external validity and reliability.

The interviews were recorded to avoid recall bias. End-users first signed a consent form to allow voice recordings. The interviews then had the participants test the prototype developed. Then, participants answered curated questions that were created to test their satisfaction with the experience of the prototype. Finally, the participants got a chance to further discuss MFA and other aspects of the mechanism in the open part of the interview. The interviews with end-users were five questions in total, with sub-questions aimed at clarification, quantification, or the opportunity to expand. This method was aimed specifically at RQ4. The interview protocol, consent forms, and interview results can be found in Appendixes A, B, and 5.3. Relevant ethical considerations and evaluations can be found in Section 2.4.

2.2.5 Interviews with Security Experts

Essential parts of DS activities shown in section 2.1 are demonstration and evaluation. Interviews with security experts were conducted to demonstrate a working prototype of the proposed authentication mechanism. The iterative process used in developing the mechanism and implementing a functional prototype was partly guided by security experts' input. The outcome of these interviews was also considered when creating the requirements checklist discussed in section 2.2.2. Thus, the interviews were directly connected to the evaluation of this project. These interviews were conducted after the end-user interviews, as this would be when the experts' opinions were the most valuable. Security experts also have more credibility when determining the feasibility MFA systems. It was also possible that they would have an opinion in relation to the adoption and sentiment of MFA for end-users.

The goal of this method was to collect as much input from the experts as possible. Due to this concept, an open-style interview was chosen where the interview was more of a discussion. Open interviews helped obtain qualitative results from the experts who were not limited by the questions asked. In order to avoid stagnation of time, guiding questions aimed at creating an effective use for the interview were also created. The interviews were recorded to avoid recall bias and improve the reliability of the results. The same ethical considerations used for end-users were also applicable for this method.

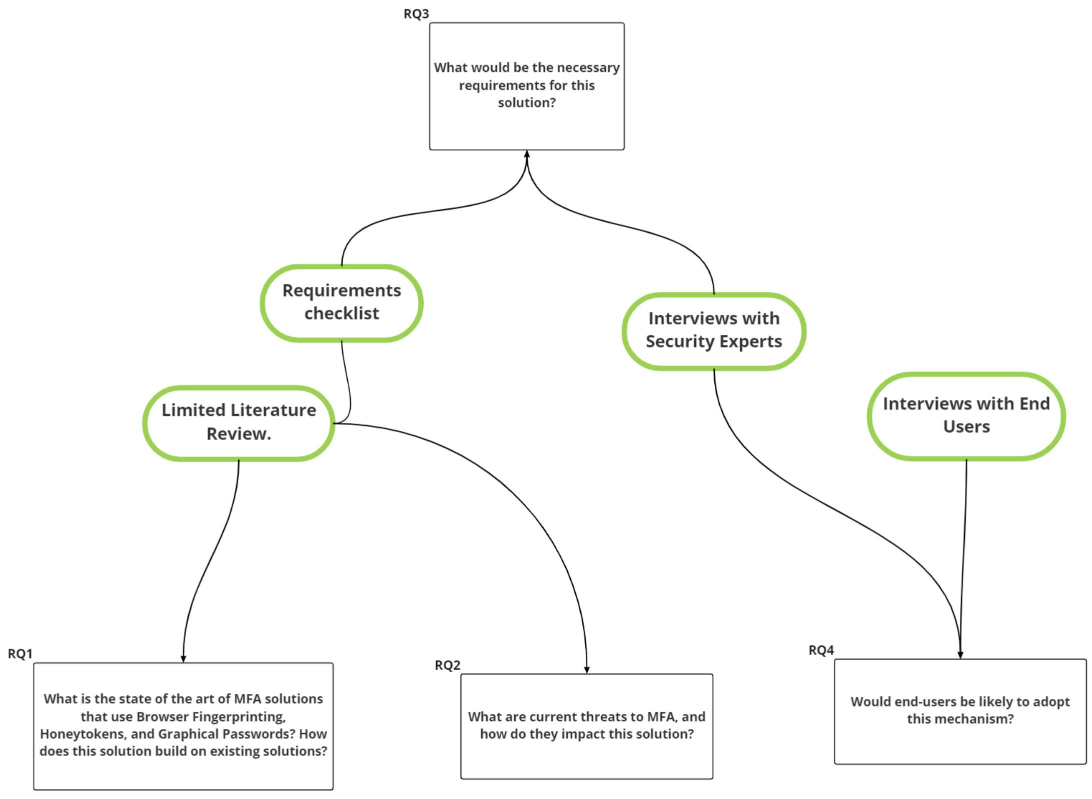

RQ3
What would be the necessary requirements for this solution?
Requirements checklist
Interviews with Security Experts
Limited Literature Review.
Interviews with End Users
RQ1
What is the state of the art of MFA solutions that use Browser Fingerprinting, Honeytokens, and Graphical Passwords? How does this solution build on existing solutions?
RQ2
What are current threats to MFA, and how do they impact this solution?
RQ4
Would end-users be likely to adopt this mechanism?

Figure 2.1: Methods and RQs

In figure 2.1 the Limited Literature Review was used to answer RQ1 and RQ2 directly. The requirements checklist is an outcome of the Limited Literature Review, which in turn answered RQ3 with the help of Interviews with Security Experts. As for RQ4, it was answered using the Interviews with End Users and Interviews with Security Experts.

2.3 Reliability and Validity

In order to retain reliability concerning this research, the procedure was documented, and standard methodologies were used in similar research. Using the DS methodology would help if someone were to attempt to recreate the prototype from scratch. There may be some inherent possibilities of different tools and ways of implementation that could cause variability in replication. However, the systems would be similar due to the documentation of all methods followed. The code is accessible through Github and could be used to replicate specific methods to a high degree of accuracy (See section 4). Different methods were selected to increase validity in different ways. Both types of interviews contributed to the external validity. The interviews with end-users were used to validate the possibility of users' adoption. In contrast, the interviews with security experts were used to validate the mechanism from a security and feasibility standpoint.

The requirements checklist was used to increase the internal validity of this research. This assisted in providing a quantitative measurement to evaluate the prototype and mechanism. Due to this checklist being built from existing similar research, the checklist also increased the reliability of the claims related to RQ3.

2.4 Ethical Considerations

This research involved interacting with participants via interviews, making ethical considerations relevant. These interviews were anonymized, and the interviews did not publish any personally identifiable information. There was no risk of harm during the interviews. Thus, ethical considerations in comparison with other possible methodologies were limited.

In accordance with the Ethical Advisory Board in southeast Sweden, An Ethical self-review form was done to account for ethical considerations. In the review and from a conversation with the project's supervisor, it was found not necessary to apply for external ethical validation. In summary, all interviews were done entirely voluntarily; participants signed a consent form before the interviews and had the ability to withdraw at any time during or after the interviews.

Sampling for end-users interviews was done purely on the availability of participants. Due to resource limitations, the participants were mainly other computer science students working on their thesis projects. Due to these limitations, there may be sampling bias; however, it was mitigated by attempting to gather students from alternate domains. The sample size is also small due to limited resources.

Testing the prototype was done on the authors' machines, which helped preserve the testers' privacy. The only personally identifiable information that was collected during the interviews was the participants' voices. The only persons with access to this data were the researchers. However, the interviews were transcribed, and the recordings were deleted after the interviews. This was done in order to ensure confidentiality. For a more detailed account of ethical considerations, see the self-review in Appendix D.

A final consideration is the ethical benefits possible with a new secure authentication mechanism. This is because privacy is intertwined with security and access control. A more usable authentication mechanism could increase the adoption of MFA, and, therefore, decrease the number of people impacted by data loss.

3 Theoretical Background

This section aims to provide background concerning the relevant area of investigation for this thesis. The following is primarily a result of the LLR.

3.1 MFA

Authentication is proving that a person is whom they claim to be [19]. This is distinctly different from identification, being "the act of asserting who a person is" [19]. These two concepts are often mistaken.

The way that authentication is done is by a person providing a factor. This is common to access a system or resource. There are authentication mechanisms that use only one factor. Many will recognize the most common factor for single authentication mechanisms being a password [20]. It is not uncommon for access to resources to be placed behind this simple authentication mechanism. However, this single point of failure has known vulnerabilities [20]. The vulnerabilities are becoming more complex as the types of attacks do as well [2]. Among the many challenges that exist with text-based passwords, some of the most common are related to the memorization of quality passwords. One study found that 80% of passwords were able to be cracked by the average personal computer in less than a week [21]. This concept is relevant as computing power increases over time and is due to low complexity passwords and a lack of understanding from end-users [20, 2]. As participants of the same study appeared "to be unconcerned about the risks associated with poor password composition" [21]. Moreover, 40% of users had never changed their passwords. This has compounded in recent years due to the increase in the number of services that require passwords, reusing passwords, and using passwords with guessable openly known content such as a spouse's name [19, 21, 2].

This concept contributed to the development of adding more factors for authentication. First, there was Two Factor Authentication, then Multi-Factor Authentication to ensure a more secure authentication of a user. Authenticating users has a deep involvement in the field of cyber security because impersonation can pass through authentication mechanisms and typically gives access to protected resources. In general, this is the foundation of cyber security, and often the first line of defense [22].

Authentication mechanisms are typically classified by how many factors they use and the categories of the factors. The types of factors can be categorized into one of the following groups. Something one knows (Knowledge factor), something one has (Possession factor), and something one is (inherence or biological factor) [22]. Examples of knowledge factors would be things like PIN numbers, text passwords, lock patterns, graphical passwords, challenge-response, etc. [4, 23]. Examples of possession factors would be: ID cards, NFC, RFID, smart cards, hardware tokens, physical keys, or a device itself [4, 23]. Examples of biological factors would be: facial recognition, iris, fingerprints, or even other physical characteristics such as typing speed and speech recognition [4, 23].

This is the most well-known way of categorizing authentication mechanisms [22]. Categorization also assists in defining whether a mechanism is 2FA or MFA. Commonly a 2FA mechanism has two different factors from each of the three groups, and MFA has more than two and can have several from the same group [4]. It is not uncommon for other factors to exist and not fit into the "textbook" definition of these categories, such as location-based authentication [24]. This is not technically a knowledge, possession, or biological factor.

2FA and MFA, however, have not solved the challenges faced with authentication. End-users have negative connotations with MFA for a variety of reasons [14, 15]. Adoption rates for MFA concerning websites have stagnated. A 2018 study showed that 2FA over a growing set of websites was close to 50% for over four years from 2014 to 2018 [25]. Both the end-users and services providing authentication options have impacted the adoption of MFA.

MFA systems themselves are still vulnerable to attacks. Although the types of attacks are numerous, familiar and relevant examples are: Leaks from a server, phishing, theft, man-in-the-middle, social engineering, shoulder surfing, key loggers, spyware, and other types of malware/Trojans [23, 26]. Despite this, it is still a safe assumption that MFA and 2FA are more secure than single-factor systems. This is because the more factors that correlate to a user, the more confidently one can assure the identity of a person [19]. Typically the more factors that there are in a system, the more actions are needed by end-users, and therefore increased inconvenience [19]. This is an important concept in relation to this study. Increasing the confidence in authenticating a user while avoiding inconvenience.

3.2 Browser Fingerprinting

The main concept of browser fingerprinting was introduced in section 1.1. This section is dedicated to giving the reader a more in-depth view of browser fingerprinting technology. One of the earliest studies concerning the uniqueness of a web browser was conducted by Peter Eckersley back in 2010 [5]. In his experiment, the author showed that they were able to uniquely identify a visitor of their website with 99.1% of certainty. Eckersley used publicly available and easy to access information to fingerprint a visitor's web browser; amongst this information, the Hypertext Transfer Protocol HTTP header user agent was collected; alongside fonts, screen resolution timezone, etc. The information/vectors that made up a browser fingerprint could be obtained actively or passively. Fingerprinting vectors sent by a visitor's browser with an HTTP request, such as specific HTTP headers, are an example of passive fingerprinting. An example of active fingerprinting is vectors collected through a JavaScript JS file running in the browser collecting information via available browser APIs and then sending vectors back to Eckersley's web server. In this paper, Eckersley was researching browser fingerprinting as a method to identify and collect unique information about users of the internet. Eckersley was warning about this potentially being a considerable user privacy risk and could lead to detrimental effects if not dealt with properly. The author also gives recommendations on how to defend against browser fingerprinting [5].

Eckersley's paper at Electronic Frontier Foundation EFF [5] was eye-opening to the security community and led to research in the browser fingerprinting area. One notable paper was published in 2012 and written by Mowery et al. introduced a new browser fingerprinting technique called Canvas fingerprinting [27]. The Canvas element is an HTML5 element that utilizes different APIs like the Canvas API and the WebGL API to draw 2D, or even 3D graphics [28]. This means that a web server could access the Canvas element and draw an animation or some graphic using JavaScript. The browser will grant access to the underlying OS and hardware like the Graphic Processing Unit GPU to render the remote server's request for a given graphic. Having access to the OS and underlying hardware via the browser's APIs gave Mowery et al. the inspiration to try and fingerprint a browser using such access. In their paper, the authors describe the steps necessary to obtain a Canvas fingerprint; a JS script running the client's machine would ask the Canvas API to render a text or other graphics like the WebGL scenes. Then, the fingerprinting algorithm would analyze the results pixel by pixel to give back a hash of the results. This hash is the fingerprint of that web browser. Due to the variation in the OS and hardware involved in rendering the requested graphics, the results are often unique to that browser/device. Canvas fingerprints are easy to obtain, highly consistent, and have a high entropy which makes them perfect for identification [27]. Mowery et al. focused in their paper on security and privacy concerns that may arise from their proposed technique. Canvas fingerprinting is an excellent choice for browser identification since it relies on the hardware used in a system. However, more people having the same hardware could potentially lead to misidentification, which is not the desired behavior in an authentication system. Mowery et al. did not have enough data to calculate appropriate entropy bits to determine the uniqueness of a Canvas fingerprint. However, Laperdrix et al. [29] have conducted a large-scale study to measure the entropy of a Canvas fingerprint and found that Canvas fingerprinting is one of the most uniquely identifying fingerprinting techniques out on the internet.

Many other pieces of information that are available via the web browser can be used in fingerprinting. Browser extension, AudioContext, various HTTP headers, browser plugins, IP address, etc [6, 30]. A fingerprint is desirable when it is consistent, unique, and stable [6]. The three attributes are hard to find in one fingerprinting technique. Often, there is a trade-off between stability and uniqueness. More fingerprinting vectors must be taken from the browser to obtain a unique browser fingerprint. The obtained unique fingerprint is now as stable as the least stable element. For example, a Canvas fingerprint is highly stable, but when coupled with IP address, it becomes less stable because IP addresses are more likely to change, thus, changing the fingerprint with them. When implementing a fingerprinting algorithm, these factors must be taken into account.

Browser fingerprinting is a common tracking mechanism used on the internet. Research in the area of browser fingerprinting is often directed toward tracking and privacy, as shown in [31, 29, 5]. Al-Hannah et al. found that 68.8% of the top 10,000 websites are using some fingerprinting method to identify and track their visitors [32]. Other studies showed similar percentages of websites using fingerprinting for user tracking. This all depends on the database of websites analyzed, the definition of fingerprinting, and how fingerprinting methods are identified. There are many tools and techniques described in the literature to defend against browser fingerprinting when used for tracking [30], but this is outside the scope of this paper. Durey et al. wrote in their paper published in 2021 [7] that the adoption rate for browser fingerprinting for security is rising. This study is one of few studies that was found when conducting the limited literature review that focused on using browser fingerprinting for the sake of security. The authors of this study visited 1485 pages and analyzed tracking scripts used on sign-up, sign-in, basket, and payment processing. They found that 405 pages have collected their browser's fingerprint. Durey et al. have found that only one of the inspected pages used browser fingerprinting for authentication upon signing up. They have also found that only 5 out of 42 used the collected fingerprints to inform them that someone was trying to log into their accounts from different devices. The authors have also concluded that the current state of the art does not provide sufficient information about the use of browser fingerprinting in MFA solution [7]. Durey et al. have found a need to study users' attitude toward MFA schemes that includes browser fingerprinting. Two attacks were proposed by Durey et al. to evaluate if the fingerprints collected from the websites in their experiments are being used for security. These two attacks are stolen credentials and cookies hijacking. In both attacks, it was demonstrated that the collected fingerprints are not being used to defend against the attack models [7] which makes it most likely that the fingerprinting is only being used for tracking.

A pattern can be seen from this section introducing fingerprinting, which is that browser fingerprinting is almost always used and researched in light of privacy and tracking and that more work is needed to adopt fingerprinting in security, especially in different MFA schemes that real-world users evaluate.

A web browser is complex and highly interconnected with other parts of a system, which gives the remote web servers many entry points for data collection to fingerprint and identify a device. Using browser fingerprinting for authentication is not a new concept and has been studied in some studies [33, 6]. However, these studies are always lacking by not designing a complete authentication system or only focusing on some theoretical parts and not considering users of the system. For the authentication solution proposed in this report, Canvas fingerprinting is the main browser fingerprinting method that was studied and utilized in the prototype described in detail later in section 4.

3.3 Graphical Passwords

Graphical passwords are passwords that have some form of a graphical element. There are three different categories of graphical passwords that have a large effect on use, memorability, and attack vectors. The three different kinds of graphical passwords are recognition-based, pure recall-based, and cued-recall [13].

Recognition-based graphical password systems generally present the user with several images and ask the user to select the ones chosen in the registration phase [34]. These images can be from a set of images provided by the system or uploaded by the user, like in the paper Creating Graphical Passwords on a Mobile Phone [34]. The graphical password factor for this research uses the former. In the latter, it is common for the images to be obfuscated in some way [8]. One of the earliest graphical password systems implemented, known as "Déjàvu" [35] used system given images. The main drawbacks of recognition-based graphical passwords are possible vulnerabilities to shoulder surfing, phishing attacks, and a time-consuming selection process [34, 23].

Recall-based graphical password systems typically have a user draw their password. This system is similar to that of checking signatures for electronic documents [34]. Challenges with pure recall-based graphical passwords depend on the device they are implemented on. On devices that do not have an accurate and familiar utensil, such as a stylus, the drawing can be challenging to replicate [13]. However, this results in these systems having better mitigation against brute force attacks such as dictionary attacks, as this is more challenging to automate [34].

Cued-recall graphical password systems often present the user with one image. Then the user selects one or more locations on the image as their password. This makes it so that the image used is important, and the password itself because users are likely to use "hotspots" that would be easy to guess [13].

3.4 HoneyTokens

HoneyTokens gets its name from the more commonly known Honeypots. Honeypots are machines or virtual machines that are used to entice attackers into a part of the network that is not connected to the actual network [19]. This way, admins can monitor and learn from the attack, then identify vulnerabilities [19]. This concept has been extended to other pseudo entities to attract and identify attackers such as Honeyaccounts, HoneyLambdas, and Honeyentries [36, 37]. These techniques function in the same way as a Honeypot. If an account, URL, or database entree is used, it can likely be because of unauthorized access.

These terms occasionally overlap. The suit words Honeyentry, sweetword, and Honeyword are all similar in that they relate to the database. Both terms sweetword and Honeyword can be used almost synonymously as they are both pseudo passwords stored in the back-end. Honeyentries are different because they can be used to store other items. The term honeytoken also obfuscates this. Honeytokens are originally defined as entities that are not honeypots but serve the same function [36].

The logic for the implementation of honeytokens works solely in the back-end. In layman's terms, there are two primary components. A Login Server (LS) and a separate server known as the Honeychecker (HC) [4]. When a user attempts to log in, the credentials are sent to the LS and checked against the entries for that user. In that entry, there are several hashed passwords correlating to that user; however, only one is the "true" password for that user. The others are the honey/sweetwords. This is where the HC comes into play. The HC contains the index of the correct password for that user. The LS then sends what (if any) index the entered password matches to [4]. If this matches another index but not the correct index, the system should alert administrators and lock the account for that user [4]. This is best functional when there are significantly more honeywords in the LS than real ones in order to avoid a successful authentication on breach [36]. The purpose of having the HC and LS separated is that if they were both stored in the same place, the attacker could see the correct hash to avoid alerts and effectively impersonate a user.

4 Prototype Implementation

This section will describe how the prototype was implemented and then describe the state of the implementation. This step in relation to the DS research method would be "Activity 3. Design and development" [17].

Any authentication mechanism needs a secure channel in order to share encrypted data confidentially [12]. An important assumption for this mechanism and other MFA solutions over the Internet is the security of HTTPS. The assumption that HTTPS is secure holds for most websites on the Internet today.

4.1 Development Process

In order to gain as much knowledge as possible from this research the methods were done in parallel, and the implementation of the prototype was done in an iterative fashion. This process is displayed in Figure 4.1.

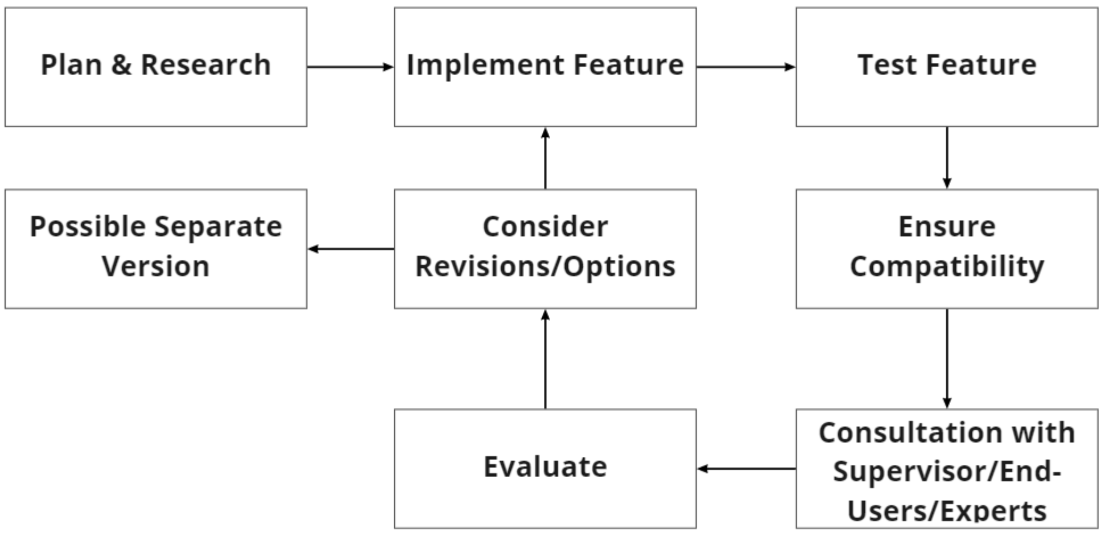

Plan & Research
Implement Feature
Test Feature
Possible Separate Version
Consider Revisions/Options
Ensure Compatibility
Evaluate
Consultation with Supervisor/End-Users/Experts

Figure 4.1: Diagram of Iterative Development

First, there was planning and research. This step involved researching authentication mechanisms with one of the authentication factors and determining its effectiveness through a prototype. For example, the primary jumping point for this project, A novel Two-Factor HoneyToken Authentication Mechanism [4], was used as a starting point for implementing the honeytokens. This step also resulted in research relating to programming languages, database theory, design architecture, and browsers due to a lack of experience.

Then a feature was implemented and tested for functionality. Due to the concept that this implementation is iterative, the features were also tested in conjunction. This was done to ensure that there were no conflicting components that impeded another feature use.

Due to the parallel timing of implementation, and consultations/interviews, the person(s) who provided input changed throughout the development of the prototype. This was done due to the availability of informants and when a person's experience was the most valuable. Through the prototype's initial development, the supervisor was used as a valuable source of information. Once a functional prototype was developed to the point that the user experience would be mostly similar regardless of a change in the implementation, end-users were interviewed for input. This also assisted in answering RQ4, see in Section 1.3. Finally, near the end of the research, experts were used as a source of information and input. This was also relevant to RQ2-3 and important for evaluating the feasibility of this mechanism.

After a feature was consulted with an informant, the feature was evaluated and considered for use next iteration. The whole process was then repeated for as long as resources were available. When there were several options for how the prototype could be implemented, but neither option was more effective, separate versions of the prototype were implemented.

4.2 Resulting Prototype

The resulting prototype was implemented in JavaScript, HTML, and CSS. Node.js was the JavaScript run time chosen alongside the Node Packet Manager NPM, with the external libraries express.js [38], Nodemailer [39], fingerprintJS [40], and bcrypt [41]. The source code for the prototype can be found on the GitHub repository following the link: https://github.com/Amin195/Own-version-of-BSC-degree-project. Figure 4.2 shows the authentication scheme, however as will be discussed later may have some variations with the different versions of the prototype.

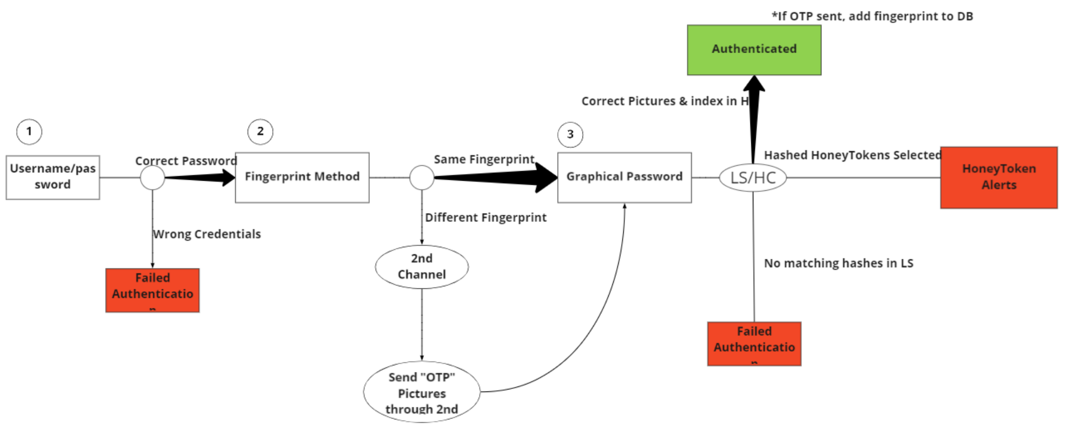


Figure 4.2: Diagram of Authentication Scheme

In the resulting prototype, the authentication scheme functions through three steps. The first is shown in figure 4.3. The username/password method where the credentials are checked against the database. If the credentials do not match, a message is displayed for incorrect credentials, and there is a failed authentication. The implementation moves to the following method if the credentials entered match the username and hashed password in the database.

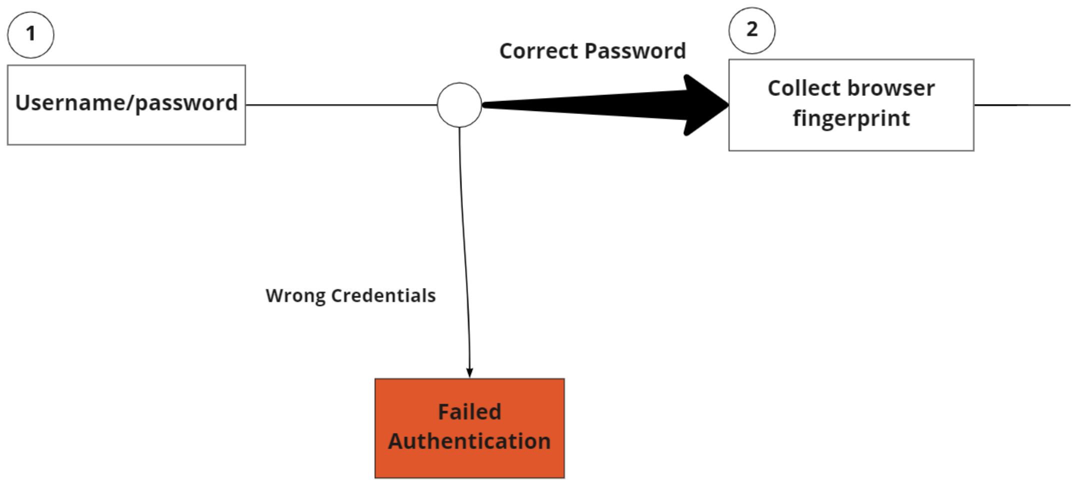


Figure 4.3: First Authentication Step

The second method shown in figure 4.4 is the browser fingerprint method, where several parameters are queried from the users' browser. The client's browser is asked to render some graphics on an HTML5 Canvas element to obtain a Canvas fingerprint of that device to identify them. If the fingerprints match the ones stored in the database that belongs to that user, the user will move on to the final method.

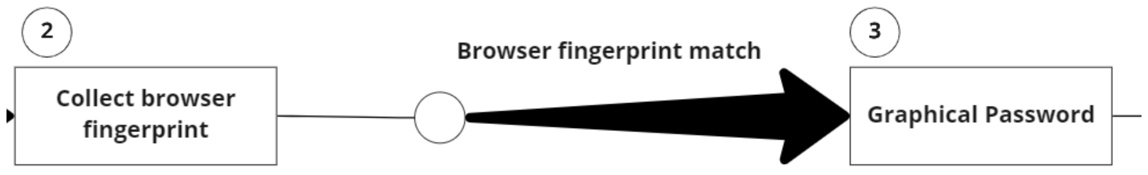
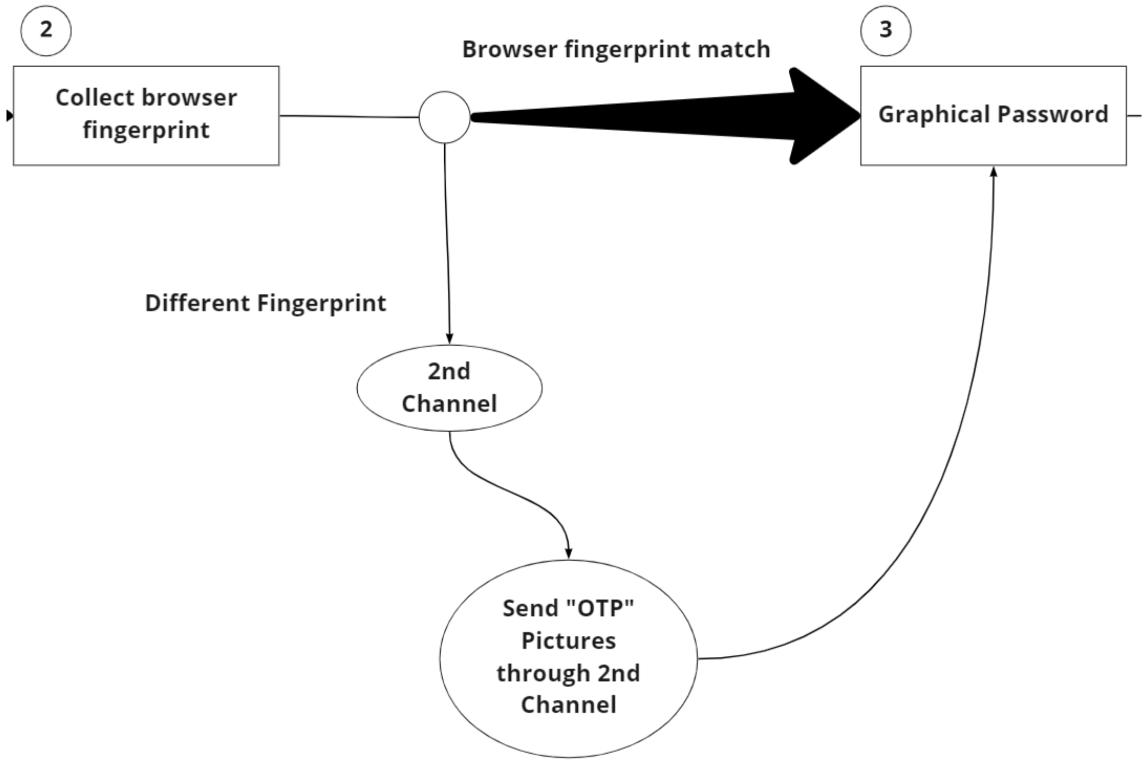


Figure 4.4: Second Authentication Step

However, if there is a fingerprint mismatch between the submitted browser fingerprint and fingerprints on the database, this next step will be slightly altered. This would be the case, for example, if the user were attempting to log in on a new device. That scenario is shown in figure 4.5 an OTP describing a different GP to enter in the following step is sent to the user's email. The user should then enter the OTP GP and not their usual GP.



2
3
Browser fingerprint match
Collect browser fingerprint
Graphical Password
Different Fingerprint
2nd Channel
Send "OTP" Pictures through 2nd Channel

Figure 4.5: New Device/Changed Browser Fingerprint Authentication Step

The final method is the graphical password method shown in figure 4.6. If the user's fingerprint matched, the user will enter their usual graphical password. This will then be checked against the LS for matching hashes in the database. If it does not match any of the hashes stored for that user, the GP is incorrect, and there is a failed authentication. If the entered GP matches a sweetword in the LS, the login server checks the Honey Checker to see if the entered GP is at the correct index. If it is, the user is authenticated. If an OTP was sent to the user for this session, the OTP is checked instead, and if this matches the new fingerprint taken in the second step will be stored for this user. If no OTP was sent, and the GP matches a sweetword a the incorrect index, a honeytoken alert is sent to the admin for the system.

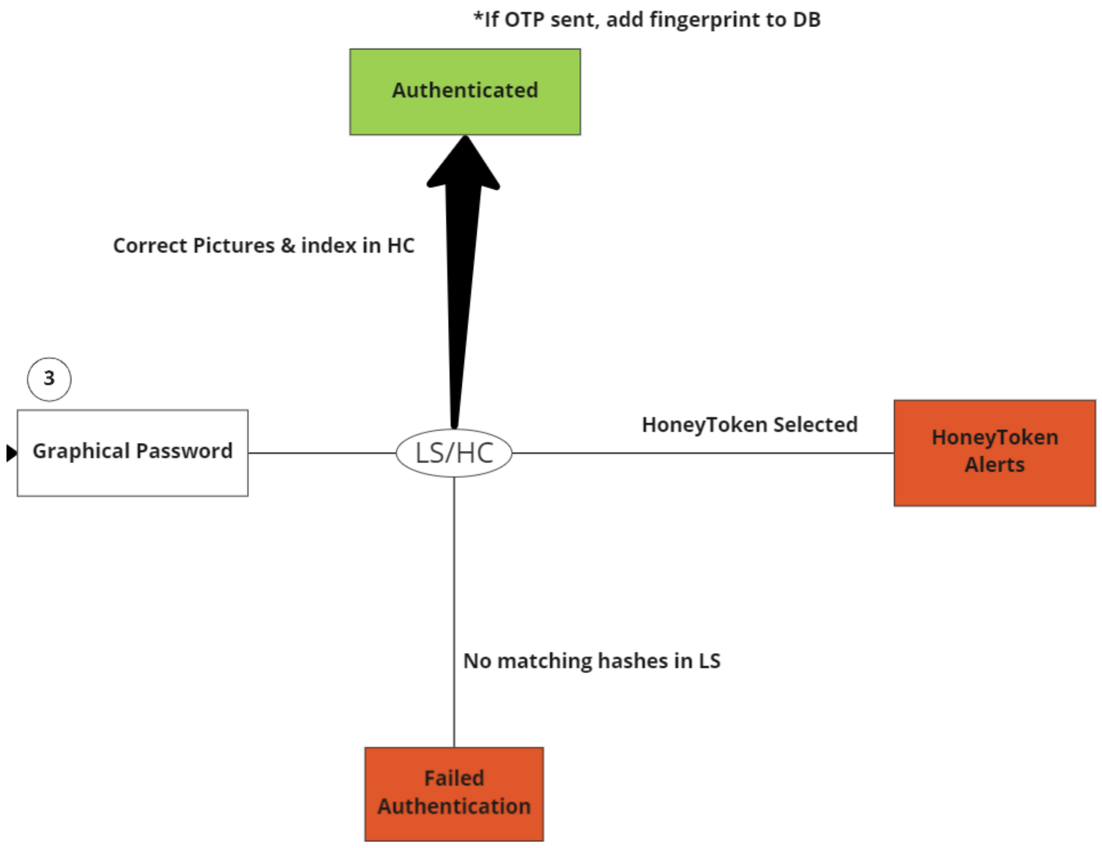


Figure 4.6: Third authentication step

Each of the methods are described in more detail in the following sections 4.3, 4.4, and 4.5. The scheme shown in figure 4.2 was used in the implementation of the prototype. This was done with the express.js library to create a web server that hosts the HTML, CSS, JS, and other file types that are served to the client. Express was also used to accept and respond to users' HTTP queries. This functionality can be found in the server.js module on the GitHub repository.

4.3 Fingerprinting

When a user of this prototype navigates to the login page, they are asked to provide a username and a password. The web server sends a fingerprinting script based on the fingerprintJS library that will run automatically upon visiting the login page, giving the user a seamless and high-quality experience. When the user clicks on the login button, the username, password, and browser fingerprint are all sent to the server simultaneously, thus, saving the user time from having to send and receive multiple requests to the server to authenticate. Canvas fingerprinting is the browser fingerprinting technique that was chosen for this implementation. FingerprintJS was chosen as one of the libraries that implement reliable and straightforward Canvas fingerprinting. Another variation was implemented based on a study by Laperdrix et al. [12] where unique Canvas queries were issued to a client, thus, preventing simple replay attacks when Canvas is used for authentication. More on the different fingerprinting variations in Section 6.

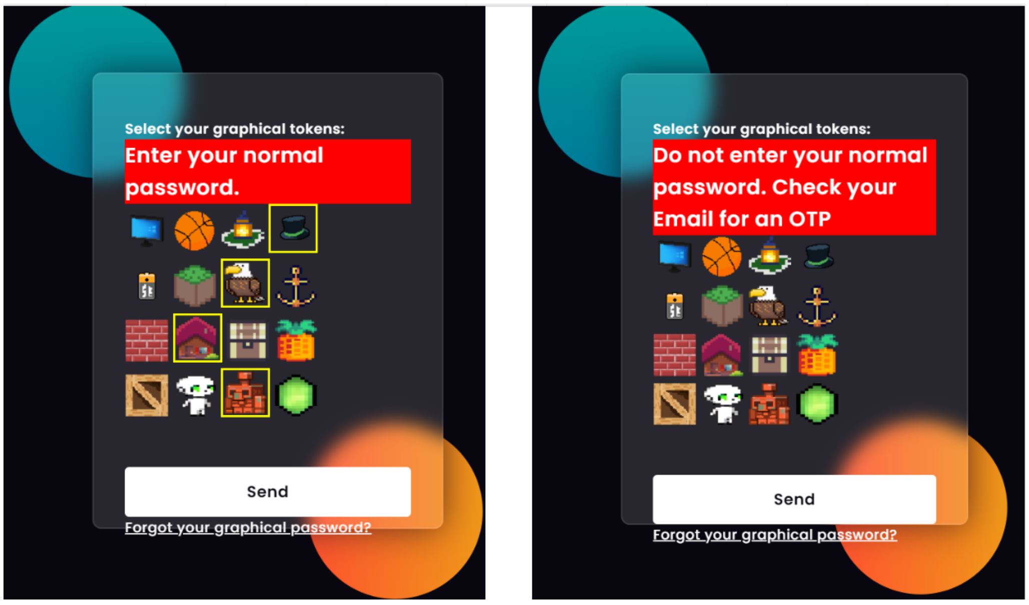
Once the server checks that the username and password match the ones stored in the database, the fingerprint is checked against the database. If the user's fingerprint has changed, the server utilizes the nodemailer library to send an OTP to the user's email address. The server also renders a new page for the user to enter their OTP. Once the correct OTP is entered and checked, the user is granted access, and the new fingerprint is added to the database.

To avoid having multiple old fingerprints that are no longer in use, the server runs a check periodically and deletes browser fingerprints that are no longer in use. For example, if a fingerprint has not been used in a month, the server would delete it from the database, thus, providing more security and preserving space.

4.4 Graphical Passwords

The graphical passwords implementation for this prototype used a recognition-based implementation. Upon reaching this stage, the users are presented with a 4x4 grid of simplistic pixel art. The images are from www.pixilart.com using their pre-made sticker tool.

At this stage, the user will be served the graphical.ejs file. This file contains the internal JS script for the functionality of selecting the images and posting them to the server. The specific data sent to the server is simply the hashed IDs for the selected images.


Select your graphical tokens:
Enter your normal password.

Select your graphical tokens:
Do not enter your normal password. Check your Email for an OTP

Send
Forgot your graphical password?
Send
Forgot your graphical password?

Figure 4.7: GP stage (Normal Password on left, and OTP message on right)

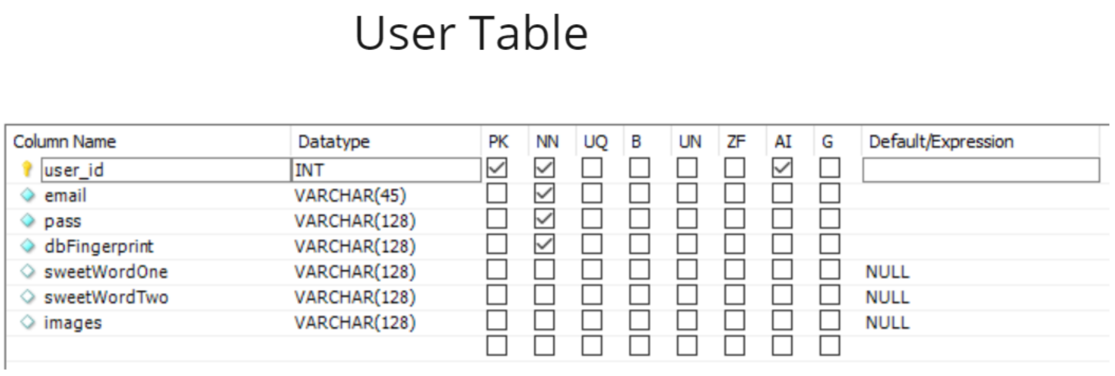
The users can then select the prompted images, and when they do, the image will have a yellow border showing what they have selected. The users are also able to click the image again to un-select accidents.

The graphical passwords were the location where there was the most variance in the different possibilities for implementation. However, the most notable difference would be if this password is a combination (order not mattering) or permutation (order mattering). This concept will be discussed more in the results section of the report.

Regardless of the specific version of the implementation, once the data is sent to the server. The post is handled in the password-login.js file, specifically in the graphicalAuth function. This function can change the order of the posted data if necessary, handle the change in logic if an OTP was sent, and query the HC to validate if the index of the graphical password is correct.

4.5 HoneyTokens

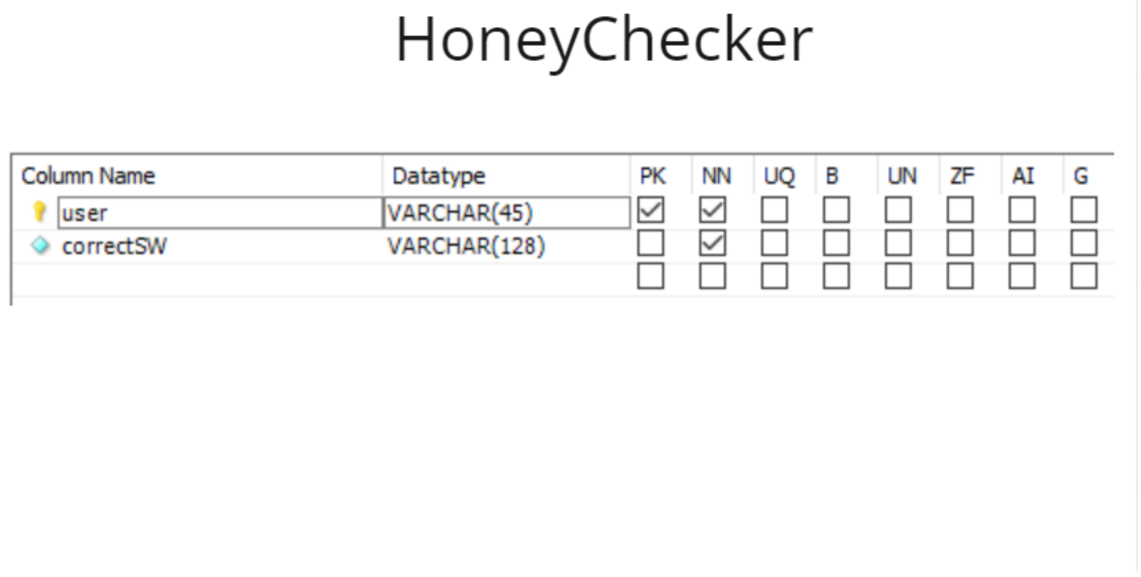
The back-end database for this prototype was implemented in a MySQL server. Due to the limitation of resources for this research, the Login Server and HoneyChecker have been emulated simply by having two separate tables. Although in a more actual implementation, this would be two separate servers. The first server would be the Login Server, which contains all the User Table information shown in figure 4.8.


User Table

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
user_id	INT	✓	✓					✓		
email	VARCHAR(45)		✓							
pass	VARCHAR(128)		✓							
dbFingerprint	VARCHAR(128)		✓							
sweetWordOne	VARCHAR(128)									NULL
sweetWordTwo	VARCHAR(128)									NULL
images	VARCHAR(128)									NULL

Figure 4.8: DB Tables/Schema

The User Table contains 5 + n columns, with the n being the number of sweetwords. This is because, in a similar manner to the study A novel Two-Factor HoneyToken Authentication Mechanism [4], the number of sweetwords in this specific implementation is arbitrary for proof of concept. However, the number of sweetwords should be substantial to function correctly and effectively. The other columns are an auto-incrementing user ID, email, hashed password, fingerprint, and the images for that specific user. This implementation allows users to have several fingerprints by having several tuples for one user. This means that the foreign key for the honeychecker is the email for a user, not the user ID.


HoneyChecker

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
user	VARCHAR(45)	✓	✓						
correctSW	VARCHAR(128)		✓						

Figure 4.9: DB Tables/Schema

The information in the table shown in figure 4.9 would be stored in a second server known as the Honey Checker. The honeychecker table only has two columns: email and correct sweetword. Despite the small amount of information held here, it is essentially the key to how HoneyTokens are implemented. Without it, the back-end would not be able to validate the index of the correct graphical password.

The logic implemented in the password-login.js file works in coordination with the database. If there is no correlating hash with any of the sweetwords, it is likely simply a wrong input. If there is one, the index is checked against the HC. If the index is correct, the user is authenticated. If it is not correct, an alert email is sent to the admin for the system.

5 Results

This section of the research will present the results found for each of the methods that were conducted. These results will also connect to the research questions. RQ1 was answered throughout the entire report and was thoroughly described in section 3 where a more comprehensive introduction to the current state of MFA was given. The different authentication factors were defined, and their properties were described in section 3. Section 4 helped answer RQ1 by utilizing all theoretical information from previous and later sections and putting them together to create a prototype that gave more insight into the development process of MFA mechanisms. Results from RQ2 contributed to answering RQ1. These questions are tied together inherently since this mechanism is only secure as its components. Answers for RQ2 can be found in section 5.1 and all its subsequent sub-sections where the security of MFA was introduced, and the impact of current threat models on the proposed solution was presented. After that section 2.2.2 and 5.1.3 the security of different aspects of the created prototype and the proposed mechanism was defined and parameterized to answer RQ2 and more so to answer RQ3. Results of the interviews with security experts and end-users testing interviews can be seen in sections 5.2 and 5.3 where RQ4 was answered.

5.1 Limited Literature Review Results

Results from the conducted limited literature review are discussed in this section. These results help answer RQ1, RQ2, and RQ3. This section begins by introducing the security of MFA mechanisms and the impact of current threats on the proposed solution. A requirements checklist is then presented. Finally, the proposed mechanism is checked against the requirements developed to determine its feasibility and security.

5.1.1 The Current State of the Security of MFA Mechanisms

The security of current MFA mechanisms is dependent on the types and frequency of attacks for that system. A simplified way to analyze this is to consider when attacks are successful and compare that to the factors used. This is what Google did in 2019 when investigating account takeovers [42]. The results presented in this study showed that the prevention rates for automated bots, phishing, and targeted attacks were higher for device-based challenges than knowledge-based challenges [42]. Of the device-based factors presented, a physical security key was found to be the most effective at preventing account takeovers [42]. None of the users investigated in the study that exclusively used physical security keys were found to be targeted for phishing in the study [42]. Secondary authentication applications (called On-device prompt in the study) were found to be the second most effective [42]. SMS was found to be more effective at preventing account takeovers than a secondary email address, and the knowledge challenges of location and phone number were found to be the least effective factors [42].

An example of exploiting one of these vulnerabilities is when Signaling System 7 (SS7) vulnerability was used to bypass 2FA for German bank accounts. This vulnerability was exploited in 2017 when attackers used this vulnerability in the cellular network to intercept OTP sent via SMS and execute a Man-in-the-Middle (MITM) to make unauthorized transfers from users' bank accounts [43].

There are common threats for every authentication factor. Knowledge-based factors can be vulnerable to several types of malware like key-loggers and screen capturing as well as in-person exploits such as shoulder surfing [23]. Other attacks on knowledge-based factors can be made through brute force, and dictionary attacks [23]. Knowledge-based factors are also more vulnerable to social engineering attacks such as phishing [44]. Possession-based factors are vulnerable to loss, and theft [23]. Biological factors are vulnerable to forgery and replay attacks [23]. An attack vector that is prevalent regardless of the type of factor is MITM attacks [23]. There have been several implementations that attempt to prevent the possibility of MITM attacks [23]. This attack, however, is effective at infiltrating even protocols that are considered secure [45]. The most common method used to combat MITM and eavesdropping attacks is through the use of a second channel with one-time challenges [46]. There are also specific attack vectors for particular implementations of graphical passwords specifically. One example of this is Frequency of Occurrence Attacks (FOA), where a GP can be determined by eavesdropping and using frequency analysis to eliminate the decoy images sent in the step [8]

There have been guidelines developed for deploying and developing authentication mechanisms. The NIST SP 800-63-3 is the Digital Identity Guidelines that is aimed at providing requirements for authentication in Federal agencies in the united states [47]. These guidelines cover more than the authentication mechanism itself, such as determining what mechanism level is needed for a specific environment. In relation to what level of security is needed in an environment, the NIST refers to the authentication mechanism itself as Authenticator Assurance Level (AAL). AALs are separated into three tiers of security for different needs. ALL1 is described as requiring *some* assurance of authentication and can be a single-factor authentication mechanism [47]. ALL2 requires *high confidence* for authentication through approved cryptographic systems and must include two separate factors [47]. ALL3 has the same requirements as ALL2 with additional constraints on the cryptography used and a hardware token [47]. There have also been more generalized requirements developed for specific types of factors. "Knowledge-Based Authentication Requirements" presents four more generalized guidelines for requirements of authentication mechanisms [46]. These requirements relate to eavesdropping, prediction, guess-ability, and server vulnerability [46].

In general, most reputable authentication mechanisms are considered to be secure. MFA is used to combat impersonation and unauthorized access. Experts in the field understand this additional layer as a tool to be used on all possible accounts. At the same time, end-users see MFA as an optional security service with primarily work-related use [48].

5.1.2 The Impact of Current Threats on the Proposed Solution

Many of the threats to current authentication mechanisms also impact the proposed mechanism. Attacks for these mechanisms would also likely be used for this solution. However, due to this specific combination of authentication factors, it is plausible that this combination would mitigate some of these threats. For example, this mechanism would be vulnerable to key-logger malware for the password factor and possibly the graphical password. However, it would not circumvent the fingerprint factor alone. This solution could be vulnerable to phishing attacks; however, the information given to the attacker would be useless unless they also steal the device itself or replicate the browser's fingerprint. This entails that a combination of common threats would be needed to authenticate another user for this mechanism. Several of these attack vectors have been analyzed in more depth in the Requirements Checklist in Section 5.1.3

Social engineering attacks would affect this mechanism in a very similar manner as other MFA solutions. Phishing, for example, could result in an attacker gaining the plain text password and less likely the GP; however, the attack would then also need to gain the fingerprint for that user. As mentioned, possession factors for authentication are vulnerable to theft. Because of this, some combination of in-person social engineering attacks could be quite effective against this mechanism. Despite this, the device-based factors are more effective at combating impersonation.

MITM attacks on systems are a threat that impacts MFA solutions. There is a possibility that an attacker would target the application layer TLS connection to intercept the knowledge-based factors for this system. This attack would also depend on the user accepting the forged certificate sent by the attacker [45].

5.1.3 Requirements Checklist

This section focuses on the proposed mechanism and the factors making this solution, such as browser fingerprints and graphical passwords. The requirements checklist is a security-focused list of must-have attributes for the proposed solution to be considered valid, secure, and feasible. The developed artifact discussed in section 4 was also part of proving that the proposed solution was feasible; hence, some aspects of the prototype will be evaluated using the requirements checklist. This section will not dive into alphanumeric password requirements and best practices as it had been thoroughly discussed and analyzed in the literature over the years [49, 50, 51]. The requirements checklist is divided into three sub-lists. The first one handles the position factor (the browser's fingerprint) of the proposed MFA mechanism. The second list concerns the knowledge factor in the proposed solution. The last sub-list contains the general requirements of the prototype and the proposed mechanism. The final list also contains the shared requirements and should be fulfilled by the graphical passwords element, browsers' fingerprints, and honeytokens.

There are many desirable attributes for a browser fingerprint to be used in authentication [6].

Browser's fingerprint requirements checklist:

1. *Stability:* Meaning that the browser's fingerprint does not change frequently. Some elements of a browser fingerprint are less stable than others. For example, using the HTTP user-agent header for authentication will create issues since it is not a stable attribute due to continuous web browser updates. Usually, a stable fingerprint is a requirement for using it in an authentication context, so end users do not change to use some other way to authenticate as OTPs sent by email every time there is a fingerprint change [6].

2. *Consistency:* Assuming no changes are made to the device/browser being fingerprinted, the fingerprinting algorithm must always collect or receive the same results [6]. This is a critical requirement for successful authentication attempts.

3. *Overhead:* The used fingerprinting algorithm should not be resource-intensive since users require a swift and efficient authentication process. The fingerprinting method should not take much time to collect or receive the desired information. Some fingerprinting methods depend on measuring a system's performance, which may take some time. If this type of fingerprinting were to be used for user authentication,

the users would most likely opt for a faster solution. That is why it is recommended to use a fingerprinting method with a low impact that takes little time [6].

4. *Requires a user's input:* Having to ask a user for their input to collect or send a fingerprint is another step that the user does not want to take to access the wanted service. Thus having the process as seamless as possible is desirable.

5. *Personally identifiable:* To be able to use a browser's fingerprint in an authentication context, the fingerprint must be distinguishable from other possible devices/browsers [6]. For example, using the timezone as a fingerprint for authentication while all service users are in the same timezone is a bad idea since all users will have the same fingerprint [6]. A fingerprint is more personally distinguished the more vectors are used in creating the fingerprint, but this makes the fingerprint less stable [5].

6. *Vulnerable to guessing attacks:* If a user could simply guess the fingerprint, that would make it undesirable. For example, using timezone as the only vector in a browser's fingerprint makes it extremely easy to guess the fingerprint and possibly bypass the fingerprint authentication step.

7. *Vulnerable to spoofing attacks:* An attacker could try to copy a device's fingerprint by using the same software and hardware as the legitimate user. An attacker could gain the browser fingerprint by other means and use it, or simply if the fingerprint is static, the attacker could use simple replay attacks [6]. These vulnerabilities should be considered before choosing a fingerprinting algorithm.

Graphical passwords requirements checklist:

1. *Shoulder Surfing Attack:* An attack vector that is common for recognition-based graphical passwords where the attacker watches the user enter the password and then knows the password [8]. One way to attempt to mitigate the attack is through randomization, although this does not always eradicate the threat [8].

2. *Frequency of Occurrence Attack:* If a graphical password system uses randomized decoy images, the used images will need to be sent every time, leading to attackers eavesdropping and learning the password through frequency analysis [8].

3. *Brute Force Attack:* An attack vector where an attacker attempts a large number of passwords for a user in an attempt to gain access. This is also commonly combined with a dictionary attack, where the guesses are formatted by a dictionary of words and commonly used passwords [19].

4. *Memorability:* The extent to which a user can remember the graphical password itself. This is a desirable satisfaction requirement for users' appeal to the mechanism. This would also reduce the frequency of resetting passwords and negative connotations for end-users [23].

5. *Easy to use:* An appealing aspect of any service that is essential for increasing adoption. Also important to increase the number of possible users regardless of disposition (Age, disability, colorblind, etc.). This is a common desire for any application offered over the Internet and common practice for web development.

6. *Easy to Understand:* Another desirable aspect of any service. The ease of understanding an authentication mechanism is important in order to avoid negative connotations from end-users.

General Requirements Checklist:

1. *Vulnerable to phishing attacks:* A social engineering attack vector where the user is deceived into giving knowledge-based factors to the attacker. This is most commonly done over mass email or over phone calls (AKA Vishing/voice phishing) [19].
2. *Prototype testers were able to perform a successful login attempt:* A requirement that demonstrates the ability for the prototype to be used by end-users. Successful login with the correct credentials also speaks to the ability of the prototype to work as expected.
3. *Warning was sent to prototype admins when honeytokens were used:* A requirement that speaks to the ability for the prototype to work as expected with the use of the second channel. This requirement also demonstrates the honeytoken functionality of the mechanism.
4. *prototype sent OTP by email upon fingerprints mismatch:* A requirement that adds to the ability for users to be able to use the second channel to authenticate for new devices or under a fingerprint change. This also demonstrates that the mechanism functions as expected.
5. *Easy to Create:* If a mechanism is more accessible to create, it would be more likely to be implemented, ergo, increase the use of the system.
6. *Man-in-the-Middle Attack:* An attack vector that is not specific to any one factor, that is described in more depth in Section 5.1.1. Where an attacker eavesdrops on the information sent over a secure channel in order to exfiltrate credentials.
7. *Determined to be Feasible by Security Experts:* Security experts determining that this mechanism is a feasible solution adds to the credibility of this mechanism.

5.1.4 Compliance with the Requirements Checklist

The proposed mechanism, alongside some aspects of the created prototype, was examined against the requirements checklists presented in section 5.1.3. The outcome of this examination was taken into account when drawing conclusions about the usability, validity, security, and feasibility of the proposed solution. Section 2.2.2 gives a more detailed description of why this method was used and how it was conceived. The requirements from the previous section 5.1.3 will be put into a question form to be answered in this section. The analysis and discussion of the results found in this section can be seen in section 6.

Compliance with the browser's fingerprint requirements checklist:

1. *Stability:* Was the used fingerprinting algorithm stable? **Results:** According to prototype testing, the used Canvas fingerprinting algorithm was stable enough to be used for authentication. Alaca et al. [6] wrote that Canvas fingerprinting is partially stable.
2. *Consistency:* Did the chosen fingerprinting method yield the same results, on the condition that no hardware and software changes were made to the machines used in testing? **Results:** Canvas fingerprinting is repeatable and consistent according to prototype testing and as it was written in [6].

3. *Overhead:* Was the chosen fingerprinting method resource intensive? **Results:** Based on running the Canvas fingerprinting algorithm ten times on the prototype while measuring the execution time of the algorithm, it took on average 19ms for the algorithm to execute. Thus, the Canvas fingerprinting chosen was not resource-intensive. The outcome of these tests was corroborated by Alaca et al., who wrote that Canvas fingerprinting did not require long processing times [6].

4. *Requires a user's input:* Did the chosen fingerprinting method require the user's input? **Results:** The Canvas fingerprinting implemented in the prototype did not ask for any user input. The fingerprint is collected passively and seamlessly. Acar et al. [31] showed that only the Tor browser [52] would ask for a user's permission in case a script was trying to access the Canvas element.

5. *Personally identifiable:* Was the generated fingerprint personally distinguished? **Results:** Canvas fingerprinting is considered to be one of the fingerprinting algorithms that produces highly unique fingerprints compared to other methods on the internet [6, 27, 29].

6. *Vulnerable to guessing attacks:* Was the chosen algorithm vulnerable to guessing attacks? **Results:** No tests were done on the prototype to measure robustness against guessing attacks. However, the implemented Canvas fingerprinting generates a 128-bit hash that would be hard to guess without prior knowledge of the target. Alaca et al. [6] wrote that an attack might try to copy a user's system to obtain a similar fingerprint to the legitimate user. This would be hard to do to get a Canvas fingerprint since the attacker must obtain the exact same hardware, OS, browser, software, and drivers. It is, however, not impossible to do.

7. *Vulnerable to spoofing attacks:* Was the chosen method vulnerable to spoofing attacks? **Results:** No tests were done on the prototype to test for this requirement. An attacker could use phishing to obtain the fingerprint because one version of the prototype used the same query to collect the Canvas fingerprint from all users. Assuming that an attacker had collected a user's fingerprint, they could easily spoof and replay the fingerprint [6].

Compliance with the graphical passwords requirements checklist:

1. *Shoulder Surfing Attack:* Was the chosen GP method vulnerable to shoulder surfing attacks? **Results:** No specific tests were done to test if this prototype was vulnerable to shoulder surfing. However, due to the highlighting function implemented when a user selects an image, this step is likely vulnerable to shoulder surfing. Because the implemented prototype had static positions of the same images for all users, the implementation is even more so susceptible to this attack. However, this could be mitigated by adding additional features such as randomizing image position for the GP.

2. *Frequency of Occurrence Attack:* Was the chosen GP method vulnerable to FOA attacks? **Results:** The prototype would not be vulnerable to FOA attacks. This is because the implementation sends the same set of images to every user.

3. *Brute Force Attack:* Was the GP mechanism vulnerable to brute force attacks? **Results:** The GP step in the prototype had a 4x4 grid of images, where the user selected four images in order. Because of this the entropy for this password can be found using the permutation formula $P_{(n,r)} = \frac{n!}{k!(n-r)!}$ Where n is the size of the set,

and r is the number of selected items. The resulting entropy of the implemented prototype is the $P_{(16,4)} = 43680$. This means that there are 43680 possible unique passwords for this step. Moreover, the probability of randomly selecting one is 0.00002289377. This means that it is theoretically possible for a brute force attack to bypass the final factor, assuming that the attack does not enter a honeytoken before the correct password.

4. *Memorability:* Were the chosen GP memorable? **Results:** The memorability of the GP selected for the prototype was not tested over time. However, both interview methods gave unstable responses about whether users thought the GP was memorable. Due to the static positioning of the prototypes' images, some users found a pattern-based GP to be more favorable. End-user results can be found in more detail in Section 5.3.

5. *Easy to use:* Was the GP step for authentication easy to use? **Results:** The prototype was deemed to be sufficiently easy to use by end-users. However, many opinions for alternate implementation variations were recommended in both types of interviews.

6. *Easy to Understand:* Was the GP step for authentication easy to understand? **Results:** The prototype was deemed to be easy to understand by end-users. However, the GP was the most discussed topic with both end-users and security experts.

Compliance with the general requirements checklist:

1. *Vulnerable to phishing attacks:* Was the mechanism vulnerable to phishing attacks **Results:** The mechanism's knowledge factors are vulnerable to phishing attacks. Although it is theoretically more challenging for users to reveal their GP than their text-based password. Phishing attacks would also not circumvent the fingerprint factor for this mechanism.

2. *Prototype testers were able to perform a successful login attempt:* Were testers successfully able to log in? **Results:** All end-users were able to preform a successful login during the interviews. See section Section 5.3 for the results of the end-users interviews, and Appendix C for the instructions for testing given to the users.

3. *Warning was sent to prototype admins when honeytokens were used:* Was an alert sent to admin when honeytokens were used? **Results:** A message is sent through the second-channel to the specified account when honeytokens were entered during the GP step.

4. *prototype sent OTP by email upon fingerprint mismatch:* Was an OTP sent upon fingerprint mismatch? **Results:** Users had sent an email when there was a fingerprint mismatch. Users were then able to authenticate with the OTP received in the email.

5. *Easy to Create:* Was the mechanism easy to implement? **Results:** The researchers do not have much extensive web development experience, despite this, the mechanism was quite simple to implement due to the libraries used. Persons with more theoretical background or experience with web development and JavaScript would likely find this mechanism very easy to implement. However, the prototype's implementation consumed a significant amount of resources, and not all of the found improvements could be implemented. The most significant example of this is the

randomized position of images for GP and a personalized set of images for each user.

6. *Man-in-the-Middle Attack:* Was the mechanism vulnerable to MITM attacks? **Results:** The prototype was not tested for MITM attacks. However, it is theoretically possible for a MITM attack to eavesdrop on all of the factors used for authentication with this mechanism. The specific vulnerability would be dependent on the cryptographic encryption used for components in the mechanism, as well as if the attacker can successfully get the user to sign a forged certificate [45].

7. *Determined to be Feasible by Security Expert:* Was the mechanism determined to be feasible by security experts? **Results:** The prototype was deemed initially acceptably feasible after an explanation and demonstration of the prototype and authentication scheme.

5.2 Expert Interview Results

As initially described in section 2.2.5, this method was used in order to answer and add results to RQ3, and RQ4, as well as add possible input to the requirements checklist. This method resulted in two open-ended interviews that lasted between 45-70 minutes.

Our initial contact with recommended experts was reciprocated with modesty concerning the term *expert*, despite their amount of experience in the field. It is possible that that specific word choice was overbearing. The experts that were interviewed for this method had previous involvement in authentication mechanisms. Either with implementation in an organization or a more software development focus. Their responses were highly correlated with their domain's focus.

5.2.1 Expert A

At the time of the interview, Expert A was responsible for implementing an MFA implementation on top of a current architecture over a large organization. This was being done due to new regulations from the Swedish government. However, the implementation was not currently active for the organization. Because of the expert's domain and the open nature of the interview, the conversation was more focused on adopting MFA. However, this was not only about end-user adoption but also adoption for implementation on a larger scale for an organization. This resulted in a conversation about the additional challenges faced by persons implementing MFA for companies, for which there are many. Examples of these challenges discussed were: not all employees having company devices, automated account closing, having different authentication mechanisms depending on a user group, as well as the numerous compatibility considerations that must be taken into account for services and back-end.

With the proposed mechanism, Expert A was favorable in that the mechanism only required one device; the fingerprint required no user interaction and simplicity. The expert was also favorable of other variations of the implementation that were discussed. For example, a pattern-based password was discussed and was more favorable for memorability. The expert brought up several concepts that this framework had not considered should it be used in a real system. Primarily these concepts are related to the administration side of MFA. Things that had not been accounted for, such as if the user gets a notification upon a honey alert. There was a conversation relating to the challenges of needing continuous support for resetting passwords. This conversation began because Expert A said that the organization recently added that password reset once a year during the COVID pandemic. End-users were not fond of this change. This conversation examined resetting a user's password and how a user is authenticated then in this framework. In addition, what factors of identification were needed to create this account in the first place. This conversation also asked if this was possible to do from a remote location. The expert also expressed that a user interface for MFA needs to be *foolproof*. Even for complicated edge cases such as adding additional passwords for other services, accidental deletion fail-safes, etc.

Additional results for Expert A were that they recommended time studies to investigate speed in comparison to other solutions for the end-user. When asked about the most significant threat to MFA, they responded, "It is only one piece of a quite large puzzle." Furthermore, there are mitigations for other attack surfaces, such as vulnerability scanners. The expert also expressed the desire for an *all in one* solution. That would contain all necessary components out of the box for an authentication product to avoid in-house development.

Expert A's opinion can be summarized accurately with the quote, "It has to be easy; otherwise, users will find a way around it." As well as, "I think it is an interesting concept. I think it depends on how many images one has to remember".

5.2.2 Expert B

Expert B's career focus is on software security, and has had experience with authorization models previously. The conversation with Expert B gave different results than Expert A in that Expert B was not familiar with some of the terminology related to the mechanisms. For example, Expert B was unfamiliar with browser fingerprinting. Because of this, a significant amount of time in the interview was spent describing this concept. This involved describing browser fingerprinting theoretical background and expanding upon the concept of uniqueness. "What if someone uses the exact same device as me?" was asked for example. The conversation then continued into a conversation about the uniqueness of a fingerprint. Moreover, how the parameters from the queries are not solely dependent on hardware. This was expanded upon with a description of Canvas fingerprinting and the possibility to use dynamic queries similar to A Morellian Analysis of browsers [12].

Expert B gave similar results concerning end-user usability as end-users and Expert A in that the graphical fingerprint requires memorization of an unfamiliar password. They also expressed that this may be more challenging to remember than text-based passwords due to the users' inability to select the set of possible images. However Expert B also expressed that, "It's not more than what we have today", in relation to other MFA/2FA mechanisms such as TOPT. Expert B recommended some alternatives to the Graphical Passwords, such as selecting images based on image categories rather than specific memorization. For example, selecting a building and sending different buildings each time. Expert B was also more favorable of a pattern-based implementation and alternative implementation variations.

In relation to feasibility for security, Expert B was hesitant to give a definitive answer and expressed that far more details would be needed to answer that question. Specifically, the software itself, as this was the expert's domain. However also mentioned, "I don't see any immediate flaw, at first look it seems like it would work.". This answer aligns with the Experts background in software security as an in-depth code analysis would not be possible within the time frame, and the prototype was done for proof of concept.

5.3 End-Users Interview Results

Section 2.2.4 introduced user testing interviews as a method that was used to evaluate the satisfaction of end-users with the proposed solution. These interviews were also interested in the users' past experiences with other MFA solutions. This section will answer RQ4 by sharing the end-users answers to the interview questions shown in appendix A.

In total, 12 interviews were conducted. The testers had been informed that the interviews were going to be recorded. They were also given the consent shown in appendix B that explains what information is being collected and that they can leave the interview at any point during or ask not to use their answers at any point after the interview. After that, the proposed solution was described briefly. The tester was then handed the instruction (appendix C) to conduct a login attempt on the prototype. The last step of the interview was to ask the users the interview questions (appendix A). The remainder of this section will list interview questions and the corresponding users' answers. When referencing a specific user, a letter will be used instead of their name to guarantee their anonymity.

1. *Do you study, work, or something else? What is your study/work domain?*
   Answers: 3 of the participants were design students, and the remaining 9 participants were computer science students.

2. *Have you ever (Or are you currently) use any multi-factor authentication MFA solution for work, studies, or personal use?*
   answers: All of the participants have or are using an MFA solution at the time of the interview.

   (a) *If you have (Or currently) use an MFA, can you tell us the authentication factors used by that solution?*
   Answers: All 12 participants had alphanumeric passwords as the first authentication factor. More variation can be seen when it came to the second authentication factor that participants used. User **H** and **F** used a Time-based one-time password TOTP using the Google Authenticator mobile application [53]. User **B** had also used similar TOTP implementations to **H** and **F**, but they also used hardware authentication tokens. The rest of the users have used a variety of different methods, but the dominant second factor used as a code sent to them via Short Message Service SMS to their smartphones.

3. *If answered yes to 2 ask the following:*
   Answers: All participants have used a multi-factor authentication solution at one point or another in their lives. Thus, the following question was asked to all 12 participants.

   (a) *On a scale of 1 to 5 how satisfied are you with the solution you have used before?*
   Answers: Users **B, C, D, F, H, I, L** gave a satisfaction grade of 4-5 on the scale. Testers **A, E** have answered with 3. Testers **G, J, K** have answered with 2.

(b) *Can you tell us one thing you would like to change about the solution you have used before?*
Answers: User **A** said that having to use their phone to check an SMS for a code made them lose focus on what they were working on. User **A, I, K, L** have also complained about the need to have a phone on them the entire time. The four users have also shared concerns about having to have coverage on their phones to receive the SMS with codes. They said that sometimes they may not have cellular coverage while still having Internet to login to a service. Users **C, H** Had no complaints about the solution they were using. User **B** did not mind having to use their smartphone for the second factor of authentication. However, they did say that they would rather have to scan a QR code instead of inserting a TOTP by hand.

User **D** were more concerned about the security of their smartphone because they must have it secure. Using their phone for authentication would increase the attack surface.

User **E** thought the time limit on their TOTP was too short because they were miss-authenticating. Due to the short time of the TOTP, they would like to have the timer on their OTPs to be longer.

User **F** had a problem with the compatibility of authenticating applications that implement TOTP MFA. In their experience different web services that they use support different TOTP authentication applications. They would like to have a universal cross-platform TOTP MFA solution.

User **G** did not have any specific complaints, but they wanted the authentication process to be more efficient. User **J** would like to have the process be faster because they share credentials to a service with a family member

4. *Questions regarding our implementation:*

(a) *On a scale of 1 to 5 how satisfied are you with our solution?*
Answers: users **A, D, E, F, G, H, I, J, K, L** have given a satisfaction grade of 5. Tester **C** answered with 4. Tester **B** has not been satisfied and gave a satisfaction grade of 2.

(b) *Would you be willing to use this solution in your daily life?*
Answers: All testers have answered yes to this question.

(c) *This implementation uses your email for unrecognised devices. Does this make you more or less likely to adopt this solution?*
Answers: All users had no problem with the use of email and a graphical OTP to authenticate a new device.

(d) *Can you tell us one thing you would like to change about our solution?*
Answers: tester **A** wanted to change the quality of the graphical passwords. They wanted the tokens to be clearer. Tester **B** did not like the graphical password element of the prototype since it forced them to remember one more thing on top of remembering the text-based password. Tester **D** was interested in having the graphical password authentication step be more of a pattern than a password. Tester **E** discussed the accessibility aspect of the graphical password factor. They have suggested making the graphical passwords more color

blindness friendly. Tester **F** also thought about the accessibility of the graphical passwords. They have also stated their interest in having simpler and more recognizable graphical tokens. Tester **G** thought that it would be hard to remember the graphical password and that it might have been easier if a pattern had been implemented. Tester **I** have also expressed concerns regarding the challenges in remembering the graphical password. They mentioned that it would be a problem remembering different graphical passwords across different platforms. Tester **J** thought that having a specific order in the graphical password input is slightly confusing, and they thought that it should be clearer what token did a user choose in what order. Tester **K** and **L** believed that more personalized graphical tokens would make it easier to remember. Testers **C** and **H** did not have an answer for this question.

5. *Do you have any other ideas/concepts that you would like to tell us about this system that have not been brought up in previous questions?*
   Answers: User **A** has like the idea behind the prototype, while tester **B** was not convinced of the proposed mechanism. Tester **C** did not like having to memorize the potential different graphical passwords across multiple platforms, and they liked that order matters in the graphical password element. Tester **F** had enjoyed the fast login process and thought that memorizing the graphical passwords would not be a problem. On the other hand, tester **G** thought that the graphical passwords would be hard to remember but enjoyed the simplicity of the login process. Tester **H** was more security-oriented and thought that this MFA mechanism would provide great security and ease of use. Tester **J** stated that she liked the graphical passwords more than the traditional use of numerical codes. Tester **L** was keen on discussing accessibility issues. They focused on how this solution would be used by the elderly or people with memory problems. Testers **D, E, I,** and **K** did not have an answer for this question.

6 Discussion and Analysis

Analyzing and discussing the report's findings and answers to the following research questions is done in this chapter.

RQ1 What is the state of the art of MFA solutions that use Browser Fingerprinting, Honeytokens, and Graphical Passwords? How does this solution build on existing solutions?

RQ2 What are current threats to MFA, and how do they impact this solution?

RQ3 What would be the necessary requirements for this solution?

RQ4 Would end-users be likely to adopt this mechanism?

RQ1 is partly answered in chapter 3.1, where the concept of MFA is presented, and different authentication principles are defined. Authentication factors such as the knowledge factors, possession factors, etc., are introduced, giving an insight into the current state of MFA. Later, in sections 3.2, 3.3, and 3.4, the rest of RQ1 is answered by introducing the different factors of the proposed mechanism to show how the proposed MFA solution builds on existing ones. First, browser fingerprinting is introduced. Fingerprinting is used on the web for user tracking. There were some attempts to use it for security. However, this technology remains only in the domain of tracking to make advertisers money, even though browser fingerprinting shows much promise in many security-related applications. Different vectors can be used to collect a browser fingerprint that can be sufficiently unique to identify a device on the internet. Canvas fingerprinting is presented and is chosen for the prototype developed in section 4. Then, graphical passwords are discussed. There are three systems for creating graphical passwords: recognition-based, pure recall-based, and cued-recall. Each of these systems has advantages and disadvantages. Recognition-based is the graphical password chosen for the prototype. Lastly, the concept of honeytokens is explained in section 3.4. End-users do not use these tokens, but they are there to alert system admins of database breaches. Section 4 shows how all the concepts explained in this paragraph come together to create a prototype of the proposed mechanism, completing the answer for RQ1.

RQ2 is answered in section 5.1.1. Knowledge factors in an MFA system are more likely to be attacked with phishing and other targeted attacks. Public and private entities are constantly publishing guidelines for creating secure MFA systems. Dominant MFA solutions in the market are considered to be secure. Threat models are also presented, and later in section 5.1.2, the impact of such threats on the proposed mechanism are discussed. Results from RQ1 and RQ2 may have some reliability and validity issues due to not conducting a systematic literature review and doing a limited literature review instead. This may have led to missing some crucial details regarding the current state of MFA and its security.

RQ3 is answered in section 5.1.3, where all security, feasibility, and usability desired attributes are presented. A browser's fingerprint stability, consistency, and distinguishability are amongst the list of preferable attributes that this mechanism and browser fingerprinting algorithm should comply with. Memorability, ease of use, and resistance to different types of attacks are amongst the required attributes that the graphical password factor should fulfill to consider the mechanism valid and feasible. General desirable attributes are also presented. Amongst the general attributes are the resistance to phishing attacks, tester accomplishing successful login attempts, etc. The positive opinion of the interviewed security experts regarding the mechanism security and feasibility is also a requirement included in the checklist. Section 5.2 shows that experts think that mechanism is acceptable. Compliance with the requirements checklists is present in section 5.1.4. The proposed mechanism complies with the majority of the requirements. Results from RQ3 are valid and reliable since the experts' interviews provide external validity, and the requirements checklist provides internal validity. All results from answering RQ3 can be reproduced if the methodology described in section 2 is followed. The results from RQ3 are generalized since a similar requirement checklist can be used by someone trying to implement an MFA mechanism using browser fingerprinting, graphical passwords, and honeytokens functionality. A developer may not make the same choices when implementing their prototype regarding fingerprinting algorithm or graphical password grid size. However, the requirements checklist remains valid, reliable, and generalizable regardless of implementation choices.

RQ4 is answered in section 5.3, where some users convey dissatisfaction with the solutions they currently use. A minority of the users are satisfied with MFA solutions they are using. All users are willing to adopt this solution and use it in their daily life. Security experts in section 5.2 discuss the usability of the prototype and the mechanism in general, focusing on the graphical password element. The experts like that the authentication happens on the same device. The results from answering RQ4 may be less reliable and valid compared to results from RQ3 because of issues with sampling and with end-users all being students and a majority being computer science students. However, having the majority of computer science students as end-users provides a significant advantage as they have a deeper level of understanding for authentication than a regular user and can offer more nuanced answers. The results from RQ4 are not generalizable because of the end-users sample size.

The aim of this study is to determine whether the proposed mechanism is feasible or not. This report also focuses on end-users perspectives. Section 1 covers the aim and focus of this work. In this section, the focus is on discussing and analyzing the results related to the mechanism's feasibility and end-user adoption sentiment. The advantages and disadvantages of the proposed authentication factors (based on results from section 5) are discussed from a feasibility point of view with a focus on security by evaluating the compliance of the mechanism with the requirements checklist presented in section 5.1.3. End-users satisfaction and willingness to adopt the proposed solution are also discussed for each authentication factor and for the entirety of the solution represented by the prototype developed in section 4. The results from end-users interviews can be seen in section 5.3.

6.1 Feasibility from a Security Perspective

The feasibility and the security of the proposed solution go hand in hand. An authentication mechanism cannot be feasible if it is insecure. To have a secure MFA solution, all factors must be secure. The following is an analysis and discussion of the feasibility and security of the different elements of the proposed solution.

The chosen Canvas fingerprinting algorithm has its advantages and disadvantages. Based on findings in section 5.1.4, Canvas fingerprinting was found to be an appropriate browser fingerprinting method to use for authentication. Canvas fingerprinting complies with almost all requirements from the requirements checklist. While Canvas fingerprinting is fast, stable, and consistent, it is still vulnerable to trivial attacks like spoofing attacks. Since an attacker could phish a Canvas fingerprint and use it to try and log in, fortunately, in the proposed mechanism, the attacker will not gain much by obtaining the legitimate user's fingerprint since the attacker will need to know the graphical and possibly the alphanumeric password. The browser fingerprinting method used in the prototype could be improved by creating more complex, customized, and personalized queries such as the ones proposed in [12]. However, due to time limitations, it was not finalized. When choosing a fingerprinting algorithm, most guidelines presented in the work of Alaca et al. were considered carefully, and Canvas was found to be the most suitable alternative. The proposed mechanism has the potential to become more secure by querying the clients' devices for a Canvas fingerprint continuously during the session to stop session highjacking attacks. If the fingerprint is changed during the session, then the session is ended, and the user is asked to re-authenticate. Using fingerprinting for authentication does not have to be used in the exact manner shown in the developed artifact. On the contrary, readers interested in implementing such a solution have the freedom to employ browser fingerprinting technology in many ways as described in this paragraph.

Canvas was found to be consistent and stable, but that does not mean that the Canvas fingerprint will not change over time. For this reason, the mechanism proposes implementing a second channel, such as sending OTPs to a user's email to verify the new fingerprint. An improvement on OTPs sent by emails would be implementing a fingerprinting algorithm that takes more than Canvas fingerprinting as a browser's fingerprint vector. Then, the algorithm checks if there is a significant change in the fingerprint to determine whether or not it should authenticate a user. However, that does not mean that the developed artifact is lacking, but it can be improved from this point to make the user experience better. Implementing such a comprehensive algorithm will potentially improve the overall security of the system since including multiple fingerprinting vectors will make it harder for attackers to use naive guessing attacks like the ones discussed in section 5.1.3.

Concerning the security of graphical passwords, this mechanism can be secure, however, there are alternate GP methods or variations on this implementation that can be used to mitigate more attack vectors. Three separate possible variables impact the security for this factor that has been drawn from the interviews and LLR in Section 5. The first is if the order matters for the selected images. This change significantly affects the password's entropy and, most significantly, how effective the factor is at combating brute force attacks. If the order does not matter for the prototype, for example, the number of possible passwords will drop drastically to $C_{(16,4)} = 1820$. Meaning that with combination the number of possible passwords would drop to an almost unacceptable number. In comparison with the commonly known PIN code would be $P_{(10,4)} = 5040$ or 0.00019841269 probability of selecting a random code. The prototype GP implementation has 38640 more possible unique passwords than four digit PIN codes. The second variable is whether or not the images are in random positions each time the user authenticates. Randomizing this assists in combating shoulder surfing; however, it does not eliminate it. It can also cause brute force attacks to need to be more complicated in effectively attempting possible passwords. It also would create a system with less guessable graphical passwords such as rows, columns, or diagonals. The final variable is whether or not the images for each user are personalized. This variable can mitigate shoulder surfing by having the images be less recognizable by people other than the user of that account. If the set of images is personalized for the user, the set should be the same upon every authentication without decoy images in order to avoid frequency of occurrence attacks. These three changes, along with removing highlighting selected images, the number of images presented, and the length of the GP itself, have a significant impact on shoulder surfing and brute force attacks for this factor. It is also important to consider that these changes can impact how resource-intensive the mechanism is. It also depicts that this MFA mechanism has its strengths and weaknesses compared to other attempted recognition-based graphical passwords. This is best depicted when comparing this mechanism to other MFA solutions analyzed in compiling research [13, 8, 26].

This research did not produce as significant controversy in interviews with end-users or security experts as the other features. This is understandable for end-users as they have no interaction with that aspect of the mechanism. However, security experts provided no additional feedback concerning the concept outside of Expert A, wondering if users are also notified upon a honey alert. Testing and the general requirements checklist from section 5.1.3 depict that the honeytokens function as expected. The honeytoken's security depends on the number of sweetwords used in the database and has not been specifically tested; however, it should have significantly more to be secure upon an authentication server breach. This feature, in comparison to the study that inspired this research, A novel Two-Factor HoneyToken Authentication Mechanism [4] furthers the research by integrating the honeytoken functionality in another platform.

Several methods showed that a singular device authentication mechanism is favorable from a usability perspective. This is depicted in several end-user interviews and Expert A's interview. However, this convenience does impact the security of the mechanism. Having the mechanism rely on one device leads to a smaller attack surface from the client-side of the authentication process. Meaning that should the device become compromised, it could negatively impact the mechanism's security. A singular communication channel also similarly has a smaller attack surface. While this mechanism utilizes the second channel of email, should an intruder monitor all traffic from a device, this second channel would also be compromised. This is less secure from eavesdropping in comparison to an SMS bases second channel, for example. However, this would also lead to a second device. Unless, of course, the user was using a mobile browser.

6.2 Feasibility from a User Perspective

These findings for GP security may appear to be counter-intuitive due to the concept that a majority of the requirements in the requirements checklist have been met. However, the results from the open aspects of both types of interviews accurately represent the changes needed to be made for the mechanism to be feasible from a usability perspective. This is best depicted in the interviews. The end-users who decided to answer question 4(d) discussed the GP factor. As seen in Section 5.3 question 4(d), participants recommended changes to the selection of images, challenges with memorability, and solutions that did not require images. Both security experts favored having a pattern-based password rather than the selected GP. Due to the GP implemented in the prototype, the entropy would not change with that alteration. However, this would no longer be able to have randomly located images and remove the mitigation to shoulder surfing.

There is no immediately obvious combination of changes that are secure and appealing to end-users. However, these variations of implementation exhibit the challenges of balancing security with usability as they can have a significant effect on how users perceive the mechanism. Permutation and randomized location of images, for example, can be a more secure implementation for brute force; however, this can have the effect that users take longer to find the images, making shoulder surfing easier and negatively impacting the appeal of the factor. The most appealing combination of these factors could be combination with static locations as this would be the fastest to input; however, this would negatively impact the mechanism's security. As Expert A mentioned, users will attempt to circumvent security for convenience. This balance is also depicted in usability because this MFA mechanism only requires one device. This was found to be a favorable feature by Security Expert A and several end-users.

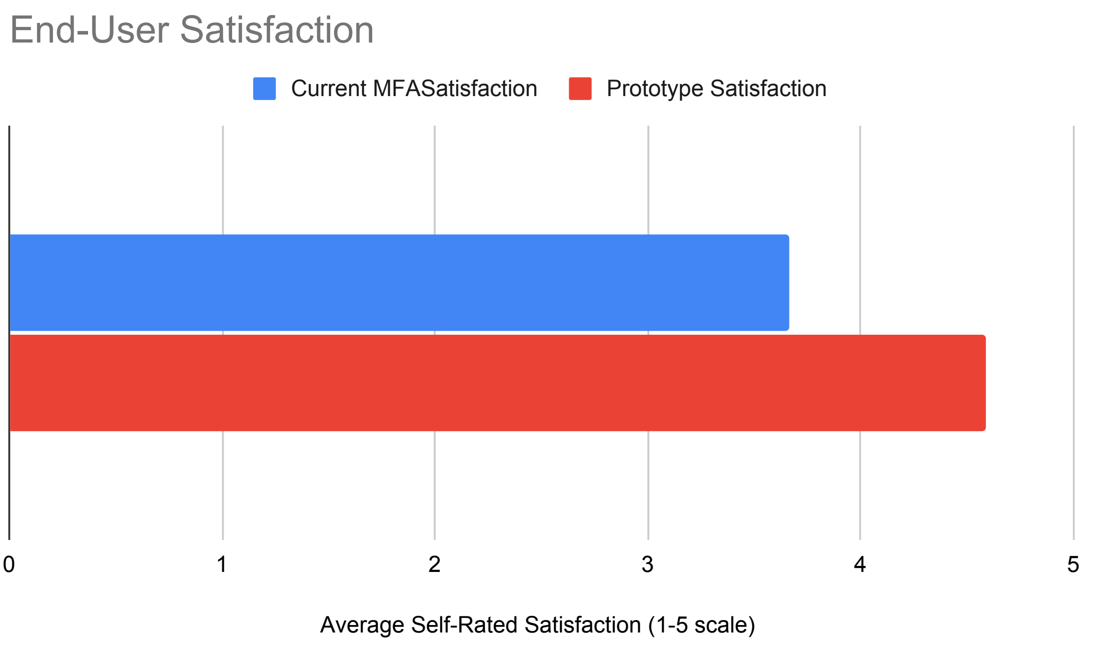


Figure 6.1: End-User Satisfaction from Prototype Testing

The end-users reported 18% higher satisfaction with the prototype than their current MFA solution as shown in figure 6.1. This speaks to user negative connotation for current MFA solutions found in research investigating user perception [1, 48, 3, 14]. It also speaks to the importance of investigating user satisfaction early on in the investigation process as user appeal is an essential aspect of if the solution is adopted. However, these results may not be able to be generalized due to the limited number of participants, a majority of these participants also already use MFA solutions, and many have a computer science background. This impacts the ability to draw these conclusions on users with no experience with MFA solutions and persons without technological backgrounds. The limited number of end-users interviewed and the grouping of backgrounds make these findings less generalizable.

7 Conclusion

This research has provided answers to the research questions presented in Section 1.3. Some provided results can be more generalized than others, as it is shown in section 6. Based on the results in section 5 and analysis in section 6, the proposed mechanism is considered feasible. No similar mechanism is found in the literature. Thus, more testing is necessary to deem this mechanism appropriate for real-world application. Of course, there is a possibility that a similar authentication mechanism already exists, but it was not found in the Limited Literature Review. The feasibility conclusion is tied to the requirements checklists developed from conducting a Limited Literature Review. Using design science to guide the process of developing and evaluating the mechanism and the prototype is crucial in determining the proposed MFA solution feasible. The last factor that helps prove feasibility is security experts' input that finds the mechanism feasible.

Results in section 5.3 show that users are generally satisfied with the usability of the prototype. End-users have also stated that they are willing to use this solution in their day-to-day lives. These results give tremendous value to the validity of this mechanism. It also fills a significant gap in the knowledge where often MFA solutions are not evaluated by end-users. These results do not reflect the opinions of the entire society but rather a small sample of students. More extensive user testing should be done to be able to measure users' attitudes and willingness to use the proposed solution. However, the scope of a bachelor's project limits conducting such interviews on a large and diverse sample of users.

Since no large-scale testing of the proposed mechanism has been done, it cannot be considered ready for real-world deployment. That does not mean that the proposed solution does not offer any value. On the contrary, since end-users are reluctant to adopt MFA, having this solution where all authentication steps happen on the same device can offer great value to the industry where secure authentication is critical such as the banking industry. By introducing a feasible MFA mechanism, this study is also relevant to society, since end-users are part of society. As for contribution to science, this paper introduces an MFA solution that uses browser fingerprinting, graphical password, and honeytoken functionality. This mechanism is deemed feasible, and user sentiment is positive based on end-users evaluation of the solution. Thus, the knowledge gap described in section 1.3 is bridged.

The results of the study can be generalized and used in the area of computer security and specifically in authentication and access control. Security professionals working with MFA can use the results from this study to implement the same mechanism or make different choices when it comes to different technologies. For example, Canvas fingerprinting can be exchanged with a different browser fingerprinting method as long as the new chosen method fulfills the requirements in the checklists. The same works for graphical passwords where people implementing similar systems have the ability to choose different size grids or other types of graphical passwords such as Recall-based systems on the condition that the chosen technologies and variations comply with the requirements checklists in section 5.1.3.

7.1 Future work

In this section, a way to forward the research is presented. Time and resources are factors that limit this bachelor project. Some parts of this study can be done better if enough time and resources are available. This section describes what can be improved and gives recommendation for further research.

The Canvas fingerprinting method chosen is valid and reliable. However, more unique and stable fingerprinting methods can be developed by using multiple vectors to fingerprint a user's browser. A browser fingerprint that is more personally distinguished makes the entire mechanism more secure since it will limit false-positive cases in case of fingerprints matching between users. It is also more secure against fingerprint guessing attacks. A more stable browser fingerprint makes the experience better since they do not have re-authenticate a device as frequently as they would when a fingerprint changes.

The open aspects of the interviews with end-users and security experts have drawn focus to the selected graphical password that has been used in the prototype. The particular implementation may be able to be improved through variations. These variations may improve the appeal and feasibility of the mechanism as a whole. If these variations are used for future research, they would need to evaluate the mechanism's security as a whole again. An alternate option also could be to remove this factor in its entirety and change the mechanism to a 2FA implementation. Increasing the size of the participants may also draw more reliable and generalizable results on this mechanism. An additional possibility for future work may be to determine how many honeywords would be necessary to detect server-sided breaches most effectively. This specific feature of the mechanism has not drawn significant evidence for its feasibility in a large-scale implementation and could lead to more research. Future research for this feature could also be focused on the resources consumed by this feature.

References

[1] S. Das, B. Wang, Z. Tingle, and L. J. Camp, "Evaluating user perception of Multi-Factor authentication: A systematic review," 2019.

[2] S. Das, B. Wang, and L. J. Camp, "MFA is a waste of time! understanding negative connotation towards MFA applications via user generated content," 2019.

[3] J. Dutson, D. Allen, D. Eggett, and K. Seamons, "Don't punish all of us: Measuring user attitudes about Two-Factor authentication," in *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2019, pp. 119–128.

[4] V. Papaspirou, L. Maglaras, M. A. Ferrag, I. Kantzavelou, H. Janicke, and C. Douligeris, "A novel Two-Factor HoneyToken authentication mechanism," in *2021 International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 2021, pp. 1–7.

[5] P. Eckersley, "How unique is your web browser?" in *Privacy Enhancing Technologies*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–18.

[6] F. Alaca and P. C. van Oorschot, "Device fingerprinting for augmenting web authentication: Classification and analysis of methods," in *Proceedings of the 32nd Annual Conference on Computer Security Applications*. New York, NY, USA: ACM, 2016.

[7] A. Durey, P. Laperdrix, W. Rudametkin, and R. Rouvoy, "FP-redemption: Studying browser fingerprinting adoption for the sake of web security," in *Detection of Intrusions and Malware, and Vulnerability Assessment*. Cham: Springer International Publishing, 2021, pp. 237–257.

[8] J. G. Kaka, O. O. Ishaq, and J. O. Ojeniyi, "Recognition-based graphical password algorithms: A survey," in *2020 IEEE 2nd International Conference on Cyberspac (CYBER NIGERIA)*. IEEE, 2021, pp. 44–51.

[9] T. Unlu, L. A. Shepherd, N. Coull, and C. McLean, "Poster: Angry birding: Evaluating application exceptions as attack canaries," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2021, pp. 701–703.

[10] S. Srinivasa, J. M. Pedersen, and E. Vasilomanolakis, "Towards systematic honeytoken fingerprinting," in *13th International Conference on Security of Information and Networks*. New York, NY, USA: ACM, 2020.

[11] Y. Song, Q. Huang, J. Yang, M. Fan, A. Hu, and Y. Jiang, "IoT device fingerprinting for relieving pressure in the access control," in *Proceedings of the ACM Turing Celebration Conference - China on - ACM TURC '19*. New York, New York, USA: ACM Press, 2019.

[12] P. Laperdrix, G. Avoine, B. Baudry, and N. Nikiforakis, "Morellian analysis for browsers: Making web authentication stronger with canvas fingerprinting," in *Detection of Intrusions and Malware, and Vulnerability Assessment*. Cham: Springer International Publishing, 2019, pp. 43–66.

[13] A. Vaddeti, D. Vidiyala, V. Puritipati, R. B. Ponnuru, J. S. Shin, and G. R. Alavalapati, "Graphical passwords: Behind the attainment of goals," *Secur. Priv.*, vol. 3, no. 6, 2020.

[14] P. Ackerman, “Impediments to adoption of two-factor authentication by home end-users,” *SANS Institute InfoSec Reading Room*, 2014.

[15] J. Colnago, S. Devlin, M. Oates, C. Swoopes, L. Bauer, L. Cranor, and N. Christin, ““it’s not actually that horrible”: Exploring adoption of Two-Factor authentication at a university,” in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2018.

[16] Duo Labs Report, “State of the auth,” https://duo.com/assets/ebooks/state-of-the-auth.pdf, accessed: 2022-2-12.

[17] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, “A design science research methodology for information systems research,” *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, 2007.

[18] B. Kitchenham, “Procedures for performing systematic reviews,” http://www.inf.ufsc.br/~aldo.vw/kitchenham.pdf, 2004, accessed: 2022-3-14.

[19] C. P. Pfleeger, S. L. Pfleeger, and J. Margulies, *Security in Computing*, 5th ed. Philadelphia, PA: Prentice Hall, 2015.

[20] D. Dasgupta, A. Roy, and A. Nag, “Toward the design of adaptive selection strategies for multi-factor authentication,” *Comput. Secur.*, vol. 63, pp. 85–116, 2016.

[21] J. Campbell and K. Bryant, “Password composition and security: An exploratory study of user practice,” https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1199&context=acis2004, 2004, accessed: 2022-4-29.

[22] I. Velásquez, A. Caro, and A. Rodríguez, “Authentication schemes and methods: A systematic literature review,” *Inf. Softw. Technol.*, vol. 94, pp. 30–37, 2018.

[23] M. H. Barkadehi, M. Nilashi, O. Ibrahim, A. Zakeri Fardi, and S. Samad, “Authentication systems: A literature review and classification,” *Telemat. inform.*, vol. 35, no. 5, pp. 1491–1511, 2018.

[24] S. Choi and D. Zage, “Addressing insider threat using “where you are” as fourth factor authentication,” in *2012 IEEE International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2012, pp. 147–153.

[25] E. Bursztein, “The bleak picture of two-factor authentication adoption in the wild,” https://elie.net/blog/security/the-bleak-picture-of-two-factor-authentication-adoption-in-the-wild/, Dec. 2018, accessed: 2022-5-1.

[26] S. Almuairfi, P. Veeraraghavan, and N. Chilamkurti, “A novel image-based implicit password authentication system (IPAS) for mobile and non-mobile devices,” *Math. Comput. Model.*, vol. 58, no. 1-2, pp. 108–116, 2013.

[27] K. Mowery and H. Shacham, “Pixel perfect: Fingerprinting canvas in HTML5,” https://hovav.net/ucsd/dist/canvas.pdf, accessed: 2022-2-12.

[28] S. Fulton and J. Fulton, *HTML5 Canvas: Native interactivity and animation for the web*, 1st ed. Sebastopol, CA: O’Reilly Media, 2011.

[29] P. Laperdrix, W. Rudametkin, and B. Baudry, “Beauty and the beast: Diverting modern web browsers to build unique browser fingerprints,” in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 878–894.

[30] P. Laperdrix, N. Bielova, B. Baudry, and G. Avoine, "Browser fingerprinting: A survey," *ACM trans. web*, vol. 14, no. 2, pp. 1–33, 2020.

[31] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz, "The web never forgets: Persistent tracking mechanisms in the wild," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security - CCS '14*. New York, New York, USA: ACM Press, 2014.

[32] N. M. Al-Fannah, W. Li, and C. J. Mitchell, "Beyond cookie monster amnesia: Real world persistent online tracking," in *Developments in Language Theory*. Cham: Springer International Publishing, 2018, pp. 481–501.

[33] F. Rochet, K. Efthymiadis, F. Koeune, and O. Pereira, "SWAT: Seamless web authentication technology," in *The World Wide Web Conference on - WWW '19*. New York, New York, USA: ACM Press, 2019.

[34] J. Fong and R. Poet, "Creating graphical passwords on a mobile phone: Graphical passwords on a mobile," in *13th International Conference on Security of Information and Networks*. New York, NY, USA: ACM, 2020.

[35] A. P. Rachna Dhamija, "Déjà vu: A user study using images for authentication," https://www.usenix.org/legacy/events/sec2000/full_papers/dhamija/dhamija_html/usenix.html, Jun. 2000, accessed: 2022-4-29.

[36] D. Fraunholz, S. D. Anton, C. Lipps, D. Reti, D. Krohmer, F. Pohl, M. Tammen, and H. D. Schotten, "Demystifying deception technology:a survey," *arXiv [cs.CR]*, 2018.

[37] K. Adel, "honeybits: A PoC tool designed to enhance the effectiveness of your traps by spreading breadcrumbs & honeytokens across your systems to lure the attacker toward your honeypots," https://github.com/0x4D31/honeybits.

[38] "Npm's express package homepage," https://www.npmjs.com/package/express.

[39] "Npm's nodemailer package homepage," https://www.npmjs.com/package/nodemailer.

[40] "Npm's fingerprintjs package homepage," https://www.npmjs.com/package/@fingerprintjs/fingerprintjs.

[41] "Npm's express package homepage," https://www.npmjs.com/package/bcrypt.

[42] Google, "New research: How effective is basic account hygiene at preventing hijacking," https://security.googleblog.com/2019/05/new-research-how-effective-is-basic.html, accessed: 2022-5-2.

[43] S. Khandelwal, "Real-world SS7 attack — hackers are stealing money from bank accounts," https://thehackernews.com/2017/05/ss7-vulnerability-bank-hacking.html, May 2017, accessed: 2022-5-2.

[44] "A survey on multi-factor authentication for online banking in the wild," *Computers & Security*, 2020, accessed: 2022-5-2.

[45] Z. Chen, S. Guo, R. Duan, and S. Wang, "Security analysis on mutual authentication against man-in-the-middle attack," in *2009 First International Conference on Information Science and Engineering*. IEEE, 2009, pp. 1855–1858.

[46] K. Skračić, P. Pale, and B. Jeren, “Knowledge based authentication requirements,” in *2013 36th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2013, pp. 1116–1120, accessed: 2022-5-2.

[47] “Draft nist special publication 800-63-3 digital identity guidelines,” https://csrc.nist.gov/csrc/media/publications/sp/800-63/3/draft/documents/sp800-63-3-draft-revised.pdf, accessed: 2022-5-2.

[48] “Mfa is a necessary chore! exploring user mental models of multi-factor authentication technologies,” https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1649&context=hicss-53, 2020, accessed: 2022-5-12.

[49] V. Taneski, M. Heričko, and B. Brumen, “Systematic overview of password security problems,” http://epa.niif.hu/02400/02461/00088/pdf/EPA02461_acta_polytechnica_2019_03_143-165.pdf, accessed: 2022-5-1.

[50] S. Moshfeghian and Y. S. Ryu, “A passport to password best practices,” *Ergon. Des.*, vol. 20, no. 2, pp. 23–29, 2012.

[51] A. Singer, W. Anderson, and R. Farrow, “Rethinking password policies,” https://www.usenix.org/sites/default/files/rethinking_password_policies_unabridged.pdf, 2013, accessed: 2022-5-1.

[52] “The tor project,” https://www.torproject.org/, accessed: 2022-5-12.

[53] “Google authenticator,” https://apps.apple.com/us/app/google-authenticator/id388497605, accessed: 2022-5-10.

A Interview Protocol

This document is the guide we follow during end-users' testing interviews.

Before the interview:

- ☐ Prepare the laptop where the user will test the prototype (run the program and open the browser to the login page).
- ☐ Have the test instruction near the laptop.
- ☐ Prepare the consent/ethical form.
- ☐ Make sure that the recorder is working properly.
- ☐ Setup another computer for note taking.

During the interview:

- ☐ Explain the ethical parts. + Sign consent form
- ☐ Give a short introduction to the project.
- ☐ Let the participant use the instruction paper and carry the test.
- ☐ After the test is done. Ask the participant the interview questions.

Interview questions:

1. Do you study, work, or something else?
   a. What is your study/work domain?
2. Have you ever (Or are you currently) use any multi-factor authentication MFA solution for work, studies, or personal use?
   a. If you have (Or currently) use an MFA, can you tell us the authentication factors used by that solution?
3. If answered **yes** to **2** ask the following:
   a. On a scale of 1 to 5 how satisfied are you with the solution you have used before.
   b. Can you tell us one thing you would like to change about the solution you have used before
4. Questions regarding our implementation:
   a. On a scale of 1 to 5 how satisfied are you with our solution?
   b. Would you be willing to use this solution in your daily life?
   c. This implementation uses your email for unrecognised devices. Does this make you more or less likely to adopt this solution?
   d. Can you tell us one thing you would like to change about our solution?
5. Do you have any other ideas/concepts that you would like to tell us about this system that have not been brought up in previous questions?

After the interview:

- ☐ Transcribe the recordings.
- ☐ Register the answers in a spreadsheet

B Consent Form

Consent form to participate in:
Multi-factor Authentication Mechanism Based on Browser Fingerprinting and Graphical HoneyTokens

The focus of this project is to investigate the feasibility of a new authentication mechanism, with a focus on user perspective. This proposed mechanism will use browser fingerprinting technology, and graphical honey tokens. This will be evaluated by conducting interviews with test users, and security experts, as well as ensuring feasibility through a requirements checklist.

By signing this consent form you agree to your personal data being processed within the framework of the thesis/study described above. You can retract your consent at any time by contacting one of the contact persons listed below. After that point your personal data will no longer be stored or processed unless required by law.

The personal data which will be collected from you is a recording of your voice during the interview. Your personal data will be processed until it is transcribed (one week after the interview) after which they will be erased.

You always have the right find out which information about you has been collected, or comment on the process or the information that has been collected by contacting one of the contact persons listed below or by contacting the university's data protection officer on dataskyddsombud@lnu.se. Any complaints that cannot be resolved with Linnaeus University may be submitted to the Swedish Data Protection Authority.

…………………………………	…………………………………
Signature	Place and date

…………………………………
Name

Contact information

Student's name: Dillon Jonsson
Student's email address: dj222jm@student.lnu.se

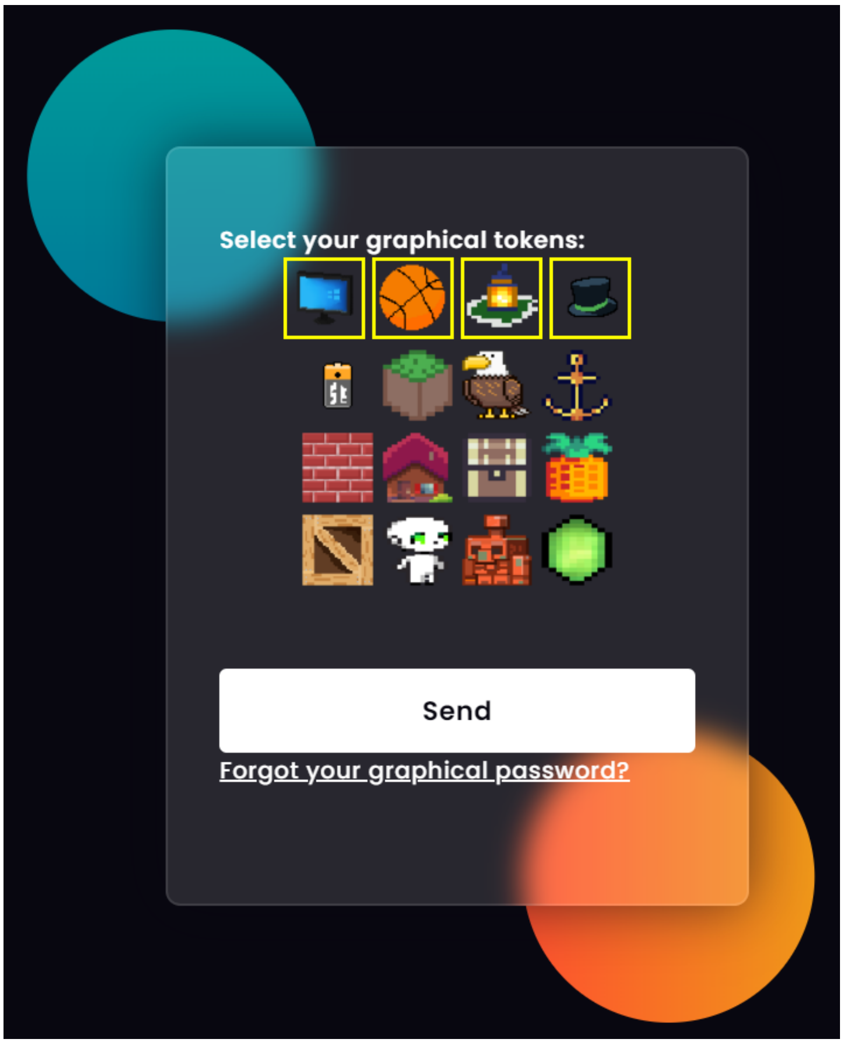
Student's name: Amin Marteni
Student's email address: am223wa@student.lnu.se

Supervisor's name: Ola Flygt
Supervisor's email address: ola.flygt@lnu.se

C Prototype test instructions

Instructions for Testing MFA Authentication System
For the thesis:
Multi-factor Authentication Mechanism Based on Browser Fingerprinting and Graphical HoneyTokens

Instructions:
1. Use the Email: test@snowbeez.com and the Password: 1234 to start the login process
2. Click the icons in the top row from left to right, in order.
   So that it looks like this:


Select your graphical tokens:
Send
Forgot your graphical password?

3. Click Send
4. Congratulations! You have logged in

D Ethical self-review

Etikkommittén Sydost
ett samarbete mellan

Linnéuniversitetet · REGION KRONOBERG · REGION BLEKINGE · Region Kalmar län · BLEKINGE TEKNISKA HÖGSKOLA · BTH

<u>Ethical self-review</u>

The following questions should be answered by the applicant and approved by supervisors.

		Yes	No
1	Intends the study to treat sensitive personal data according to the Swedish Authority for Privacy Protection (IMY) as: political opinions, ethnical origin, religious belief, sexual orientation health etc.		**x**
2	Does the study involve a physical intervention on the research subjects (even that which is not different from the routines but which is part of the research)?		**x**
3	Is the purpose of the investigation to physically or psychologically affect research subjects (e.g. treatment of obesity) or does it pose a clear risk to affect?		**x**
4	Is there use of biological material that can be traced to a living or deceased person (e.g., blood samples)?		**x**
5	Can free will/ voluntariness be questioned (e.g. vulnerable groups such as children, people with mental disability as well as anyone in obvious dependence such as patients or students who are directly dependent on the investigator)?		**x**
6	Is there an aim to publish scientifically such as at conferences or in a scientific journal after the study is conducted?		**X**
7	Will personal data registers be established (where data can be linked to an individual) and reported to the registry responsible person (GDPR).		**x**
8	The purpose and the method is well balanced regarding benefit-risk and adapted to the level of study.	**x**	
9	In the written information, the project is described so that participants understand its purpose and structure (including what is required of the individual, such as visits, project duration, etc.) and so that any details that may influence the decision on participation is clear. Minors are generally to have parental consent (e.g. questionnaires in school classes).	**x**	
10	Participation in the project is voluntary and this is evident in the written information to patient or research subject. It is also clear that the participants at any time and without giving any reason can cancel the trial, without affecting the person's care or treatment or, for students, grades, etc.	**x**	
11	There are resources for implementing the project and responsible for the study are named (supervisor and student).	**x**	