Combating Shoulder-Surfing:
A hidden button gesture based scheme

DISSERTAÇÃO DE MESTRADO

Mário Amilcar Freitas Rodrigues
MESTRADO EM ENGENHARIA INFORMÁTICA

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

Novembro | 2010

1 UMa
4
D Com
V +CD-R
.1

COMBATING SHOULDER-SURFING A HIDDEN BUTTON GESTURE BASED SCHEME

MASTER'S THESIS

Mário Amílcar Freitas Rodrigues
Student No. 2012203

Universidade da Madeira
Masters in Informatics Engineering
amilcarrodrigues[at]gmail[dot]com

Supervisor: Prof. Vassilis Kostakos

ABSTRACT

This project describes an authentication technique that is shoulder-surfing resistant. Shoulder surfing is an attack in which an attacker can get access to private information by observing the user's interaction with a terminal, or by using recording tools to record the user interaction and study the obtained data, with the objective of obtaining unauthorized access to a target user's personal information. The technique described here relies on gestural analysis coupled with a secondary channel of authentication that uses button pressing. The thesis presents and evaluates multiple alternative algorithms for gesture analysis, and furthermore assesses the effectiveness of the technique.

AUTHOR KEYWORDS

Shoulder-surfing, peeping attack, security, usability, authentication, cognometrics, biometrics, memometrics, token authenticators, multi-factor authentication.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 From watchwords to retina scans

Since long there has existed a need to protect information. In old times a type of authentication called the watchword was used as a way of securing the communication of orders between soldiers, to assure that the information would not be spread to an enemy, and verify that all the guards were who they were. Back then it was already necessary to transmit these watchwords in extremely secure ways that would avoid external access to it [1].

Nowadays multiple forms of authentication have come to exist, some pay more attention to security by focusing on techniques that increase the complexity or attempt to hide completely the authentication token from the exterior, others are not so secure but became more commonly used due mostly to their ease of use and memorability. Some even use unique parts of the human body as an authenticator like a retina scan system. But yet most of these popularly known authentication systems can still be breached, be it through observation or replication. Even the most secure of authentication systems can be breached if there is a way to observe the user interaction with the secured system and a way to replicate the authentication key.

Be it a complex video and audio recording system or to the simple act of looking over the shoulder of another person while entering a password on a computer, these acts have come to be known as shoulder-surfing attacks. This thesis focuses on this aspect of authentication security, and tries to find a plausible solution so that it may still feel easy to use and memorable for the user, while making it hard for an attacker to observe and replicate the interaction.

1.2 What is Shoulder-surfing

A shoulder-surfing attack consists of a deliberate attempt to gain knowledge of protected information through observation [21]. This kind of attack is also commonly known as a peeping attack [5]. The most common example of shoulder-surfing would be a person looking over the shoulder of a second person that is entering a password on a computer. This kind of unauthorized disclosure of information can lead to serious consequences, information misuse, and consequently identity theft or even theft of personal possessions. A shoulder-surfing attack is more common to happen during user interactions that involve user authentication. Even though the most common shoulder-surfing attacks happen when the attacker is close to the victim observing his actions during an authentication, it is not a necessary circumstance for it to be considered a shoulder-surfing attack. More complex means can also be used which are still considered shoulder-surfing: the attackers can use video cameras, electronic sensors and other tools, without the need to be present during the shoulder-surfing attack. Such incidents have been happening lately and have become a common threat [27] [28]. The main cause of this kind of attacks is a poorly designed user interface because it forces users to input their secret directly, thereby enabling an attacker to identify a secret visually.

1.3 Combating Shoulder-surfing with hidden gestures

To combat shoulder-surfing attacks during authentication it is necessary to find solutions that help protect the user from being observed during his interaction with the machine. Recently has appeared projects proposing the use of gestures as authenticators, by exploring the user's kinesthetic memory to memorize gesture passwords [24]. It has also been suggested that this kind of authentication can be used as a biometric, knowing that no two persons would perform the same gesture in the same way [20]. This project proposes to verify the efficiency of such gesture authentication methods and explore ways to hide gesture passwords, from any kind of possible shoulder-surfing attacks. In this project it is proposed a way to hide and compare gesture passwords, in which a set of gestures will act as an authenticator that is only read when a certain hidden button is pressed, making it extremely hard for any successful shoulder-surfing attack to occur. This concept can be even converted in various types of input schemes that would require a gesture as a password, be it obtained from a coordinate system, or an accelerometer.

1.4 Organization

This document is organized as follows: Section 2 will explore existing information about shoulder-surfing and authentication systems, and displays some of the current projects created to combat-shoulder-surfing with a personal evaluation of which, ultimately comparing them. Section 3 will describe the creation and development of the Hidden Button Gesture system, an application to test and evaluate the security of any gesture input obtained through accelerometer. Section 4 will describe the tests created using the application and results obtained development and evaluation. Finally Section V will describe the potential of hidden button gesture password and future options.

2. LITERATURE REVIEW

2.1. Shoulder-Surfing

The effectiveness of a shoulder-surfing attack is highly dependent on the adversary's ability to record the victim's interaction with the system.

During a shoulder-surfing attack the attacker stands near the target user and observes his actions to attempt steal his/her secret. Some measures are useful against this attack such as a privacy filter, which is an effective measure to some degree. Yet other forms of shoulder-surfing have emerged and changed the threat model, as an attacker can also use a video camera, or other tools to capture an authentication action at a safer distance, and record it for later use. This attack method has been used in actual incidents around the world in ATM scams [5]. We can distinguish between these two types of shoulder-surfing attacks as cognitive shoulder-surfing and recording-based shoulder-surfing.

Cognitive shoulder-surfing

The attacker stands behind the victim and looks over him/her to obtain passwords, PINs or other sensitive information. Despite its effectiveness against normal PIN entry schemes, this type of shoulder-surfing attacks is relatively easy to defend against since shoulder-surfers are limited by their cognitive capabilities [3]. This can also be done in relative distance with usage of spying devices such as binoculars.

Recording based shoulder-surfing

With the help of advanced technologies, such as concealed miniature cameras and video mobile phones, the adversaries' capabilities of observing and recording the login process have improved significantly [27]. This class of shoulder-surfing attacks has become a serious threat to many security applications that rely on the combination of magnetic stripe cards and PINs to perform the authentication [3]. Even though it is very unlikely that more than two authentication processes will be done by the same user on the same device within a reasonable time frame, with this method, just one observation might be enough to get all the info.

Entities involved in shoulder-surfing attacks

A shoulder-surfing attack always includes two types of entities, the shoulder-surfer who performs the attack, and the target user, who holds the private information.

Shoulder-surfer:

- Needs to accurately reconstruct every part of the input.
- Can be probably several feet away from the terminal, during cognitive shoulder-surfing.
- Might not even be present, in the case of recording based shoulder-surfing.

Target user:

- Knows the secret needed to access the information, and is only looking for errors.
- Is close to the terminal where the data is being entered and can view what is on the screen with relatively low contrast.
- The target user is always present.

These differences should be exploited when producing an interface that combats shoulder-surfing attacks [12].

The shoulder surfing threat is an issue that is still not taken seriously enough by executives and workers which hold important confidential data on their devices. 65% of the businesses in the UK do not offer a comprehensive security policy that combats the issue of shoulder-surfing. Important executives appear to not worry about reviewing confidential information on a laptop in public places, leaving them at the mercy of complete strangers nearby. According to research done by 3M 55% of management professionals questioned admitted to working on their laptops while on public transport or in shared work places at least once a week [17].

Security breaches resulting from lost or stolen laptops can result in serious penalties, including heavy fines or permanent bans from obtaining and holding customer details in the future. This demonstrates the severity of such laxity in the eyes of regulatory bodies. Ineffective security policy enforcement can have a detrimental impact not only on the organization but also on public confidence in personal data protection and the individuals' right to privacy [17].

2.2. Authentication

As shoulder-surfing attacks are mostly focused on obtaining authentication information, it is necessary to know what is authentication and what different systems exist.

Three distinct steps can be identified during the utilization of a secured system: identification (asking the user to identify himself), authentication (the action of providing the user's evidence of his identity) and authorization (allowing an authenticated person to access a set of actions, according to his identity) [13].

During an authentication process the system determines whether the person being authenticated has possession of the pre-agreed secret. If the user proves knowledge of the secret, the system will authorize the person to access the system. In the case of biometrics, the system records a digital representation of some aspect of a person's physiology or behavior at enrollment, and this is confirmed at authentication time [13].

Authentication systems

Authentication systems are usually categorized by the number of factors that they incorporate. The three factors most often taken into consideration are: 1) something the user knows (example: a password), 2) something the user holds (for example, an ID badge or a cryptographic key), 3) something the user is (example: a finger print or other biometrics) [2]. Recent works take in account a fourth factor 4) something the user recognizes (for example, identifying objects within a picture) [13]. Identifying authentication systems this way by relating them to what the user does is a good scheme but it is not completely correct. For instance, a password is not exactly known: it is memorized. Therefore, it can be forgotten, either in the short term or over a longer period. And a biometric is simply one feature of your appearance.

A good way of identifying the different authentication systems is by using the following categories [2][13]:

1) Knowledge-based or Memometrics ("something the user knows"): Characterized by secrecy or obscurity. This includes two types of passwords: random passwords in which the secret is known only by the user (PIN's, passwords, and passphrases), and cultural passwords which can be loosely defined as "secret from most people" (challenge-questions and cognitive passwords such as "mother's maiden name"). A security drawback of secrets is that each time it is shared for authentication it becomes less secret.
2) Recognition-based or Cognometrics ("something the user recognizes"): This involves two types of systems, recognition-based that require the user to select information within a set of distracters (example: recognizing a picture from a set of similar pictures), and position-based that require the user to identify target objects based on their location (example: recognizing the position of an object that was previously drawn on a picture).
3) Identity-based or Biometrics ("something the user is"): Characterized by uniqueness to a person. These can be distinguished as behavioral biometrics, based on an object's usage patterns, latencies, or signature dynamics, and physiological biometrics, that can be based on any recordable physiological characteristic of the user, such as fingerprints, voice record, and even body geometry. The main security defense is that they are difficult to copy or forge. But unlike the other authentication types, biometrics is obviously not secret, and it is difficult to keep a fingerprint or iris secret from a determined attacker. Note that some works [3] also consider the personal identity information, such as the information contained on an ID card or driver's license, as an Identity-based authenticator, although this kind of information is not usually used by itself as an authenticator.
4) Object-Based ("something the user holds"): Characterized by physical possession. This can be based off an object that the user holds, typically called as token. These can include typical keys but also, can be hardware or software devices that hold the secret information required to authenticate. This information is not required to be known by the user. A security drawback of a physical token is that, if lost, it enables its finder to enter the house. This is why many digital tokens combine another factor, an associated password, to protect a lost or stolen token. On the other hand, this also implies an advantage that if the token is lost, the owner sees evidence of this and can act accordingly.

Multi-factor authentication

Different types of authenticators can be combined to enhance security, this is called multifactor authentication. For security purposes, each authenticator result must be satisfied, typically a Boolean AND operation is performed for the authentication results of each factor so all results must be affirmative [2]. A common example of multifactor authentication is a bankcard, the combination of a bankcard with a password can be considered a two-factor authentication, and it is a better choice than a card alone because the card can be stolen.

Furkan Tari et al proposed a comparison of shoulder-surfing risks between alphanumeric and graphical passwords. This comparison explored the real and perceived vulnerability of four configurations of authentication systems (dictionary password input, non dictionary password input, graphical password input with mouse and the same graphical password input with keyboard). This study revealed a perceived higher level of vulnerability to shoulder-surfing by the use of the

graphical password with mouse, but a much lower level of vulnerability to shoulder-surfing was shown using that graphical password with keyboard input. Possibly due to the speed entry with the keyboard, and the need to look at the keyboard and screen at same time. Non-dictionary passwords also proved to be vulnerable to shoulder-surfing, more than dictionary passwords or keyboard input graphical password, verifying that a password that is more resistant to a dictionary attack is likely more vulnerable to shoulder-surfing [16].

2.3 Gesture Passwords

Accelerometers have been increasingly integrated into mobile phones, and other devices. Such devices with built-in accelerometer can sense and acquire data from the user's movement having become a new form of user input.

Ming Ki Chong and Gary Madsen [23] explored ways to use this kind of input as authentication, by using an accelerometer to detect directional movements as gesture inputs. Following the idea that instead of using text-characters or images, a password can be made up of multiple gestures: through practice and repetition, the password movements can gradually consolidate into the user's memory. Although they had negative results when comparing them to pin input memorability, their work also showed a path for possible improvements.

Fuminori Okumura et al [20] also explored the use of gestures as authentication, but in this case as a new biometric authentication method based on the differences between various users performing the same arm sweep gesture, through the usage of algorithms that help compare the consistency between the gestures of two users.

2.4 Shoulder-surfing solutions

PIN Entry Scheme resistant to recording-based Shoulder-Surfing

This project proposes a scheme which is resistant against shoulder-surfing attacks conducted by shoulder-surfers with normal cognitive capabilities. This consists of a new PIN entry scheme in which the PIN is actually a sequence of 4 locations, instead of the typical sequence of 4 digits. Additionally, this scheme offers a relatively good level of security when the shoulder-surfer can record the entire login procedure for one or two times with a video device [3].

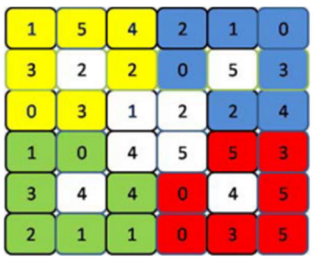


Figure 1. An example of the new PIN entry scheme, using colored patterns in displayed tables in order to improve its usability

The idea is to use the contrast between different colors to enhance the identification and memorization of the positions in a table. The secrets in this are: PIN code, and position.

Click to Zoom-inside (CTZ)

This was suggested as a new graphical password authentication mechanism. In CTZ, users have to click on one point of some specific regions of a large theme image that consists of several objects, shown by dotted lines. The user has to click 6 times to create a password. Each click in the specific region results the next image which is the zoom image of the specific region and the user has to click again on some specific areas shown dotted in this given image which results the next image. The sequence of clicked images creates a password. The user has to click the specific areas in the correct sequence. A wrong click leads down an incorrect path, with an explicit indication of authentication failure only after the final click. Users can click the regions only to the extent that their click-point dictates the next image. With this system genuine users would get immediate feedback about an error when trying to log in. When they see an incorrect image, they know that the latest click-point was incorrect and can immediately cancel this attempt and try again from the beginning. The secret here would be the sequence of images [4].

fakePointer

Using fakePointer, a legitimate user does not leak a secret even if an attacker captures a video record of an authentication interaction. The fakePointer scheme introduces two features to realize security against the threat. The first feature is a double-layered user interface for a secret input. This user interface makes it difficult for attackers to identify a secret visually. The second feature is that fakePointer makes use of two secrets. One secret is a fixed secret, which is the same as that of a traditional authentication. The other secret is a disposable one-time secret named "answer indicator" [5].

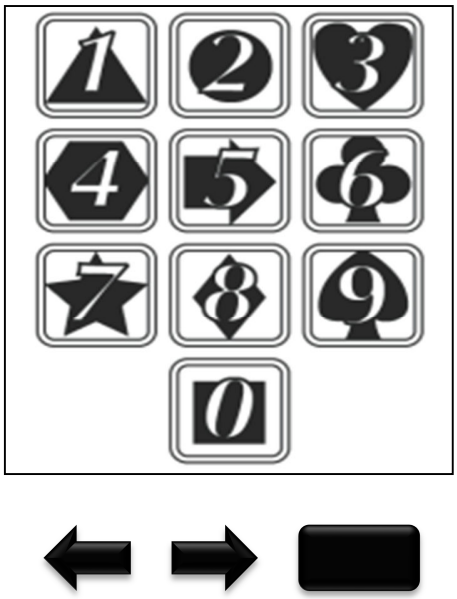


Figure 2. An example of the fakePointer interface with randomly sorted pictures behind the numbers.

The secrets in this suggestion are PIN code and a 10 picture code.

Password image and Key image

An example of a multi-factor graphical solution was Password image and Key image. This authentication process comes in the form of two images. The first image is the password image which is sent to the user's terminal as a challenge for password input. This password image can be plain or encrypted. The password image is encrypted only if it contains some information about click points. In this case the password image and key image are identical [6].

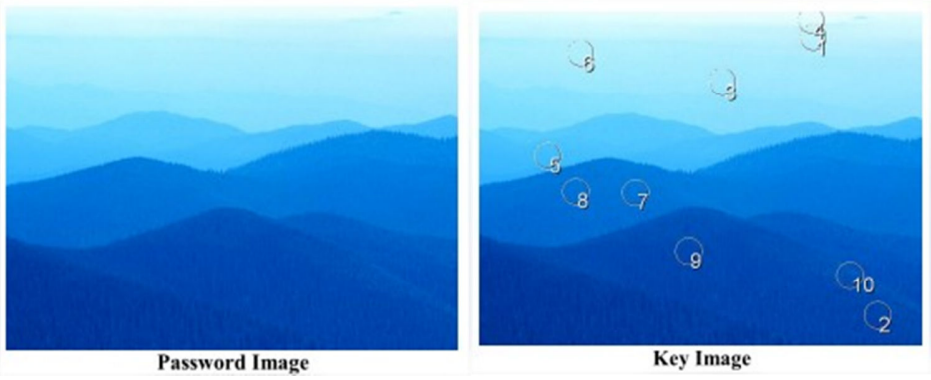


Figure 3. Example of a password image and a possible key image used

The Key image is a copy of the password image which is always encrypted and signed by the challenger and can be validated and decrypted on the user's handheld device. The key image contains enough information to show the click spots to the owner of handheld. There are some clickable areas in the password image. The user's password is the click points and their order. The click points are clickable areas in the password image which a user can identify them by looking at the key image. The click points and their order are either highlighted in the key image or the user can determine them with some prior knowledge.

YAGP

Yet another graphical password (YAGP) is a position-free scheme, in which the user can draw his graphical password anywhere on the canvas, which makes shoulder surfing a difficult task. The stroke sequence cannot be reflected by the graph in YAGP, and authentication process sees it as a critical checking factor. This property ensures that the peeper still cannot sign in even if he glimpses the images, because he could not recall the correct stroke sequence set by the legal user. YAGP also takes into account the drawing trends, which means it records the user drawing style to a certain extent [7].

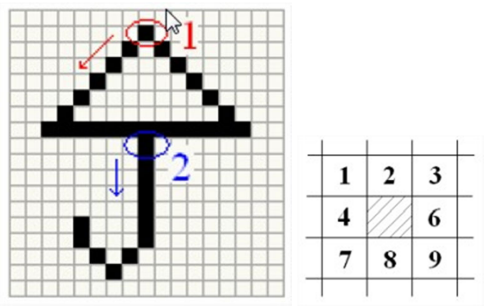


Figure 4. Example of drawing an umbrella as a YAGP password

The main drawback of YAGP is that it's hard to redraw the password precisely. The legal user cannot always be assured to login successfully because the gaps between user drawings are uncertain while the similarity threshold value is fixed [7].

Keystroke patterns

Keystroke identification examines the timing pattern that is produced as a typist presses the different keys on the keyboard. From this typing pattern, there are several unique features that can be extracted. One such characteristic (feature) is the key hold-down time, which is the amount of time that a particular key is held down. Another feature is the keystroke latency, which is the time between pressing two consecutive keys; we shall refer to this feature as the down–down keystroke latency [8].

EyePassword (Gaze-based password entry)

A system that mitigates the issues of shoulder surfing via a novel approach to user input. With EyePassword, a user enters sensitive input (password, PIN, etc.) by selecting from an on-screen keyboard using only the orientation of their pupils (i.e. the position of their gaze on screen), making eavesdropping by a malicious observer largely impractical [9].

Spy-Resistant Keyboard

This keyboard randomizes the spatial location of all characters as each password character is entered. This approach introduces indirection by utilizing an auxiliary mapping that allows typists to focus their attention on a particular part of the keyboard, while observers have to pay attention to and memorize the entire keyboard [10].

In order to select a character on the Spy-Resistant Keyboard, the typist first locates the tile that contains the character to be typed. They remember the mapping by noting the location of this tile. Next, the typist clicks on one of the interactors at the bottom of the keyboard to cycle through shift states and move the red underline to the desired character. Clicking on the Interactor moves the underline to the next character on each tile. Finally, the typist drags the Interactor towards the Character Tile on which the desired character resides.

A user's mobile device displays a gesture sequence on the public terminal, and the user authenticates to his device by shaking it in the required back-and-forth pattern. This scheme requires no secret knowledge for authentication [10].

Convex Hull Click Scheme

The Convex Hull Click Scheme (CHC), is a graphical password scheme that guards against shoulder-surfing attacks by human observation, video recording, or electronic capture [11].

In a challenge the user must recognize some minimum number of his or her password icons, or "pass-icons," out of a much larger number of randomly arranged icons. The user responds to the challenge by clicking within the convex hull of the pass-icons.

Figure 5. Example of Convex Hull scheme with 3 pass icons

The secret in this authentication scheme is the complete set of icons selected by the user to be his pass-icons, of these only a few randomly chosen icons are shown during each authentication process, making each authentication process diferent from the previous.

Other Solutions

Toni Perkovic et al, sugest 3 diferent simple PIN-entry methods, designed for the partially observable attacker model. All of them consisting of challenge-response protocols that allow a user to login securely in the presence of an adversary who can observe user input. The first method "Mod10" consists of the user performing a simple mathematical operation, while in the other two methods STL and Mod10-table, the user performs a simple table lookup. All these methods show a reasonably low login time and minimal error rate, with Mod10 being the fastest but also the one with higher error rates, and also found to be the most suitable for utilization by younger users [18].

Pavel Lazhnikov presented a biometric system based on user's identification through handwriting dynamics. This consisted of obtaining data of handwriting dynamics through two functions of time of changing the position of a light pen on a tablet plane x(t) and y(t), and a function of pressure changes of the touch-sensitive light pen tip on the tablet plane. The biometric system then compares the processed measurements of a user's signature with a particular collection of templates that had been input earlier, and makes a decision about the closest match [25].

2.4.1 Comparison

In [15] it was concluded that password security involves striking a balance between having enough rules to maintain good security and not having so many that users will take evasive action that compromises security. This not only applies to passwords, but to any authentication scheme.

In order to compare the current state-of-the-art solutions to combat shoulder-surfing, it is necessary to look at various factors that can be divided between usability factors and security factors.

Usability Factors

Regarding usability the following five concepts, mentioned in Usability Engineering by Jakob Nielsen [14], were taken into consideration:

- Learnability: how easy to learn is the system so that the user can quickly start working with it.
- Efficiency: how efficient to use is the system so that once learned, the system can be used with a high level of productivity.
- Memorability: how well does the system allow a user to remember how to use it, so that the user is capable to return to the system after a long period of time, and still be able to use it without having to learn everything about it again.
- Errors: How well does the system prevent errors and how does it recover from them if they occur.
- Satisfaction: How pleasant it is to use a system and how satisfied a user is with it. (Note that this factor was not used in the comparison due to lack of data on it)

Security Factors

Security systems and methods are often described as strong or weak. When used in relative terms, the meanings are clear. A door with a lock offers stronger security than one with no lock. In order to measure absolute strength and weakness of security systems the following is a possible way: A strong system is one in which the cost of attack is greater than the potential gain to the attacker. Conversely, a weak system is one where the cost of attack is less than the potential gain [2]. To evaluate the security of a system we must focus on which attacks can be done to it. In this comparison were considered the following factors directly related to shoulder-surfing attacks:

- Cognitive-based resistance: based on how easy is it to observe the user interaction.
- Recording-based resistance: based on how easy is it to record the user interaction with the system using any tool.
- Secret strength: based on how much data should be gathered during successful attacks, in order to obtain the secret information.
- Brute-force resistance: based on how long would it take to discover the authentication information by trying every possible combination. As it won't matter how well hidden is the authentication process, if the user can easily try all possible combinations.

Comparison Results

Based on the usability and security factors, a table was built in which each solution was evaluated using a score system. The scoring system chosen had a short range (from 1 to 5, with 1 being less secure or less usable and 5 being the most secure or usable) and was based on personal opinions with conclusions taken from what was read about each of the solutions. The results can be seen in Table 1 and Table 2.

These results allowed to obtain a graphical comparison, in which can be more easily verified, the relation between security and usability in the solutions that attempt to fight shoulder surfing, as it can be observed in Figure 6.

	Learnability	Efficiency	Memorability	Errors	TOTAL
New PIN entry scheme	3	3	3	4	13
Click to Zoom-Inside	3	4	4	4	15
fakePointer	3	2	4	4	13
Password Image & Key Image	4	3	3	3	13
YAGP	5	3	4	4	16
Keystroke Patterns	4	4	4	2	14
EyePassword	4	4	3	3	14
Spy Resistant Keyboard	3	2	4	3	12
Convex Hull	3	4	4	4	15
Typical PIN	4	5	5	5	19
Typical Text Password	5	5	4	5	19

Table 1. Usability factors evaluation

	Cognitive-based	Recording-based	Secret strength	Brute Force	TOTAL
New PIN entry scheme	3	1	3	5	12
Click to Zoom-Inside	3	1	2	4	10
FakePointer	4	2	5	5	16
Password Image & Key Image	4	1	3	5	13
YAGP	3	2	3	5	13
Keystroke Patterns	5	3	2	4	14
EyePassword	5	3	2	5	15
Spy Resistant Keyboard	4	4	3	4	15
Convex Hull	3	3	4	4	14
Typical PIN	1	1	1	1	4
Typical Text Password	1	1	1	2	5

Table 2. Security factors evaluation

From these comparison results it was noticeable that the most secure solutions were the ones based in multifactor authentication. The fakePointer was considered one of the most secure of these due to the high quantity of shoulder surf attacks that would be needed to identify the PIN even through recording based shoulder surfing, yet this solution was not very usable due to the different way than the usual to input, and the need to get the picture password separately. Meanwhile YAGP was the most usable of the new solutions as it was based in a person's ability to draw, though it failed a bit in security due to being possible by multiple observations of a recording based shoulder surfing, as the secret was totally observable during the interaction.

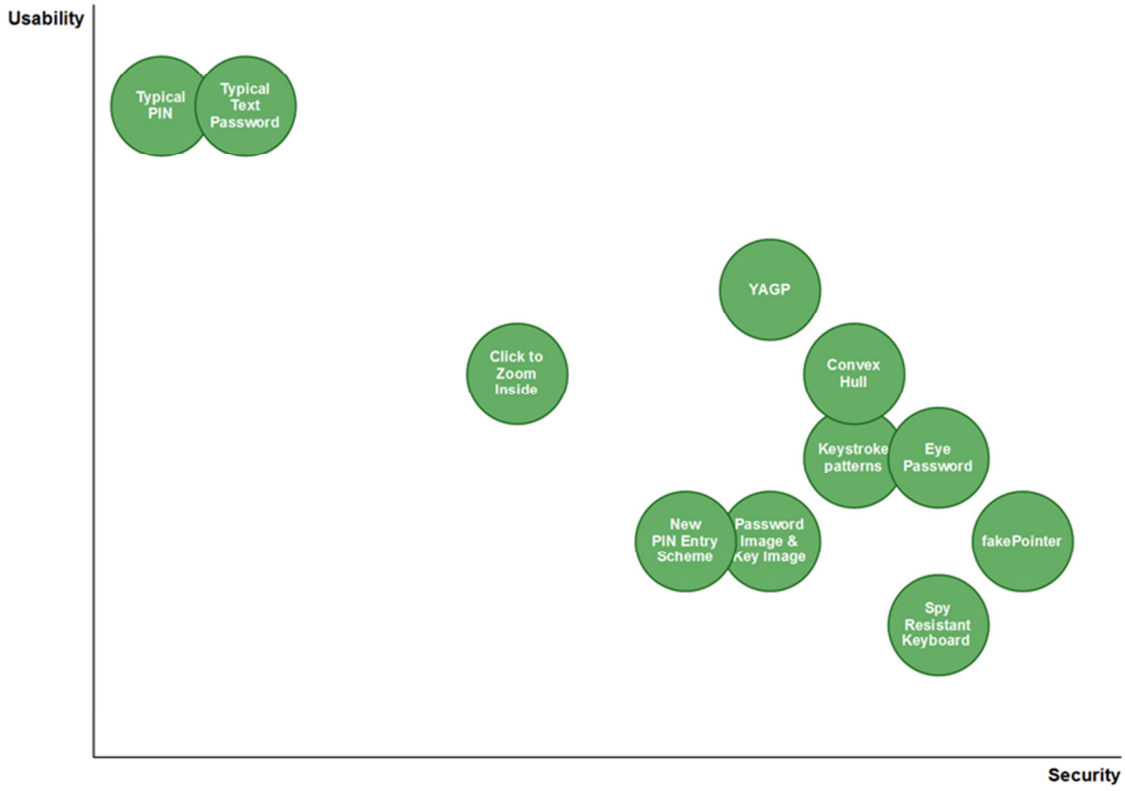


Figure 6. State-of-the-art Usability vs Security comparison

As it can be seen in Figure 6 a balance is necessary, between the most important usability factors found in works like YAGP and Click to Zoom-inside and the security factors found in fakePointer and the EyePassword.

3. SYSTEM OVERVIEW

3.1 Rationale

During the state-of-the-art comparison it was noticed that the most usable of the solutions were the ones closely related to drawing or movement, consisting of an interaction that involves user movement leading to an input in the form of multiple coordinates or directions. On the other hand the solutions most secure against shoulder-surfing managed to hide the data input thanks to complexity during interaction. It was also noticed that there is a lack of solutions that would hide the user's interaction with the system during a gesture based or graphical based input.

A new solution for would have to be an input scheme that could hide the whole input with the user from outside, without hiding it from the user, and without adding too much complexity that would provoke incorrect use.

3.2 Description and requirements

The concept of the Hidden Button Gesture input scheme is based in a gesture or a set of gestures that would act as an authenticator, but that would only be read when the hidden button would be pressed. So even if a user would do a long movement, the only parts of it that would count as an authenticator would be the movements done during button pressing. This concept could be even converted in various types input schemes that would require a gesture as a password, be it obtained from a coordinate system, or an accelerometer.

To test this concept it is necessary to create a system where gesture input data obtained from an accelerometer, can be processed into obtaining a comparison score between multiple gestures, in order to identify the degree of similarity between attempts by the same or different individuals. For this it would be necessary a device with an accelerometer to input gestures, preferably with small button on it for the button pressing.

For this prototype the following tools were used:

- Wii Remote™
- PC with Wii Remote compatible bluetooth stack
- Microsoft Visual C#
- WiimoteLib, open source Wii Remote library for C#

This authentication system will need to be able obtain and analyze the movement, and button pressing data, obtained from the Wiimote, in order to authenticate as the real password. Also taking into consideration is that humans rarely make the exact same movements; therefore a system using this scheme will heavily depend on a certain level of error tolerance.

3.2.1 Wii Remote

The Wii Remote is a one-handed remote controller, with motion sensing capabilities. The Wii Remote has the ability to sense acceleration along three axes through the use of an ADXL330 accelerometer [21]. The Wii Remote also features a PixArt optical sensor, allowing it to determine where the Wii Remote is pointing.

iMEMS® Accelerometer ADXL330

The ADXL330 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs, all on a single monolithic IC. The product measures acceleration with a minimum full-scale range of ±3 g. It can measure the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration. It uses a single structure for sensing the X, Y, and Z axes. As a result, the three axes sense directions are highly orthogonal with little cross axis sensitivity [22].

3.2.2 WiimoteLib for C#

When the Wiimote is paired with a PC, it will be identified as a HID-compliant device. Therefore, to connect to the device through a C# application, it is recommended using the HID and Device Management Win32 APIs.

WiimoteLib is a .NET managed library for using a Nintendo Wii Remote (Wiimote) and extension controllers from a .NET application. This library makes use of the P/Invoke signatures, which allows it to directly call methods of the Win32 API from the .NET framework. Most of the methods used in this application were defined by the P/Invoke Wiki resource.

3.3 Implementation of the Hidden Gesture Data Processing tool

A gesture data processing application is necessary to obtain, store and manage user gestures data, that will be used later to help compare algorithm efficiency. This tool acquires data from the Wiimote's accelerometer by storing the value of each of the accelerometer's axis (X,Y,Z) within a certain period of time.

Using the WiimoteLib library functions to obtain raw gesture data from the Wii remote, the application was planned acording to the following class plan.

The MainForm class would handle the user interface and calls to the system, handling the creation of new gestures, requests for gesture data processing, wii remote button calls, and returning results in xml or excel spreadsheets.

The GestureProcessing class will handle the functions related to gesture data processing, including gesture data filtering functions to optimize the data used in the gesture comparison, and the algorithms used to compare different gestures.

The GestureSet class will be used to create objects of the type GestureSet which may hold one or more gestures for the same user. The gesture class will be used to create Gesture objects each containing the multiple positions recorded by the Wii remote during the gesture input period.

The XSLSEditor class will handle the creation of excel spreadsheet tables needed to visualize multiple gesture comparison results, that will be later used to evaluate the success of the gesture comparison.

The XMLEditor will be used to store and load gesture data previously recorded by the system for future comparisons.

```
<MOTION>
  <POSITION>136,108,108</POSITION>
  <POSITION>138,104,104</POSITION>
  <POSITION>135,101,101</POSITION>
  <POSITION>134,100,100</POSITION>
  <POSITION>130,99,99</POSITION>
  <POSITION>128,98,98</POSITION>
  <POSITION>123,97,97</POSITION>
  <POSITION>119,96,96</POSITION>
</MOTION>
```

Figure 7: Sample gesture data, as it is stored in an XML file.

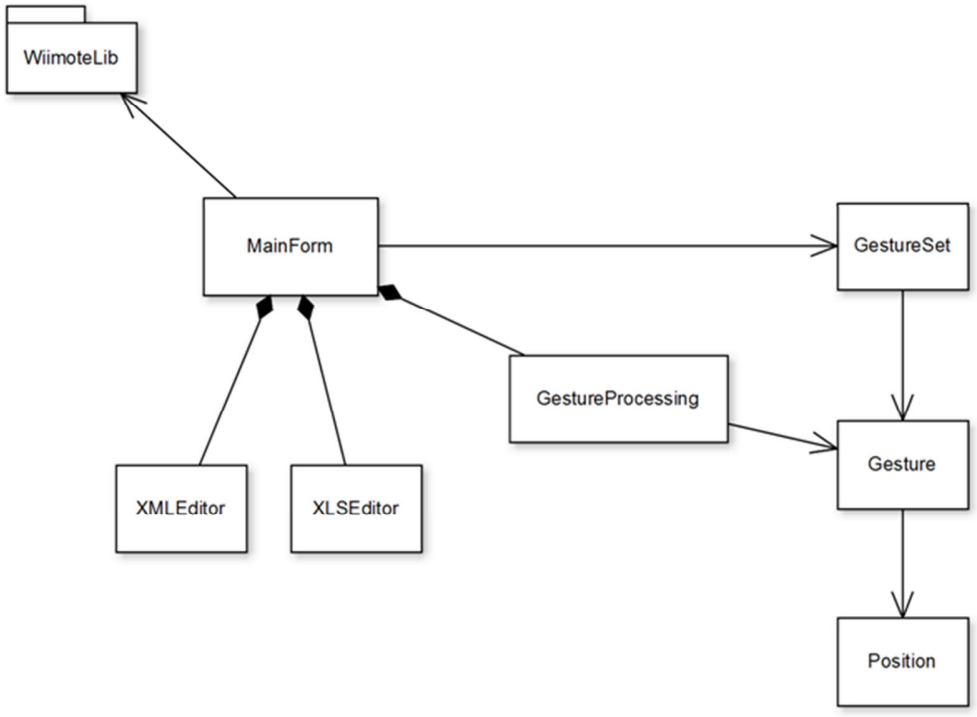


Figure 8: Basic UML Class Diagram showing the dependencies between the main classes used in the application

3.4 Gesture Data Recording and Filtering

The gesture data obtained through the gesture processing tool is always different every time a user makes a gesture input. Some variables can be used to help evaluate if two gestures are plausible for comparison or not.

Relative size: Even though the user should try to be as close as possible to his previous gesture it will always change size. As too large differences between 2 gesture sizes have a higher probability to return unreliable results, and those results even if positive should be immediately considered as a failed input. But it should also be considered that the bigger the gesture sizes the harder it will be to replicate and the range of error should increase accordingly. For these tests we decided to go with the rule that at a minimum, one gesture should never be bigger than the double of the gesture that it is being compared with.

Minimum size: gestures with too small size lasting under 1 second are too short to be evaluated with precision. Therefore a minimum size should also be implemented during the gesture input of any user.

Gesture creation verification: even though it is the same user, the gesture creation system should be able to identify if the user made a mistake, or is being too inconsistent during the gesture creation. In order to avoid bad gesture data. This might decrease the usability, possibly forcing a user to input the gesture more times than ideally, during the gesture creation, but it is a necessary factor to increase the precision of the system, indirectly increasing the memorability of the gesture password. An optimal score filter was created to filter every gesture comparison of similar size, to compare the most similar possible section.

3.5 Gesture Comparison Algorithms

Fuminori Okumura et al [20] proposed a method to authenticate a user through the use of acceleration signals obtained by an acceleration sensor embedded in cellular phones during arm sweep action. This method would verify the user's acceleration signals by using a DP-matching algorithm which can adapt fluctuations caused by different grip. They also tested this method against 2 other algorithms (error of squared Euclidean distance and error of angle). As these algorithms proved to be considerably successful they were also considered and used in this project to help process the data obtained in order to determine how successful the hidden gesture can be.

3.5.1 Error of Squared Euclidean Distance

The Euclidean distance between two points p and q is the length of the line segment between them. In Cartesian coordinates, if p = (p1, p2,..., pn) and q = (q1, q2,..., qn) are two points in Euclidean n-space, then the distance from p to q is given by the formula:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

The Squared Euclidean distance metric uses the same equation as the Euclidean distance metric, but does not take the square root. As a result, clustering with the Euclidean Squared distance metric is faster than clustering with the regular Euclidean distance.

$$e_{pq} = \sum_{i=1}^{n}(p_i - q_i)^2$$

In order to compare different gestures, if e_{pq} the difference between acceleration signals p and q, was less than predetermined threshold, we decide that those signals were from the trials of the same person.

3.5.2 Error of Angle

The error of angle is commonly used to evaluate the difference between acceleration signals.

The error of the angle between two points a and b is the angle formed by them with intersection on the origin. In Cartesian coordinates, if a = (a1, a2,..., an) and b = (b1, b2,..., bn) are two points in Euclidean n-space, then the distance from a to a is given by:

$$e_{ab} = \sum_i \omega_i = \sum_i \left| arccos\left(\frac{\langle a_i, b_i \rangle}{\|a_i\| \|b_i\|} \right) \right|$$

3.5.3 DP-matching

Dynamic programming is a method of solving complex problems by breaking them into simpler steps. Any algorithm, in which an optimization problem is solved by saving the optimal scores for the solution of every sub-problem instead of recalculating them, can be considered a dynamic programming matching (DP-Matching) algorithm.

A DP-Matching algorithm is used to find the least cost path in a grid. It works by first evaluating the least cost distance to reach every square in the grid and then tracing back the path which corresponds to the overall least cost distance.

A common DP match algorithm to search the least cost path in a grid usually consists of the following:

- Initialize an array g to hold the lowest cost path to each grid square.
- Set the values of the first element of the grid as g(0,0) and the values of the first row g(i,0) and first column g(0,j).
- For each row i and column j, calculate g(i,j) as the minimum of the costs of the three possible ways to get to (i,j).

The overall distance is 1/N * g(I,J) where I and J are the lengths of the input and stored patterns respectively. This algorithm involves one simple calculation per grid square and relies on the

observation that there are three ways to get to any grid square (horizontally, vertically or diagonally). In the end we have a measure of the cost of the best path but we don't know the path that gave rise to it. In most situations we're not really interested in the path but if we do want it can be reconstructed by keeping track at each grid square which of the three directions gave the minimum cost path. The path can then be traced back from the endpoint [26].

DP-Matching over error algorithm data

The DP-Matching algorithm used in this project is similar to the one used by Fuminori Okomura et al [20]. Although in this project DP-Matching will be tested with both the error of angle and Euclidean distance error, in order to verify the possible advantages or disadvantages of using this method with data already filtered and resized with an optimal score filter.

There are two parameters in this algorithm, an entrance penalty and a toll penalty. In this method the error, calculated through the use a gesture comparison algorithm for two positions i and j err(i,j), is used as an entrance penalty, and a constant value as toll penalty.

$$e_{ab} = score(I-1, J-1)$$

$$score(i,j) = min\begin{bmatrix} score(i-1, j-1) + err(i,j) \\ score(i, j-1) + err(i,j) + toll \\ score(i-1, j) + err(i,j) + toll \end{bmatrix}$$

$$score(0,0) = err(0,0)$$

$$score(0,j) = score(0, j-1) + err(0,j) + toll$$

$$score(i,0) = score(i-1,0) + err(i,0) + toll$$

$$0 \le i < I, 0 \le j < J$$

I and J are the number of frames in the acceleration signals a and b, toll is the toll penalty, score(i,j) is the grid with the lowest cost path to each grid square, err(i,j) is the value of the error of angle.

This kind of DP-Matching algorithm has proven to be considerably successful in some kinds of gesture authentication systems [20] and it has also been proven useful in signature authentication systems [19].

4. User Study and Discussion

Two user tests were organized in order to test the efficiency of the Hidden Button Gesture solution. The objective of the first was to verify the efficiency ratio of the algorithms, and the second test will analyze how effective against shoulder-surfing is the new Hidden Button Gesture solution.

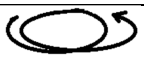
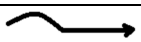
4.1. Algorithm efficiency test

4.1.1. Description

A few facts were considered in order to test the effectiveness of using hidden gestures as an authentication scheme:

- A gesture would only be recorded while the button was pressed.
- The gestures tested should be commonly known gestures that any person can perform.
- The test should not be gender-specific or age specific.
- Test it in an area with enough space to perform any gesture freely.

Gestures chosen for the algorithm efficiency test:

1	Punch	
2	Golf Swing	
3	Cowboy Lasso	
4	Whiplash	
5	Fencing	
6	X	
7	Wave	
8	Z	
9	O	

Each user was asked to perform each gesture 10 times, having a quick pause of about 3 seconds between each.

The motion data of each gesture was shown on the screen to confirm that the gesture was being detected correctly every time.

Each gesture was recorded as a set of coordinates that were obtained from the Wii Remote, and each group of gestures was kept in a XML file respective to the motion number.

4.1.2. Demographics

The users tested were 10 students and 1 teacher at the University of Madeira, and all of them had at least once previous experience with accelerometer equipment. Their ages ranged between 20 and 40 years old. The user considered as the "victim" was one of these students.

4.1.3. Collected data

Comparison Steps for each of the gestures:

1. Choice of the optimal gesture made by the victim, by running the comparison over the victim's 10 repetitions of the same gesture and choosing the one with the best relative scores.

2. Creation of a table in which all the repetitions of the same gesture by every user (victim and shoulder-surfers) are displayed.

3. Creation of threshold/algorithm success table in which is shown the success of the algorithm in this gesture for different chosen thresholds, showing true positive and false positive ratios, and failure ratio.

4. Creation of a failure rate/threshold table for each algorithm in which is shown the failure ratio on each gesture. (As the threshold for each comparison depends on a few factors, most importantly the gesture size, it was favorable the usage of multiple value ranges for each gesture).

The threshold values were chosen based on the average scores of each gesture with each algorithm used were different for each algorithm, as each algorithm shows numeric results in a different proportion.

The failure ratio is a calculated score based on the assumption that a very low amount of false positives is considered more important than a medium amount of false negatives. As a system with over 10% false positive ratio cannot be considered secure, while having a reasonable amount of false negatives is more acceptable.

Therefore the failure ratio formula used was: Failure ratio = False negative ratio * 0.2 + False positive ratio * 0.8

Example: a result of 5% false positives with 50% false negatives should be considered better than a result of 15% false positives and 25% false negatives.

Threshold	False Negatives	True Positives	True Negatives	False Positives	False Negative Ratio %	False Positive Ratio %	Failure Rate
51941	3	7	86	14	30	14	17,2
20000	9	1	100	0	90	0	18,0
30000	8	2	95	5	80	5	20,0
40000	5	5	91	9	50	9	17,2
50000	3	7	88	12	30	12	15,6
60000	2	8	82	18	20	18	18,4
70000	2	8	80	20	20	20	20,0
80000	2	8	77	23	20	23	22,4
90000	2	8	71	29	20	29	27,2
100000	2	8	67	33	20	33	30,4
110000	2	8	59	41	20	41	36,8
120000	1	9	55	45	10	45	38,0
130000	1	9	47	53	10	53	44,4
140000	1	9	44	56	10	56	46,8
150000	0	10	42	58	0	58	46,4
160000	0	10	38	62	0	62	49,6
170000	0	10	34	66	0	66	52,8
180000	0	10	29	71	0	71	56,8
190000	0	10	26	74	0	74	59,2
200000	0	10	25	75	0	75	60,0

Table 3: Threshold comparison results obtained from gesture nº1 data (punch gesture)

In this example it can be observed that the failure rate of the pre-calculated average threshold was 17,2%, and that the optimal threshold range was near the 50000 score threshold ratio, as it had a 15,6% failure rate.

Thresholds	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6	Gesture 7	Gesture 8	Gesture 9	Average Ratio
AvgThreshold	17,2	16,4	11,6	15,6	25,6	13,2	6,8	12,8	8,8	14,22
20000	18,0	18,0	18,0	18,0	18,0	18,0	18,0	18,0	8,8	16,98
30000	20,0	19,6	18,0	18,8	18,0	18,0	16,8	14,0	4,8	16,44
40000	17,2	18,4	18,0	20,4	18,0	18,0	10,8	12,0	0,8	14,84
50000	15,6	20,0	18,0	22,0	18,8	18,8	6,8	12,8	0,8	14,84
60000	18,4	19,2	18,0	17,6	18,8	18,4	6,8	10,4	2,4	14,44
70000	20,0	16,8	18,0	14,4	18,8	12,4	4,8	9,6	4,8	13,29
80000	22,4	18,0	18,0	12,4	18,8	13,2	2,8	10,4	11,2	14,13
90000	27,2	22,8	18,0	10,4	18,8	13,2	3,6	8,8	25,6	16,49
100000	30,4	25,2	18,8	6,4	18,8	13,2	3,6	12,0	30,4	17,64
110000	36,8	32,4	18,8	6,4	15,6	13,2	3,6	13,6	32,8	19,24
120000	38,0	35,6	16,8	6,4	16,4	13,2	4,4	15,2	36,0	20,22
130000	44,4	35,2	14,8	6,4	16,4	13,2	4,4	16,0	37,6	20,93
140000	46,8	35,6	15,6	6,4	18,8	13,2	4,4	16,8	41,6	22,13
150000	46,4	35,2	16,4	6,4	19,6	9,2	6,0	16,8	45,6	22,40
160000	49,6	37,6	10,4	6,4	21,2	9,2	4,0	17,6	47,2	22,58
170000	52,8	40,8	11,2	6,4	23,6	9,2	4,0	18,4	52,0	24,27
180000	56,8	44,0	10,0	6,4	24,0	10,0	4,0	18,4	56,0	25,51
190000	59,2	44,0	11,6	7,2	25,6	11,6	4,0	20,0	57,6	26,76
200000	60,0	44,8	11,6	8,0	24,8	13,2	4,8	20,8	58,4	27,38

Table 4: Threshold failure ratio comparison results for all gestures using the Euclidean Distance Algorithm

For the Euclidean distance error algorithm it can be observed that there are more near ideal results at lower threshold of 4000-6000 range than at a higher threshold range. It can also be observed that the average threshold ratio in this gesture had a good average failure ratio of 14,22%

Thresholds	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6	Gesture 7	Gesture 8	Gesture 9	Average Ratio
AvgThreshold	16,8	40,8	17,6	20,4	44,8	13,2	19,6	12,8	12,8	22,09
0,01	18,8	18,0	18,0	18,0	18,0	18,0	19,6	18,0	18,8	18,36
0,02	18,8	26,0	18,0	22,4	18,8	19,6	19,6	18,0	10,8	19,11
0,03	20,8	38,4	18,0	9,6	21,2	21,2	19,6	13,6	0,8	18,13
0,04	33,6	42,8	18,8	12,0	30,8	19,2	15,6	16,0	1,6	21,16
0,05	53,6	49,6	18,8	18,4	41,6	13,2	11,6	16,8	22,4	27,33
0,06	64,0	51,2	23,6	34,4	47,2	14,8	5,6	20,0	38,4	33,24
0,07	68,8	63,2	16,8	55,2	51,2	14,8	3,6	26,4	49,6	38,84
0,08	71,2	66,4	17,6	72,0	48,0	28,8	3,6	32,0	62,4	44,67
0,09	74,4	69,6	16,0	77,6	50,4	36,0	2,4	36,8	71,2	48,27
0,1	76,0	72,0	18,0	79,2	52,0	46,4	3,2	38,4	72,0	50,80
0,11	77,6	74,4	26,0	80,0	59,2	56,8	5,6	45,6	72,0	55,24
0,12	78,4	77,6	32,4	80,0	60,0	64,8	13,6	54,4	72,8	59,33

Table 6: Threshold failure ratio comparison results for all gestures using Angle Error Algorithm

In the case of the angle error algorithm were obtained similar results as the previous, as the smaller the threshold the more precise were the algorithm results, but in this case the average threshold ratio was not as successful showing a bigger average failure ratio than the previous algorithm 22,09%.

Thresholds	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6	Gesture 7	Gesture 8	Gesture 9	Average Ratio
AvgThreshold	19,6	30,4	12,4	29,2	63,6	13,2	8,8	13,6	8,8	22,18
20000	19,6	25,2	18,0	22,8	20,4	18,8	16,8	18,0	8,8	18,71
30000	22,8	27,6	18,0	26,8	23,6	19,6	14,8	14,0	4,8	19,11
40000	17,2	30,4	18,0	30,8	26,0	20,4	10,8	12,0	0,8	18,49
50000	14,8	33,6	18,0	30,4	30,8	20,4	6,8	12,4	3,2	18,93
60000	17,6	29,2	18,8	31,2	37,2	16,4	3,6	10,8	9,6	19,38
70000	24,0	28,8	18,8	30,4	46,0	12,4	3,6	12,0	17,6	21,51
80000	36,8	32,4	18,8	29,6	49,2	13,2	4,4	12,4	29,6	25,16
90000	44,8	35,6	18,8	29,2	51,6	13,2	4,4	12,0	40,8	27,82
100000	53,6	41,2	18,8	31,2	54,4	13,2	4,4	13,6	44,0	30,49
110000	57,6	43,6	18,8	32,8	54,0	13,2	5,2	15,2	47,2	31,96
120000	59,2	42,4	17,6	34,4	57,2	13,2	5,2	16,0	52,0	33,02
130000	68,0	44,8	15,6	35,2	58,8	13,2	6,8	16,8	56,0	35,02
140000	73,6	46,8	15,6	38,4	58,4	12,8	5,6	16,8	57,6	36,18
150000	76,0	45,6	12,4	41,6	60,0	11,6	6,4	17,6	60,0	36,80
160000	77,6	48,0	8,4	43,2	62,4	14,8	7,2	19,2	64,8	38,40
170000	77,6	48,0	10,0	47,2	64,0	18,0	8,8	21,6	67,2	40,27
180000	79,2	48,8	10,8	49,6	63,6	18,8	9,6	23,2	69,6	41,47
190000	79,2	52,0	12,4	52,8	62,4	19,6	11,2	23,2	70,4	42,58
200000	79,2	56,0	12,4	56,0	62,4	22,8	13,6	24,0	72,0	44,27

Table 5: Threshold failure ratio comparison results for all gestures using DP-Matching over the Euclidean Distance Algorithm

Using DP-Matching to process the results obtained through the Euclidean Distance Error algorithm, it is noticeable a decrease in the range between best threshold values showing more focus between 20000 and 40000 score, making it more viable for use of static value threshold instead of the average threshold, but overall the failure ratios were higher in this method.

Thresholds	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6	Gesture 7	Gesture 8	Gesture 9	Average Ratio
AvgThreshold	18,8	42,4	16,8	23,2	66,4	13,2	11,6	14,4	10,8	24,18
0,01	18,8	20,4	18,0	20,4	18,0	18,8	19,6	18,0	18,8	18,98
0,02	16,8	30,8	18,0	26,4	30,0	19,6	19,6	18,8	10,8	21,20
0,03	25,6	45,6	18,8	14,4	44,4	21,2	19,6	14,4	0,8	22,76
0,04	49,6	50,0	18,8	38,4	62,8	15,2	15,6	14,8	5,6	30,09
0,05	77,6	60,0	19,6	68,8	65,6	14,0	11,6	17,6	35,2	41,11
0,06	79,2	62,4	26,0	76,8	66,4	20,4	3,6	24,8	56,0	46,18
0,07	79,2	65,6	16,8	77,6	66,4	28,4	6,8	28,0	63,2	48,00
0,08	79,2	69,6	18,8	80,0	67,2	39,2	10,8	34,4	70,4	52,18
0,09	80,0	74,4	24,4	80,0	74,4	56,0	18,4	44,8	74,4	58,53
0,1	80,0	76,0	34,8	80,0	76,8	65,6	28,0	59,2	78,4	64,31
0,11	80,0	78,4	45,2	80,0	78,4	71,2	37,6	68,0	80,0	68,76
0,12	80,0	78,4	58,0	80,0	80,0	74,4	41,6	76,0	80,0	72,04

Table 7: Threshold failure ratio comparison results for all gestures using DP-Matching over the Angle Error Algorithm

Finally applying DP-Matching over the Angle Error obtained results with a similar range for optimal static thresholds, but the average threshold ratio gave a result of 24.18 which is the highest of all tested so far.

From this it was observed that the threshold variations would depend mostly on two main factors:

- Gesture size: even the attempts by the same user for the same password will vary in size, the more the size varies the less precision we can expect from the result.
- Gesture complexity: the more complex the gesture the harder to repeat it, gestures way too complex may create a very difficult to repeat setting even by the same user, creating a threshold too big that will end up worsening the differentiation between gestures.

Considering these two factors, it was decided that a good way to identify if a new gesture created by a user is optimal for use as a password or not. This inconsistency score is obtained through the formula: Inconsistency = Average Complexity/ Model Gesture Size.

A high inconsistency score is a sign that the user most likely failed at replicating his own gesture during gesture creation or that the created gesture is too complex to be easily replicated through human movement, while a low consistency score means that the user was able to replicate his created gesture perfectly. A maximum inconsistency score should be used to help decide if the user should retry creating his password or not. During this test a few of the comparisons that had biggest failure ratio scores were performed in gestures with a consistency score over 1000, so it was decided to use this as a maximum for the next test.

Algorithm Efficiency Comparison for all the gestures

From the previously obtained data we could compare the efficiency of each algorithm, by comparing how many gestures would be scored as true positives or negatives with different threshold values. As

the average threshold obtained revealed to be quite close to the ideal threshold for each gesture, this was used as the main threshold for the efficiency comparison.

	False Negatives	True Positives	True Negatives	False Positives	False Negative Ratio	False Positive Ratio	Failure Ratio
Euclidean Distance	38	52	835	65	42,2	7,2	14,2
Angle Error	47	43	779	121	52,2	13,4	21,2
Euclidean Distance DP-Matching	41	49	753	147	45,6	16,3	22,2
Angle Error DP-Matching	48	42	775	125	53,3	13,9	21,8

Table 8: Algorithm comparison results obtained from using the calculated average threshold

It can be verified that the for the set of gestures tested in this system the optimal algorithm to use in the Hidden Gesture password tests would be the Euclidean distance algorithm with the threshold calculated from the average consistency of the user's input, as the Euclidean distance obtained the biggest quantity of true values.

Note that although in this test the results obtained through DP-Matching were inferior to the ones obtained without it, this can only be concluded for this project, and not for gesture processing systems in general, as different ways of filtering the obtained data could possibly improve the results obtained through DP-Matching. This test's objective was just to find out of 4 methods used with the sample data obtained with the project would work better for the gesture comparison with the hidden button gesture.

4.2. Hidden Button Gesture efficiency against shoulder-surfing

4.2.1 Introduction

The purpose of this was to test the efficiency of the Hidden Button Gesture system in a worst case scenario shoulder-surfing case, in which the attackers are in close range to the victim able to witness the whole user interaction with a Wii Remote using it for a password, and to compare how the hidden gesture improves the security over a normal gesture password.

The users were asked to think of a gesture that would be used as a password in the Hidden Button Gesture application. It was explained to them that during the gesture input they would be asked to press the A button on the Wii Remote, and that only that part while the button was pressed would be the actual authentication password, being important that they would try to remember with some precision the parts of the gesture where they started and stop holding the button. They were then asked to use the Hidden Button Gesture application, by entering a username followed by the creation of their personal hidden gesture using the Wii Remote, holding the A button during the important part of the gesture, and pressing the B button at any time to finish the gesture recording.

While one of the users was entering his hidden gesture password, the other users would act as shoulder-surfers trying to memorize and figure what was the hidden gesture through observation from a short distance of about 1m. Some minutes after each observation event each of the shoulder surfers was asked to try to copy the hidden gesture password up to 5 attempts each.

Finally, in order to test the effectiveness and short term memorability of the hidden gesture the initial user was asked to try login again 1 hour after the password creation, attempting this 5 times.

4.2.2 Demographics

The testers selected for this test were 3 males and 1 female with ages between 23 and 27, they all had previous experience with the usage of a Wii Remote.

4.2.3 Collected Data

In order to collect and compare the information of obtained from each hidden button gesture with normal gestures, the Hidden Button Gesture application was set to save the gesture data in two ways at the same time:

- The entire gesture movement from the start of the gesture to the B button pressing to finish the gesture, recording the gesture as if it had no hidden button press.
- The hidden gesture password data that was only recorded during the time the user was holding the A button on the Wii Remote.

Considering this, two sets of results were obtained for each attempt at replicating the recorded gesture password. The results were organized in tables that show how consistent the user was when entering the password, the amount of true positives and false negatives when the user was asked to attempt login again after 1 hour, and the amount of failed and successful shoulder-surfing attempts. Obtaining the failure ratio in the same way as the previous test, in which a lower amount of false positives is considered more important to get than medium amount of false negatives.

Data set 1: Gesture password without hidden button pressing

	Consistency Score	False Negatives	True Positives	True Negatives	False Positives	False Negative Rate %	False Positive Rate %	Failure Ratio
User 1	972,8	2	2	14	1	50,0	6,7	15,3
User 2	93,4	2	2	12	3	50,0	20,0	26,0
User 3	77,5	1	3	14	1	25,0	6,7	10,3
User 4	94	2	2	14	1	50,0	6,7	15,3

Table 9: Comparison of the results for each user gesture without the hidden button

As we can see the in Table 9 for each of the gestures there was at least one successful shoulder-surf attack, being the 2nd user the one with the most amount of false positives, also should be noticed that the user that was most consistent in this had the highest amount of true positives.

We can conclude that for each of the users at least one of the shoulder-surfers managed to authenticate as if he was the original user at least once, the worst case being the gesture of user nº 2 which was replicated 3 times

Data set 2: Hidden Button Gesture

	Consistency Score	False Negatives	True Positives	True Negatives	False Positives	False Negative Ratio	False Positive Ratio	Failure Ratio
User 1	998	2	2	15	0	50,0	0,0	10,0
User 2	41	1	3	15	0	25,0	0,0	5,0
User 3	94	2	2	14	1	50,0	6,7	15,3
User 4	117	2	2	15	0	50,0	0,0	10,0

Table 10: Comparison of the results for each user gesture without the hidden button

Looking at the results obtained only when the A button was being pressed. We can see that there was significant decrease in the amount of successful shoulder surfing attacks, while keeping a similar amount of true positives. We can also notice that most of the consistency scores changed considerably showing that: user 2 was more consistent during the button press, than during the entire gesture process where he was not pressing the button, therefore even though his normal gesture could was the easiest to copy in the common gesture password; it became safe in this one. The only user whose hidden gesture was discovered once was user 3, but this was probably due to his gesture being composed of mostly button pressing without releasing from the start, while the others attempts lasted less, making it more obvious for the shoulder surfers to get even accidentally by just trying to hold the button during the entire gesture. The common 50% false negative ratio, might still prove to be an annoyance to some users and it is possible that during longer periods of time this would increase due to forgetfulness, and actual body changes. Although this could be increased or decreased through editing the way the threshold is calculated, increasing it to decrease security but have more tolerance to input errors, or decreasing it to have a much more secure system.

During this test the actions of the users were observed to verify if there were any problems in using the button while creating the gesture as they had previously used a gesture system. It was observed that every one of the users managed to follow the instructions given, and successfully create their own hidden button gesture, having only one of them created a gesture with a high (over 500) yet still tolerable inconsistency score.

4.3 Discussion

This project explored what efforts are being done to combat shoulder-surfing, by analyzing various projects, and trying to formalize ideas to improve them, be it in usability or security.

A comparison of some of the most relevant projects related to the shoulder-surfing problem were studied, from which was concluded that the more that is done to improve a system's security by increasing its complexity, the less usable it becomes. And a system with too complex of a secret, will most likely have low usability and fail to be used correctly, ending up with its users practicing unsafe procedures, be it by writing down their secret or constantly making mistakes during the interaction, leading to easier attacks. Therefore a good system should always have a good balance between security and usability. Knowing that even a small increase on the data obtained from could sometimes give a considerable increase of security. Having focused further readings on recent gesture authentication works, came the idea of improving these through the simple addition of a simple button press, that would be invisible to the observer, that would confirm when the real authentication token was being entered.

For these tests a gesture data comparison application was necessary to help record and verify which commonly used algorithms in this kind of systems would be optimal for a gesture, and how efficiently these algorithms work in a gesture application. From these tests a scoring system was created to help evaluate user inconsistency while creating a new gesture password, and an average threshold calculator in order to apply an optimal threshold depending on the complexity of the each user's password, so that the password is always possible to replicate by the same user, but never by someone who doesn't know the secret. The system was optimized so that password creation would filter gestures and utilize the most similar areas between different sized gestures, which proved to improve results as some of the false positives detected on the first user test were due to huge gesture size discrepancies.

The Gesture comparison tool was then prepared to compare data obtained from hidden button gestures recording only data when a button was being pressed while the accelerometer was used for gesture creation. And a new test was done to verify if the new Hidden Button gesture scheme was an improvement over the typical gesture used on the same system. This test showed that just adding the hidden button gesture decreased the amount of false positives without decreasing the optimal amount of false negatives, as only one of the shoulder-surfing attempts was successful while without the hidden gesture all the three gestures were replicated at least once by a shoulder-surfer.

The Hidden Button Gesture system created in this project has proven to be useful as an optional authentication method to combat shoulder-surfing, as the tests showed a decrease on the ratio of successful shoulder-surfing attacks performed with direct close range user observation on gesture passwords where it was only recorded the section of the gesture where the user presses the button, when compared with normal gestures. Even so, the gesture precision rate is still inferior over typical written passwords and other static data authentication types. Still it was proven that this kind of authentication can be used to help fight shoulder-surfing.

6. CONCLUSION

6.1 Comparison with other works

As a gesture is something humans do naturally it is easy to learn and create a gesture password, adding the Hidden Button Gesture scheme to it improves the security without decreasing its usability. In this project we used only one optimal score filter to select the best section of a gesture to be compared with another gesture, between two gestures of close size. The gesture comparison tool created to test these gestures helped to verify and select an optimal algorithm to test this kind of gesture input.

6.2 Future Work

From what was learned in this project, further investigation could be done to improve combating shoulder-surfing in gesture password, and even in other similar kind of authentication systems.

The same system could be tested with multiple hidden presses instead of just one during the password, although this could lead to unwanted complexity.

The hidden button press could also be added to other types of authentication that might involve interaction with devices that may hold buttons.

Other secondary channels of authentication could be explored in gesture authentication like spatial positioning detected through infra-red sensors, or video input received from a camera during the authentication process.

To obtain more precision in the comparison, the hidden button gesture could be tested with multiple types of data filtering and gesture data comparison algorithms, further investigation could be done researching the optimal algorithms and data filtering to be used with the hidden button press, or other secondary channels of authentication.

Finally to improve reliability of this kind of gesture system over time, as it people change physically over long periods of time and that might decrease the amount of successful logins, the gesture authentication system could be improved with machine learning, recording every single new gesture password and selecting a new model gesture from more recent sets.

6.3 Conclusion

The Hidden Button Gesture authentication technique became a reality on the theme of gesture and motion related authorization schemes. And proved that just a small addition of a secondary channel of authentication, like a hidden button press, can improve greatly the security, of the system while not adding too much complexity that would make the system unusable.

7. REFERENCES

1. Polybius, The General History of Polybius Volume 3 – The Roman Military System, Public domain translation by N.S.Gill. http://ancienthistory.about.com/library/bl/bl_text_polybius6.htm
2. Lawrence O'Gorman. Comparing Passwords, Tokens, and Biometrics for User Authentication. Proceedings of the IEEE, Vol 91, NO. 12, Pages 2022-2033. December 2003
3. Peipei Shi, Bo Zhu, and Amr Youssef. A PIN Entry Scheme Resistant to Recording-based Shoulder-Surfing. In *Third International Conference on Emerging Security Information, Systems and Technologies,* Pages 239-241. Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Quebec, H3G 2W1, Canada, 2009.
4. Varun Kumar, M. K. Gupta, Ashish Chaturvedi, Anuj Bhardwaj, and Manu Pratap Singh. Click to Zoom-inside Graphical Authentication. In *International Conference on Digital Image Processing*, Pages 238-242. 2009.
5. Tetsuji Takada. fakePointer: An Authentication Scheme for Improving Security against Peeping Attacks using Video Cameras. In *The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, Pages 395-400. National Institute of Advanced Industrial Science and Technology, 2-41-6, Aomi, Koto-ku, Tokyo, 135-0064, JAPAN, 2008
6. Alireza Pirayesh Sabzevar, Angelos Stavrou. Universal Multi-Factor Authentication Using Graphical Passwords. In *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems.* Pages 625-632. Computer Science Department, George Mason University, Fairfax, Virginia, 22030, USA, 2008
7. Haichang Gao, Xuewu Guo, Xiaoping Chen, Liming Wang, and Xiyang Liu. YAGP: Yet Another Graphical Password Strategy, In 2008 Annual Computer Security Applications Conference, Pages 121-129. Software Engineering Institute, Xidian University 710071, China, 2008
8. Danoush Hosseinzadeh, and Sridhar Krishnan. Gaussian Mixture Modeling of Keystroke Patterns for Biometric Applications. In *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, Vol. 38, No. 6*, Pages 816-826. November, 2008
9. Manu Kumar, Tal Garfinkel, Dan Boneh, and Terry Winograd. Reducing Shoulder-surfing by Using Gaze-based Password Entry, Pages 1-7. In *Symposium On Usable Privacy and Security (SOUPS) 2007*. July 18-20, 2007, Pittsburgh, PA, USA.
10. Desney S. Tan, Pedram Keyani, and Mary Czerwinski. Spy-resistant Keyboard: More secure password entry on public touch screen displays, Pages 1-9. In *Proceedings of OZCHI 2005,* Canberra, Australia. November 23 - 25, 2005.
11. S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *Proc. of the working conference on Advanced visual interfaces (AVI 2006)*, Pages 177–184. Venezia, Italy, May 2006. ACM Press.
12. Bruce Tognazinni. Design for Usability. In *Security and Usability, By Lorrie Faith Cranor, and Simson Garfinke,* Chapter 3, Pages 49-51, O'Reilly, August 2005, ISBN: 0-596-00827-9

13. Karen Renaud, Evaluating Authentication Mechanisms. In *Security and Usability, By Lorrie Faith Cranor, and Simson Garfinkel,* Chapter 6, Pages 111-134, O'Reilly, August 2005, ISBN: 0-596-00827-9
14. Jakob Nielsen. Chapter 2: What is Usability? In *Usability Engineering, Chapter 2.* Published by Morgan Kaufmann, San Francisco, 1994. ISBN 0-12-518406-9
15. Edward F. Gehringer. Choosing Passwords: Security and Human Factors. In *2002 International Symposium on Technology and Society (ISTAS'02)*, Pages 369-373. Department of Electrical and Computer Engineering, Department of Computer Science, North Carolina State University, USA, 2002
16. Furkan Tari, A. Ant Ozok, Stephen H. Holden. A Comparison of Perceived and Real Shoulder-surfing Risks between Alphanumeric and Graphical Passwords. In *SOUPS '06 Proceedings of the second symposium on Usable privacy and security*, Pages 56-66. UMBC, Baltimore, USA, 2006
17. Nick Huges. Preventing Careless Data Breaches - Who's responsible?. Infosecurity Europe 2008
18. Toni Perkovic, Mario Cagalj, Nitesh Saxena. Shoulder-Surfing Safe Login in a Partially Observable Attacker Model, Pages 1-8. University of Split, Polytechnic Institute of New York University.
19. Peerapong Uthansakul, Monthippa Uthansakul. Online Signature Verification Using Angular Transformation for e-Commerce Services. In *International Journal of Signal Processing* 6:1, Pages 33-38. 2010
20. Fuminori Okumura, Akira Kubota, Yoshinori Hatori, Kenji Matsuo, Masayuki Hashimoto, Atsuhi Koike. A Study on Biometric Authentication based on Arm Sweep Action with Acceleration Sensor. Department of Information Processing, Tokyo Institute of Technology, Yokohama, Japan. 2006
21. CMS Information Systems - Threat Identification Resource Version 1.0, Pages 2-3. Centers for Medicare & Medicaid Services, Baltimore, Maryland, May 7, 2002
22. Howard Wisniowski. Analog Devices and Nintendo Collaboration drives video game innovation with IMEMS motion signal processing technology. Analog Devices, 2006. http://www.analog.com/en/press-release/May_09_2006_ADI_Nintendo_Collaboration/press.html
23. Small, Low Power, 3-Axis ±3g iMEMS® Accelerometer ADXL330., Analog Devices, 2006 http://www.analog.com/en/sensors/inertial-sensors/adxl330/products/product.html
24. Ming Ki Chong, Gary Marsden. Exploring the Use of Discrete Gestures for Authentication, department of Computer Science, In *Human-Computer Interaction – INTERACT 2009 Lecture Notes in Computer Science*, Part II, Pages 205-213. University of Cape Town, Rondebosch, Cape Town, South Africa, 2009
25. Pavel Lozhnikov. "Teofrast" – A Biometric System Based on Users' Identification Through Handwriting Dynamics. In *Exploiting the Knowledge Economy: Issues, Applications, Case Studies*, Pages 1-7. Siberian State Automobile and Highway academy (SibADI), Omsk, Russia, 2002
26. Steve Cassidy. COMP449: Speech Recognition, Chapter 11. Matching Patterns in Time, 11.3. Practical DP Matching. http://web.science.mq.edu.au/~cassidy/comp449/html/index.html Department of Computing, Macquarie University, Sydney, Australia. 2002

27. M. Brader. Shoulder-surfing automated. Risks Digest, 19, 1998. http://catless.ncl.ac.uk/Risks/19.70.html#subj8.1
28. Robert Fried. I've got my eye on you. Crime and Clues, 2009. http://www.crimeandclues.com/index.php/digital-evidence/44-computer-investigations/136-ive-got-my-eye-on-you