DUBLIN INSTITUTE of TECHNOLOGY
Institiúid Teicneolaíochta Bhaile Átha Cliath

Multi-Factor Authentication System
Final Year Project Report

DT282
BSc in Computer Science

Daire Grimes
Paul Doyle

School of Computing
Dublin Institute of Technology

13th April 2018

Abstract

This project addresses the need and demand for more secure electronic devices. Customers want more security with their electronic devices, but expect it to be fast and user friendly.

The work investigates techniques directed towards making computers more secure by adding multi-factor authentication within a Windows 10 environment. The security is split into two parts. The first part is during the login process. The user must provide biometrical and token authentication alongside the password. The second part is based on an additional authentication process that ensures that the system self-locks while the user leaves the PC and unlocks using facial recognition when the user returns.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

_____

Daire Grimes

13th April 2018

Acknowledgements

I would first like to thank my supervisor Paul Doyle for guiding me through project and for giving me great ideas to work with. I would also like to that my friends and family for their support throughout the past four years.

Finally, my thanks go to Neti Shah for proof reading my work.

Table of Figures

Table of Contents

Introduction

1.1 Background

In 2013, whistle-blower Edward Snowden released leaks from the National Security agency (NSA). These leaks showed the NSA were spying on the whole world. (Glenn Greenwald, 2013)

Ever since these revelations, the public has become more aware of their digital security.

1.2 Project Objectives

The objective of this project is to create a secure multi-factor authentication system that can easily distinguish users based on multiple characteristics. For many years, biometrics and the need for better security have been exponentially growing and this project is designed to meet that growing need.

However, over time the user undergoes physical changes and the system will need to compensate for that. This project will utilize machine learning to learn about the user so that they can continue to use the system seamlessly.

1.4 Project Challenges

One of the main Challenges of this project was to access Windows authentication process due to it being proprietary software. Also, time management, there were a lot of tasks in this project to complete from a wide range of different programming languages.

1.5 Structure of this Document

1.5.1 Research

The research section discusses everything that had happened before the implementation had begun. This includes, discoveries, machine learning and authentication techniques.

1.5.2 Technologies Researched

This section highlights and discusses the multiple programming languages and applications that were researched to aid with the development of this project.

1.5.3 Approach and Methodology

This section goes over which methodology was chosen.

1.5.4 Design

The design section, each component of the project is described and it shows how the whole project works.

1.5.5 Architecture and Development

This is the most technical part of this document. It outlines in detail how each component was implemented.

1.5.6 System Validation

In this section, there will be the testing and the demonstration.

1.5.7 Conclusion

Finally, the conclusion has the project plan, strengths and weaknesses and the future work of this project.

1. Research

2.1 Background Research

2.2 Introduction

Numerous types of authentication methods had to be investigated along with machine learning to start development.

2.3 Introduction to Multi-Factor Authentication

Multi-Factor authentication focuses on making computers more secure. This includes authenticating from:

1. **Something you Know:** This is when the user provides authentication with their knowledge. Normally, only a password is only used for authentication, which can lead to many problems. First, passwords are prone to be being weak, if users ignore standard security rules like avoiding trivial sentences or not using upper case letters, numbers or special characters. Even if standard security rules are ensured, it can be difficult to remember them. (Neves, n.d.) Additionally, many users use the same password for multiple accounts. This is dangerous, considering that if a company of which the user is a customer of is hacked the password could get into the wrong hands, all their accounts could be compromised. (Pin Shen Teh, 2013)

2. **Something you are:** Also known as biometrics. This is the measurement and analysis of people's behaviour and characteristics. In the last ten years the field of Biometrics had significantly advanced. Today, biometrics is being used in every major smart phone. In 2013, Apple released their first phone with fingerprint recognition and in 2017 they introduced the iPhone X with facial recognition. The two main types of biometrics that will be of focus are fingerprint and face. (Pin Shen Teh, 2013)

3. **Something you do:** This is when the computer tracks your behaviour while using the computer. Behavioural biometrics analyses human behaviour for authentication. It continuously collects information on how a user interacts with an electronic device. When the behaviour discontinues to match the (standard behaviour/expected behaviour), the user will be locked out of the machine. (Pin Shen Teh, 2013)

4. **Something you have:** Physical Authentication is when the user is required to have a physical object. This can be a wide range of devices such as, UBSs, key cards or smartphones. For example, at an ATM machine a user must provide a card and a PIN. For this project a smart phone would be ideal. According to (Statista, 2014), by the year 2020, there will be 2.87

billion smart phone users on the planet and smart phones come with their own biometric features.

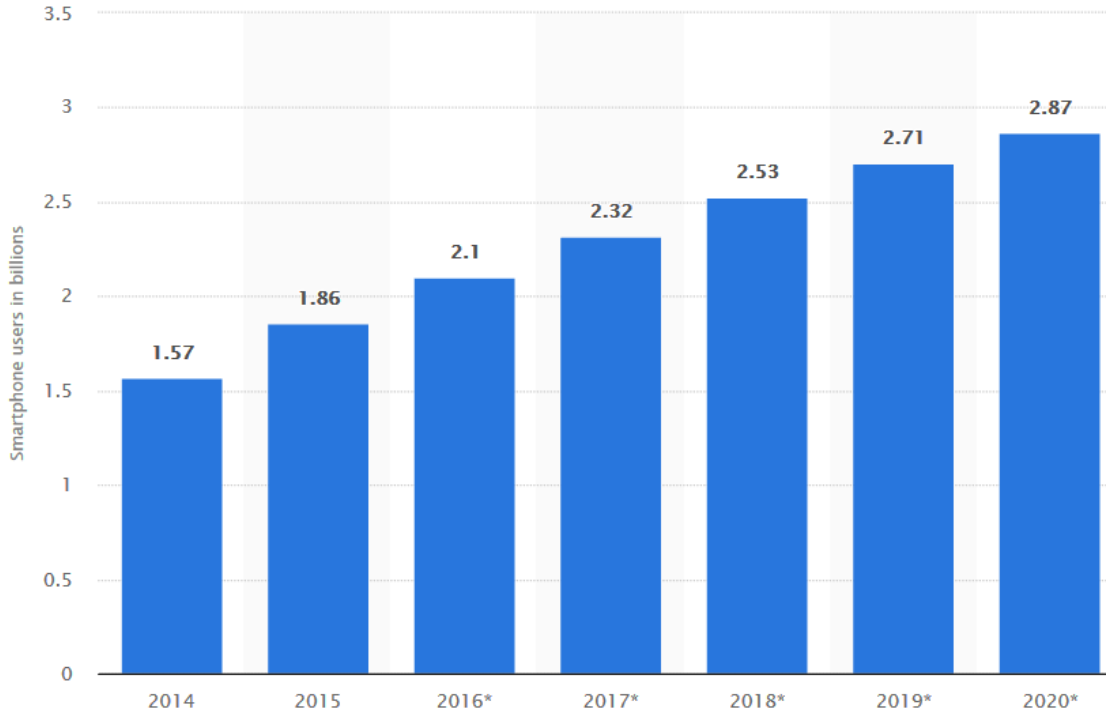


Fig 1. Usage of smart phones (Statista, 2014)

2.3 Authentication methods

Here, multiple types of multi-factor authentication will be explored.

2.4 Facial Recognition

Facial recognition is authentication based on the user's facial patterns. Now that facial recognition has improved significantly, more and more smart phone companies are adopting the technique and it will be used significantly in the future. First, the process of facial recognition is completed in three steps, Detection, Extraction and Recognition.

2.4.1 Face Detection

Detection is the process of locating a face is a given frame. For humans, this task is easy due to evolution but for machines this can be difficult due to facial expressions and to light interference. (Ankit Srivastava, 2017)

There are multiple types of face detection techniques, the most well-known and used is the Viola-Jones algorithm. First, the frame is converted into greyscale because then the algorithm only needs to work with 256 values. Then it works

by searching for the common shared properties of every face. For example, in every face, the eyes, nose, and mouth will always be in the same location. Common features of the face are:

- Nose is brighter than other regions
- Upper cheeks are brighter than the eyes
- Eyes are above nose, nose is above lips

These features are calculated in an algorithm, this then uses a 24*24 window and scans the frame. Each dark part of the frame is subtracted by the lighters parts. This produces 16,000 features of the frame. The next stage is optimising these features quickly and loading them into a classifier to detect the face. (Ankit Srivastava, 2017)

In face detection, there are many other different types of face detection techniques such as, the Hausdorff, the Eigenface and the Neural Network cascade. But in this system the Viola-Jones technique will be used to the vast amount of research in it and its impact in recent years. (Subrat Kumar Rath, 2014)

2.4.2 Extraction and Recognition

For the processes of extraction and recognition, there are two different types of methods. Holistic and Feature-Based.

2.4.2.1 Holistic

In this type, the identification happens with the entire face. A popular method is the eigenface method. This uses PCA (principal component analysis) to convert the 2-D face into a linear graph and to strengthen the features of the face. This helps to find the main component of the face. Also, a similar technique is the Fisherface method, but instead of using principal component analysis, it uses (LDA) linear discriminant Analysis. The technique generalizes the face through the mean values. Unfortunately, these two algorithms can be badly distorted under strong lighting conditions.
(Mejda Chihaoui *, 2016)

Fig 2. Eigenfaces
(Mejda Chihaoui *, 2016)

Another holistic method is using Local Binary Patterns. First, the face is converted to grayscale. Then, the algorithm scans through each pixel in a 3*3 matrix. The pixel in the centre is then compared to each surrounding pixel. If the centre pixel is greater or the same as to its surrounding pixels, it is then given a one. If it is smaller it is given a 0. Each of the eight pixels that are then calculated are put into bytes of data. This method is produced throughout the image. (Mukesh D. Rinwa, 2015) These values are then graphed, so when the authentication happens the faces are then compared to what is stored. An advantage for this method is that it is still sufficient in different light sources from fig. 3

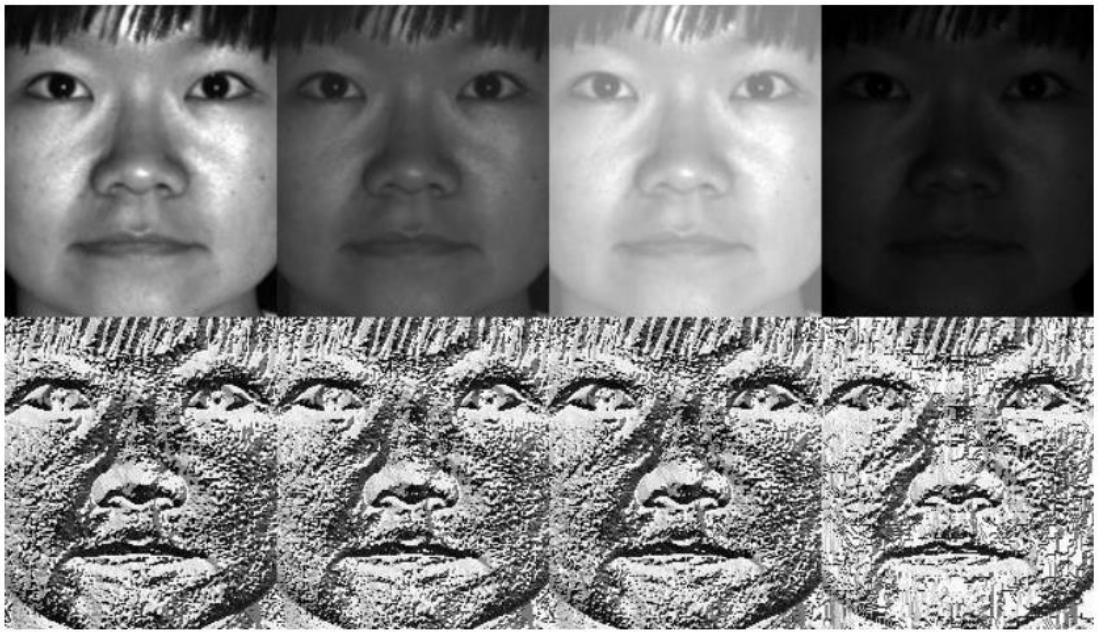


Fig 3. Local Binary Patterns (OpenCV, 2017)

2.4.2.2 Feature-Based

In feature-based techniques, the main landmarks of the face are detected such as the face, eyes, nose etc and a geometric algorithm is calculated or the data will be used in classifiers. But the extraction phase can be difficult to implement. After the face is detected, each landmark must be precisely detected. Unfortunately, a face under strong light conditions can distort the position of the landmarks or a specific pose could throw off the algorithm. This also involves very advanced image processing skills.

2.4.3 Conclusion

After researching these two different types of facial recognition, feature-based algorithms were not going to be used to the difficulty of detecting each landmark on the face. Because of that, a holistic technique will be used instead. Eigenfaces and the Fisherface algorithm both look very promising but unfortunately, they can be distorted through strong light conditions. Therefore, LBP will be used in this system.

2.5 Spoof Detection for Faces in Visible Light

Spoofing is when an adversary tries to authentication with using an object that resembles the user's face. It can be achieved in three different ways.

1. A photo
2. A video
3. A 3-D model of the user

The detection of spoofing is spilt into four different. Motion, Texture, life-sign and optical flow analysis.

1. Motion and Life-Sign Based

The first method is motion based analysis. This looks for signs of motion in the face. If an adversary holds up a photo of the user, it will immediately detect spoofing. Other methods involve asking the user to move their face or eyes in a certain direction. The user could be asked to blink or to roll their eyes or head. This is excellent for detection spoofing in videos.

There are other techniques in motion based are where the foreground and background are compared. This is the read into an optical flow analyser and a score is outputted if the face is real. (Manpreet Bagga, 2016)

Additionally, there are motion based methods which try to see the blood flow the user's face through Euler Video Magnification originally based on a paper from (Hao-Yu Wu, 2012). This technique tries to see show changes in the face that cannot be seen through the human eye. Every time the heart beats, skin becomes slightly redder during this process. This technique, first locates the face and amplifies the red colour from the skin. This technique is already used in the medical industry and would be excellent for spoof detection. (Klitsie, 2015)

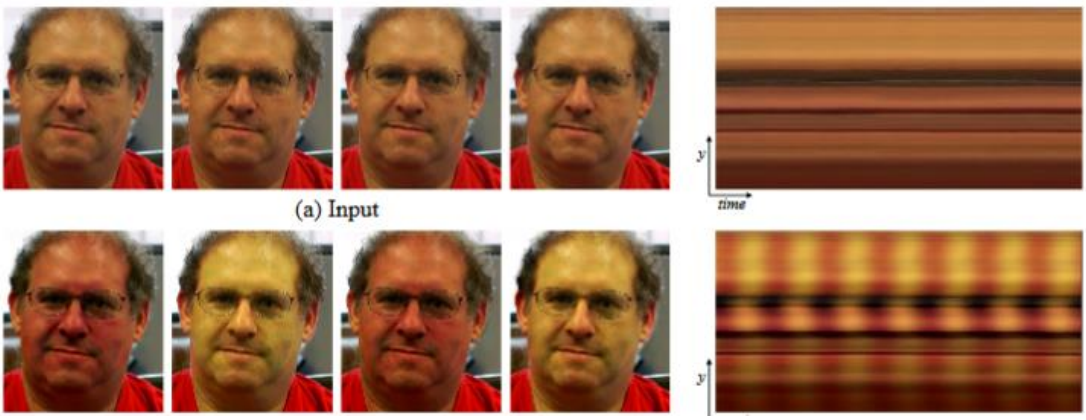


Fig 4. Euler Video Magnification
(Hao-Yu Wu, 2012)

2. Texture Based

Texture based methods, try to differentiate fake faces from real ones from the actual texture of the face. This is completed by analysing the shape and the detailedness of the face. If an adversary was to print a face on paper, this would have a different texture to an actual face. Local Binary Patterns (LPB) are popular for carrying out this this technique. Them a SVM is used to try and classify the real from the fake. (Hao-Yu Wu, 2012).

3. Optical Flow Based

This technique is the sum of four basic movement types, rotation, transition, moving movement and swing. In 2D and 3D, transition, moving and rotation have the same optical flow fields but the swing movement produces a different is different from 2D to 3D images. These four movements types are the used to differentiate the dimensions of the image.

2.6 Facial Recognition in Infrared (IR)

Another way to combat this is not to use traditional visible light cameras. But instead to use an Infrared (IR) camera. Both the iPhone X and the Microsoft surface laptops have both switched to IR cameras for their respective facial recognition authentication. If an adversary holds a video or certain type of pictures of a face. The IR camera will not even register it. This is due to IR being on a different wavelength to regular visible light. Other advantages of IR include being authenticate in dark conditions and being able to work during any lighting conditions. (Reza Shoja Ghiass, 2013)

Also, another problem with facial recognition with is the twin paradox. In visible light, the appearance of identical twins is nearly indistinguishable. To distinguish them, in visible light and iris scan must be used. But unfortunately, with that there are hardware costs. (P. Jonathon Phillips, 2011) However, on the IR thermal spectrum, twins can be distinguished. (methodologies, 2014)

2.7 Fingerprint Recognition

Fingerprint recognition is the most commonly used and well-known form of biometrics. For the past few decades, it has been used extensively and because there is an extremely low probability that a user shares a fingerprint pattern with someone. The authentication is done by detecting the whorls, loops and arches of a print. There are four main types of fingerprint readers:

1. Optical: This is the most common. This type takes a digital photograph of your fingerprint and then carries out the authentication.
2. Capacitive: This method uses an electrical current. When a fingerprint is placed on a reader, a distinction is made from the valleys and ridges of the print.

3. Ultrasound: This is the most recently developed type of fingerprint recognition. It is When the fingerprint is placed it sends out high frequency waves. The skin and the ridges are exploited, and this information gets sent back to the sensor.
4. Thermal: This method uses a pyroelectric material which converts the variants of temperature into a voltage. This voltage is recorded, and the authentication is carried out. (Marasco, E, 2015)

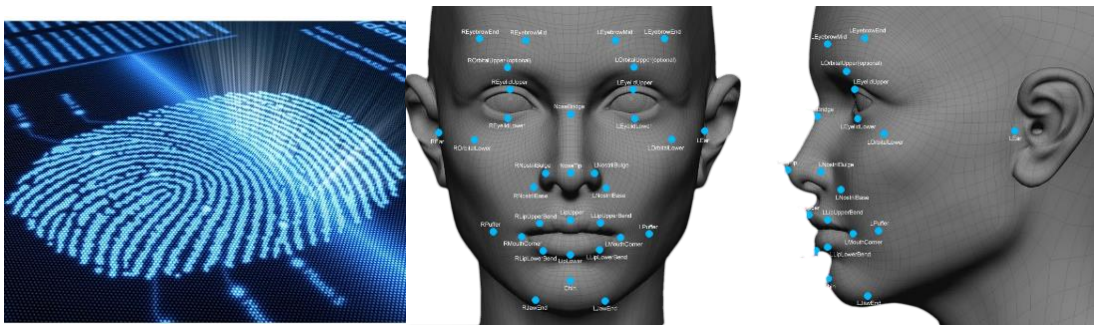


Fig.5 Facial and fingerprint recognition
(Herh, 2016) (Chaney, n.d.)

2.8 Keystroke Dynamics

Keystroke Dynamics monitors the users typing patterns for authentication. This is based on by using three factors.

1. Key Hold time
2. Key Switch time
3. Speed

It is added the data to a classifier and returns a result. If the keystroke is within a small margin of error the authentication is successful. (Pin Shen Teh, 2013)There are two types of this biometrics technique. Static and Dynamic. Dynamic monitors the keyboard throughout the usage of the computer while static is only used at specific time i.e. entering a password. This can be encoded in many ways, but the two main ones are using binary or using the time. To encode with binary every time a key would be held down it would encode a variable with zeros. Then during the switch time, it would encode the same variable with ones. (Mariusz Rybnik, 2009)

The advantage of building a system that monitors keystroke patterns does have an advantage where no extra hardware needs to be purchased, unlike other types of biometric methods where acquiring hardware can be expensive. But unfortunately, it does have disadvantages. Keystrokes can be affected if the user is fatigued, injury or distraction. Also, if the user uses a different type of keyboard, that could affect the over result. Also, building a system that constantly monitors the user's keystrokes is very difficult to implement and needs mush training. But designing a system that is just for passwords is

possible. Here, the user must type their password out multiple times while registering and must have a large password to be successful.
(Warwick, 2013)

However, there are also some security concerns with this method. These can be broken down into four different types.

1. Shoulder Surfing

When the adversary looks that the user typing their password in to try to replicate the user's typing pattern. In static typing, this could be dangerous but if there is a threshold of failed attempts, it would be difficult to bypass the system, with dynamic typing, this is nearly impossible. (Arwa Alsultan, 2013)

2. Spyware

Spyware is malicious software people download unintentionally and without their consent. This is the biggest threat to keystroke dynamics according to (Arwa Alsultan, 2013). But fortunately, to program spyware to do this can be a difficult task.

3. Social Engineering

Social engineering is when is when the user is tricked or manipulated into giving away their private information. But with keystroke dynamics, this can be very difficult to implement. But it happens through phishing. This normally happens when the user receives an email and is tricked into giving away private email. Adversaries can get typing patterns through the user typing in a form. (Arwa Alsultan, 2013)

4. Guessing

Fortunately, guessing is the hardest way to bypass keystroke authentication. There is too much combinations that the user can choose to type with. (Arwa Alsultan, 2013)

2.8.1 Factors Affecting Keystroke Dynamics (KD)

Unfortunately, there are multiple factors that can affect the performance of typing patterns.

First according to (Gupta, 1990), the length of the password is very important for successful authentication. The smaller the password the easier it is to spoof. Also, the timing of the clock resolution is important for clear and accurate results. When a clock speed of 15 ms was used instead of 1 ms there was a difference of 4.2%

The emotional state of the user can affect the typing speed dramatically according to (Sasikumar., 2010) by a 70% reduction. When users are in a positive emotional state their speed can increase by 83%. Additionally, the position where the user is sitting drastically alters their typing patterns.

(M. Rybnik, 2008) discovered how each language affects typing patterns. Having special characters that are not in English can be difficult for English speakers to spoof.
Having multiple alphanumeric characters can boost the chances of being identified with the valid user.

2.9 Mouse Movement Dynamics

Mouse movement dynamics is a technique of monitoring users on how they use their mouse. The features that are used to distinguish are:

1. Mouse-Move: Movement between two points.
2. Drag and drop: The time the user holds and drops the mouse buttons.
3. Point and click: Movements between two points and a click
4. Silence: No movement

(Feher, C)

But unfortunately, the amount of research compared to keystroke dynamics is very low. Additionally, there are many problems to this method according to a survey done by (Yu, 2011). First, the enrolment phase was difficult to implement. It is not other biometric techniques where the enrolment phase happens instantly. Other factors that caused problems were environment variables. Depending in how the user sat and whether they were using a touchpad dramatically affected their results.

2.10 Token Authentication for Smartphones

Token Authentication falls under the category of “Something you have”. This method works by a server creating a token. This token can be anything but it is normally just a large of numbers. In the past number of year this method has become very popular and it is very widespread online. In the past, when signing up for a new website, people had to complete pages of sign-up sheets but now all people just sign-up with their email and the authentication happens in a matter of seconds. But in this project, the token authentication must be offline because there are many situations where the user will not have internet access, also there are many things a user can do while being offline on a computer. This would require the client receiving the token at the setup up phase instead of it receiving the token at the authentication phase. A smartphone would be ideal for this because as most people have already.

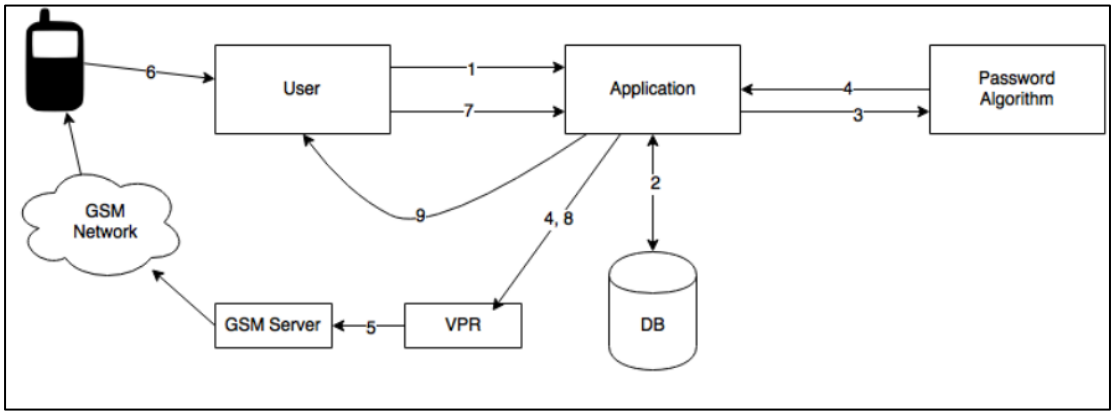


Fig 6. Token Authentication (Neves, n.d.)

2.11 Conclusion

After researching these types of authentication. The factors that will be included in the system will be facial recognition, keystroke dynamics and token authentication with a smart phone. For the system to know that the user has left, this would need to be implemented in facial recognition. Token authentication with a smartphone was chosen due to it combining multiple factors. This would include fingerprint recognition due to it being available on the smartphone and it is "something you have" which included physical biometrics. For Keystroke dynamics, static is chosen over dynamic because this would be difficult to implement due to user's typing patterns constantly changing due to the angle of how they are sitting or due to fatigue. Mouse Movement dynamics will not be interested due to little research in that area.

2.12 Account Lockout Threshold

As this is a security and authentication project. The user will make mistakes during their authentication and adversaries will try to breach the system. The Account Lockout Threshold policy determines how many times the user can attempt to login incorrectly before being locked out of the system. Here this topic will be explored.

This system a system that is implemented on a Windows environment and Microsoft have their own guidelines for custom authentication methods. For passwords, the user can specify form unlimited attempts to 999 attempts. Additionally, For the facial recognition factor, the user can also manually change the threshold number for failed attempts and for the duration of the authentication time.

For token authentication with a smart phone, the recommended time is 20 seconds. When the user logs in, there will be a message on the screen which asks to connect the smartphone. If there is not reply within 20 seconds the session will expire. (Satran, 2017)

2.12.1 Conclusion

After researching this topic, the account lockout threshold will be set manually by the user. This will allow the user to easily customise their security preferences to their liking. But there will be a backup in this system, when the user is first setting up the system, they will have to setup an eight-digit PIN number. This will be the only time that they will be asked for that number. If during the multi-factor authentication process the user gets locked out, they will then have to revert to that PIN to sign in. When they user signs back in via the PIN, they will reset their entire multi-factor credentials.

2.13 Object Tracking

In this system, when the user moves from the frame of the camera. It will lock the system. To achieve this, object tracking must be implemented.

Object tracking is implemented by

OpenCV comes with a wide range of tracking algorithm. For this system, the face must be constantly tracked and the algorithm must not affect the speed and runtime of the system.
OpenCV currently implements six algorithms.

1. BOOSTING
2. MIL
3. KCF
4. TLD
5. MEDIANFLOW
6. GOTURN

But according to (Mallick, 2017), the Median flow algorithm is the most sufficient due to it working well in predictable conditions. The face of the user will close to the camera and will only on rare occasions have other objects covering it.

2.14 Machine Learning

For this project, machine learning would help achieve better authentication for my facial recognition and keystroke dynamics.

2.14.1 Keystroke Dynamics

Machine learning algorithms that were used with static keystroke dynamics will be explored in this section are Statistical Algorithms, Neural Networks and Support Vector Machines.

2.12.1.1 Statistical Algorithms

The is the simplest approach to machine learning in this area. This algorithm consists of using the mean and standard deviations to classify data. From the surveys that were done in (Salil P. Banerjee, 2012). On small data sets, this algorithm was successful with reaching a false acceptance rate (FAR) of 0.25% and a false reject rate of 16.36%. But unfortunately, using these algorithms lack a training stage and are not sufficient in large data sets.

2.12.1.2 Artificial Neural Networks (ANN)

An artificial neural network is machine learning technique inspired by biological neurons in the brain. There are two different types of ANN, supervised and unsupervised. In (Salil P. Banerjee, 2012) , three different types of algorithms were used for testing. Backpropagation, Sum of All products and a hybrid of the sum of all products. The best algorithm was the latter was the best with a 97.8% identification rate. Overall, the results produced very good results but the disadvantages were that the training phase takes a long time.

2.12.1.3 Support Vector Machine (SVM)

A support vector machine is a supervised machine learning algorithm that is used for classification and regression. However, it is rarely used for regression. This algorithm works by plotting data in n-dimensional space and differentiating the data. Also, in (Salil P. Banerjee, 2012) an identification rate of 95% was reached with less training time than the neural network.

2.12.1.4 Conclusion

For the implementation of the keystroke dynamics in this project, the user will be asked to type their password in multiple times for the algorithm to learn their typing patterns. Implementing this with a neural network would cause problems because there would not be enough data for it to be sufficient. Implementing the algorithm using a statistical algorithm would be sufficient in the short run due to it learning quite quickly. But in the long run, it would not learn as well as other algorithms. That leaves the SVM, this learns quickly and it would work well in the long run.

3 Technologies Researched

3.1 OpenCV

OpenCV is an open source computer vision library. It was built to create computer vision applications. The library contains over 2500 algorithms and would be perfect for the facial recognition implementation in this project.

Since OpenCV is under the BSD license, it is available for academic and commercial use. Technology giants such as Google, Microsoft and Intel use this library and it also has a great online open source community for help. (OpenCV, 2017)

OpenCV comes in two languages Python and C++. Python was first considered because of its simplicity and flexibility and it is the forth used language on GitHub. (Python, 2016) Due to Python's simplicity, it was chosen for this implementation.

3.2 Android Studio

The user will use token authentication while logging in. This involves connecting their computer to their phone which has a fingerprint reader. The App that will be used is to be implemented in Android studio. Android was picked over iOS since as of 2016 Android holds 86.2% of the worldwide market share (Android, 2017).

3.3 Windows Authentication

In this system, the user will have to add extra security to the Windows OS to achieve the multiple factors of Authentication. This can be a difficult task, due to Windows being proprietary software. But, fortunately Microsoft have their own guidelines on how to achieve this and there are third- party applications which also can achieve the same goal.

3.3.1 PGina

PGina is third-party and open source credential provider for Windows. It allows users to manage their own process for logging in to Windows by adding in extra plugins to control the login process.

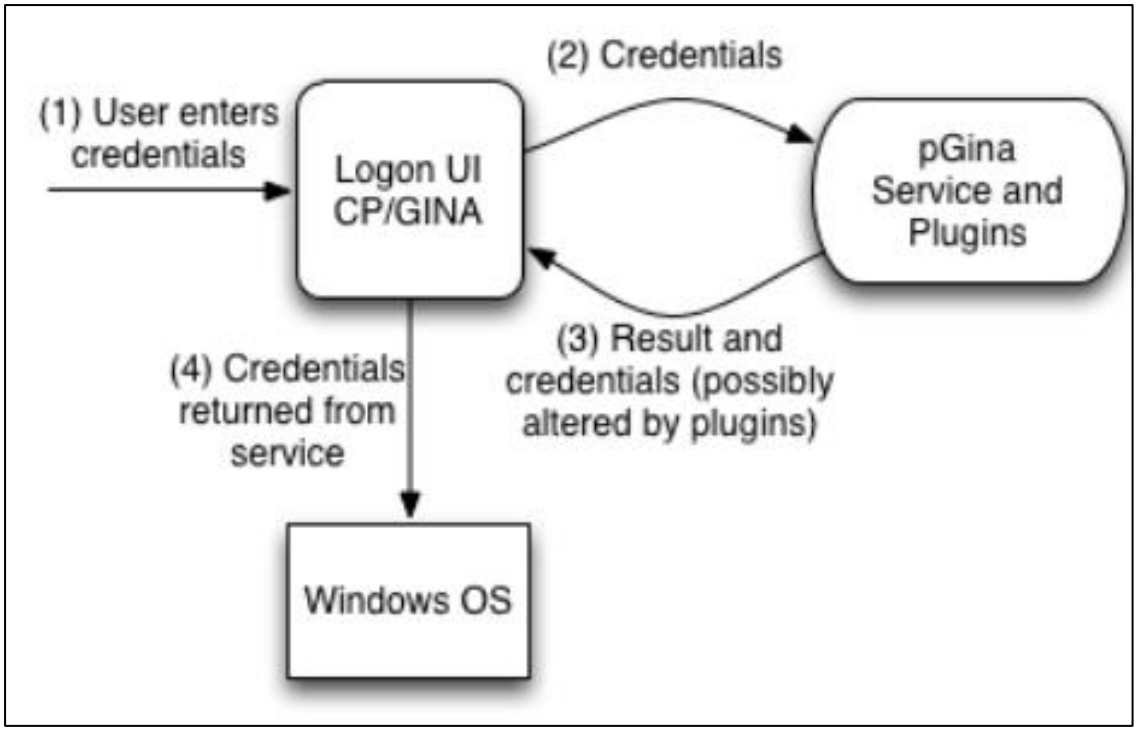


Fig 7. pGina architecture

PGina works in the following steps:

1. The user enters their credentials in the pGina interface. The interface can be several options e.g. Facial recognition or a password. These are called credential providers.
2. The credential provider passes the credentials through a plugin pipeline. This is where the information gets processed. This is created by the user.
3. After the processing the plugin returns success or failure.
4. If it is successful, pGina notifies Windows to log in. pGina does not log the user in, Windows does that. This is because Windows ensures/ that the account exists on the machine. (pGina, 2014)

3.3.2 Windows Credential Provider

To add custom authentication to Windows without third party apps like pGina, Microsoft uses the Credential provider. Ever since Windows Vista, they have been using this system which dramatically decreases the difficulty of developing this. (Microsoft, 2016) The authentication process is called Winlogon and it uses C++ and COM programming. With this, the developer creates two COM interfaces. ICredntialProvider and the ICredentialProviderCredential.

The ICredentialProviderCredential interface is used to respond to the user input and the ICredntialProvider interface defines the behaviour of the credential provider. This means that if the users wanted their token authentication on a

different thread from their password input, they would use the ICredntialProvider to achieve this.

Ever since this version of the Credential provider was created, the developer does not have to worry about most of the implementation and can easily manage everything. (Microsoft, n.d.)

3.4 Connectivity

In this project, for the login process, the smart phone needs to the computer for the token authentication. Also, I want the project to work without internet access. The two most popular types of connectivity are Near Field Communication, Bluetooth Low Energy and Bluetooth.

3.4.1 Near Field Communication (NFC)

NFC allows two very close devices to connect with very low energy. Lately with contactless payments NFC has become very popular. Now it is normal for a credit card to contain a small NFC chip inside it for quick payments. Data transfers from each device at a rate of 106-424 Kbit/s which would be sufficient for the token authentication but unfortunately the range of the two devices must be in millimetres. For the token authentication to be easily completed there must be wider range. (Gleason, 2016)

3.4.2 Bluetooth

Bluetooth is a very popular method of communication. It is used frequency and allows to connect to multiple devices. Bluetooth uses the frequency 2.4GHz which is the same as Wi-Fi. But Bluetooth hops from different frequency to stop congestion. With a high data rate of 1.2 – 24 Mbit this technology would be sufficient for the token authentication. (Gleason, 2016)

3.4.3 Bluetooth Low Energy (BLE)

BLE is a subset of regular Bluetooth which was designed for low energy applications. The maximum data rate id 0.27Mbit/s and the power consumption is 0.01W – 0.5W. This is approximately half of regular Bluetooth. Today, BLE is used in smart watches, fitness trackers and medical devices. (Gleason, 2016)

3.4.4 Conclusion

For the connectivity, BLE is the most optimal choice for this system because of its low energy usage and its faster rate of sending small amounts of data. But unfortunately, BLE is new technology and the resources for it are scarce. NFC looks also like a promising technology but the range is too short for it to be practical for this project. Therefore, Bluetooth will be the choice for connectivity

due to its plentiful supply of resources online and its long range for authentication. (32.feet.NET, 2017)

3.5 Testing

After researching on how to create a custom credential provider, many recommendations were found that advised testing in a virtual environment (Phaetto, 2017), (Alun Jones, 2011). A virtual environment offers protection against mistakes during the implementation of the credential provider. To avoid being locked out of the computer while testing, I will not be testing on my computer.

For this task, it was a choice between VMware and Virtual Box. First, VMware was considered because it is the industry leader. But virtual box was chosen because it is free, open source and prior knowledge.

3.6 Password and Information Storage

For this system, many factors must be secured and protected from adversaries. For the token authentication, the token that the server creates must be secured and encrypted on the Windows and android OS. For the facial recognition and keystroke dynamics information, they need to be stored on the Windows OS.

3.6.1 SQLite

SQLite is a light server less database developed in C that can be used on any platform. This works great with small apps and it can travel along the program. It is public domain and free for use of any purpose. It is also very popular, and there are many resources online that can aid this project. It also works on android devices. (SQLite, 2017)

4 Approach and Methodology

The approach for the project is based on the prototyping methodology. This is when the developer creates a prototype then tests and rebuilds until the system is good enough to be deployed. This was chosen because it allows for early identification of what works well and what doesn't.

First, the requirements will be established in as much detail as possible. Then a design is created for the system and a prototype is made. At this stage, the developer notes the strengths and weaknesses and analyses what should be added and has should be removed.

After this analysis, the second prototype in made. These steps are repeated as many times as necessary until the developer is satisfied and reaches the expected outcome. The final system is then evaluated and tested. (shopeemart, n.d.)

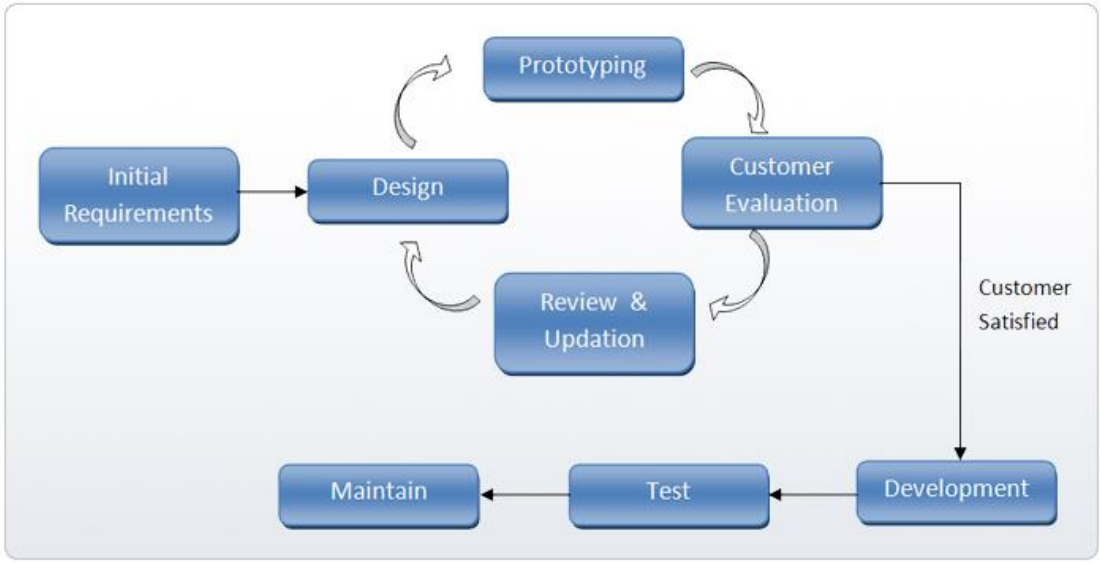


Fig 8. Prototype methodology
(shopeemart, n.d.)

5 Design

5.1 Technical Architecture Diagram

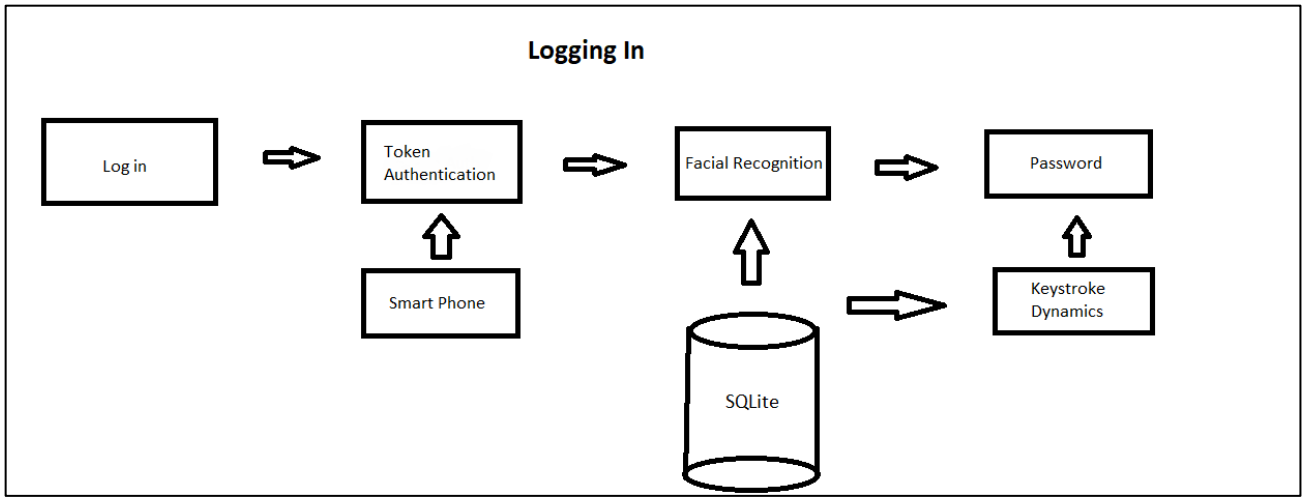


Fig 9 Login Architecture

5.2 Use Cases

Possible use cases are elaborated on in the sections below.

5.3 Sign Up

Figure 10 demonstrates the use case of when the user first pairs their smartphone with the system.

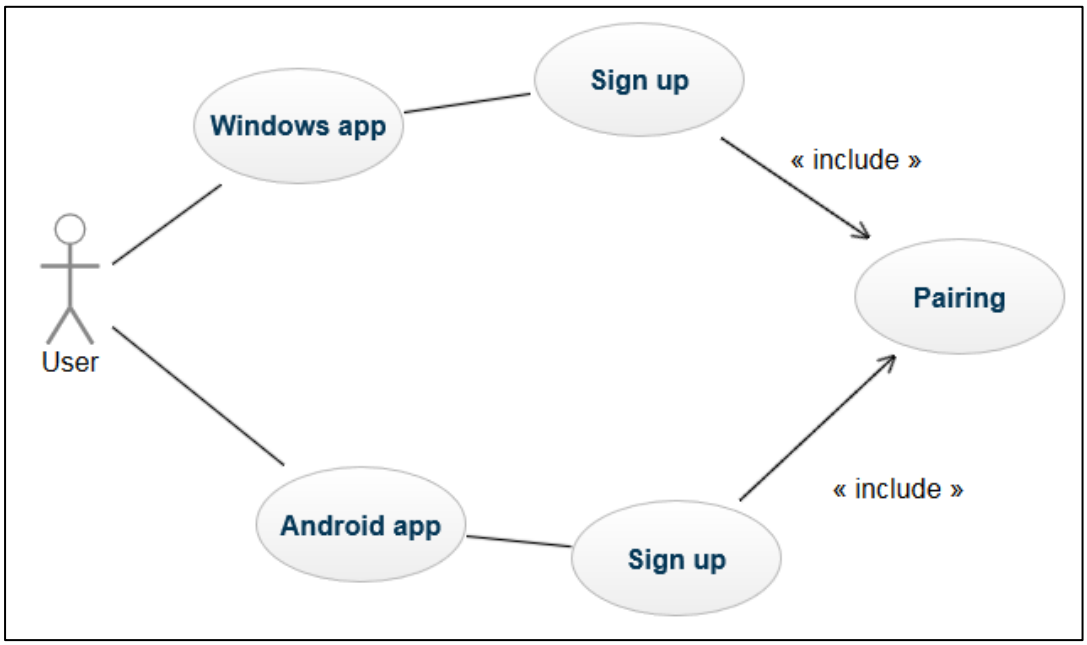


Fig. 10 Sign up use case

5.4 Facial recognition

The use case of when the user leaves the PC can be seen in figure 11. The computer waits for a period of five seconds before it locks itself in the absence of the user in sight.

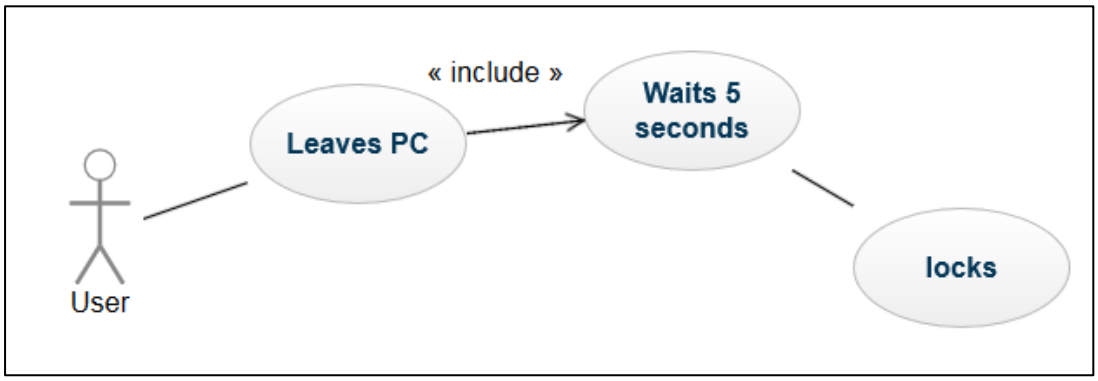


Fig. 11 Facial recognition use case

The computer automatically initiates the recognition process as the user is in sight, and unlocks automatically.

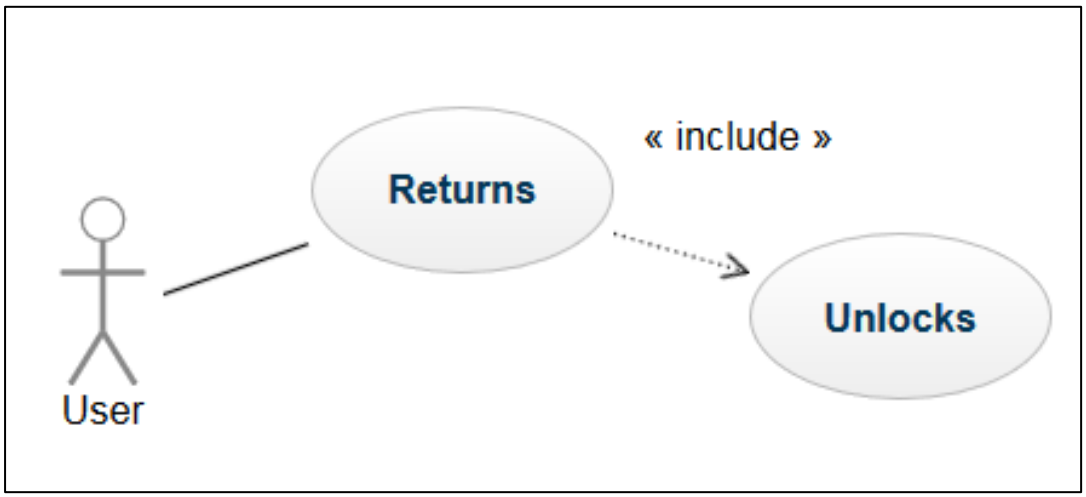


Fig. 12 User return use case

5.5 Logging In

Figure 13 displays the process for the login.

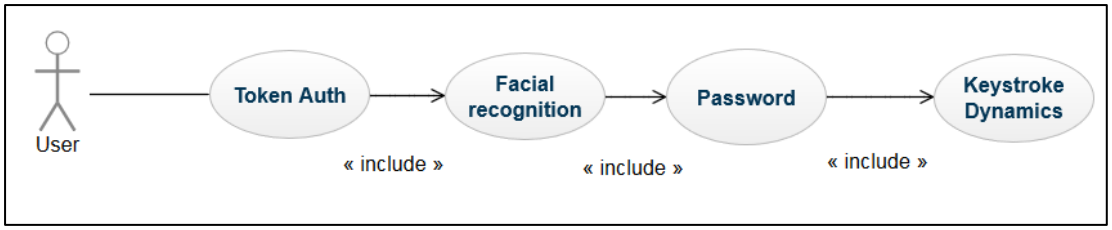


Fig. 13 Signing in use case

5.6 Switching user

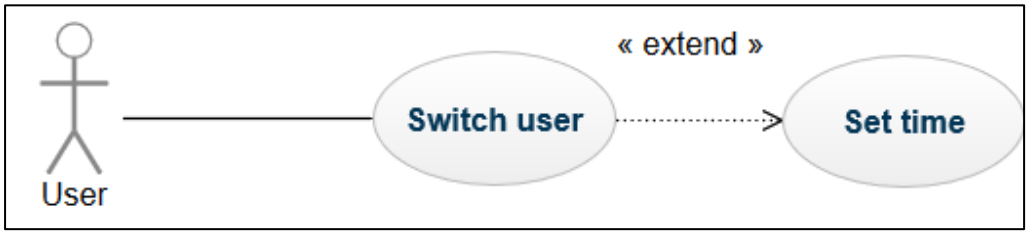


Fig. 14 Switch user use case

5.7 Lost Phone

In the event that a user loses his phone, they will have to acquire a sim card with the same number as before.

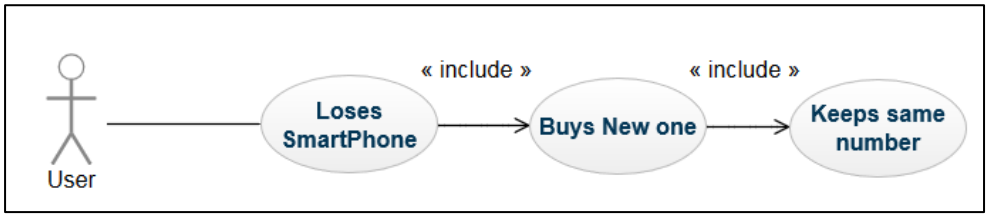


Fig. 15 User loses phones

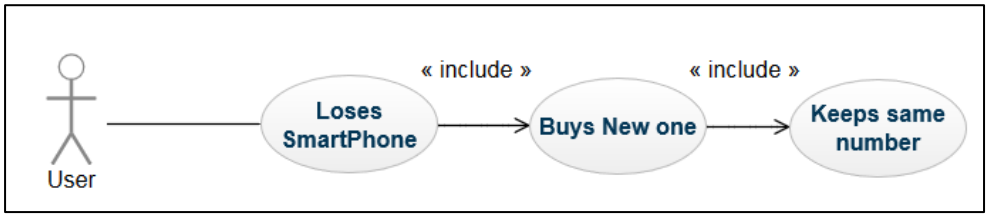


Fig. 16 User buying a new phone

Once the same number has been obtained, when the user tries to login again, they will have an option to click “Lost my phone”, the Bluetooth pairing will process again and they can login. To achieve this, the number of the smartphone must be the same as the one they signed up with.

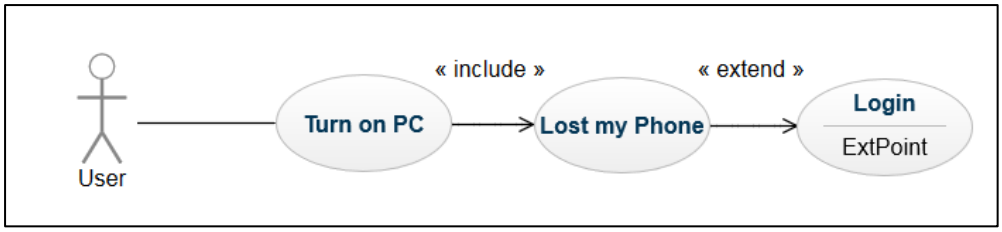


Fig. 17 Using lost my phone function

6 Architecture & Development

6.1 Introduction

This section shows the main architecture and development of the system and the challenges that came along with implementing the project. This section also has the problems that this project had and the explanation of external APIs.

6.2 Credential provider (CP)

The main component of the system is the credential provider. It allows users to login and access the system. First, the third-party application pGina was used to carry out this this task because of its simplicity. But unfortunately, not all the systems components could be fulfilled while using this application. For example, the facial recognition and keystroke dynamic factors could not be implemented due to the application's restrictions. Due to pGina not being sufficient, the credential provider had to be implemented with Microsoft's custom version. The disadvantages were that it would be more difficult and time consuming to implement but the components of the system could be fulfilled.

From Microsoft's GitHub page, developers can download the basic template for creating their own Credential Provider. In this template, there are two COM interfaces called the ICredntialProvider and the ICredentialProviderCredential. The ICredntialProvider is responsible for how each tile at the login process is remunerated, but in this system, that interface is irrelevant. The ICredentialProviderCredential is responsible for the user interaction. This interface is where the implementation of the system will be.

To add the custom credential provider to the start-up screen, the application must be built in visual studio as DLL file. This file is added to the System 32 directory. Also, a GUID number must be implemented for the custom CP, in order for it to be unique.

This number must then be added to the registry as follows:

at:HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion \Authentication\Credential Providers\.

But fortunately, there is a register file that comes alongside the CP template so it does not have to be added manually.

After the GUID number and the DLL file are added, the custom CP will now show up at the login screen. The process of how the windows authentication happens is at figure 18.

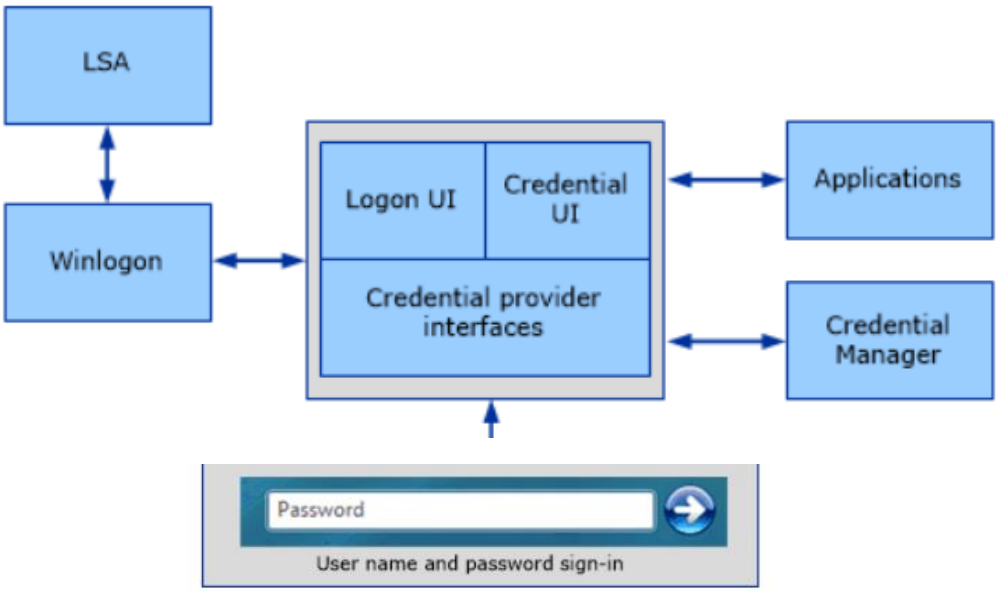


Fig. 18 Windows Credential provider layout

6.4 Token Authentication

The first factor of the login session is the token authentication with an Android smartphone. First, a Bluetooth server is created in C# using the 32feet.net Bluetooth library.

The algorithm starts with reading in the token from a file. This token is a GUID number and the smartphone, which connects to the server, would also have this number. The reason that a GUID number was chosen for the token was that is it a large number, it can be easily created and it is very unique.

Here in the “while” loop, the server is listening for the client to send the token. When it receives the data, its compares it with the token. If it is correct, the user can progress to the next stage of authentication.

```
while (true)
{
            try
      {
            byte[] received = new byte[36];
            mStream.Read(received, 0, received.Length);


          if (token[0].Equals(Encoding.ASCII.GetString(received).ToString()))
          {
            Console.WriteLine("You are signed in!");
                        Result = 0;
            break;
          }
          else
          {
            Console.WriteLine("You are NOT signed in!");
                        Result = 1;
            break;
           }
            }
        catch (IOException exception)
```

```
        {
                        Result = 1;
            break;
        }
}
```

The client side was implemented in Android Studio with Java. To begin this process, it must get the token from the server. This is done through a QR code. A QR code scanner was thus implemented implement as a button. This reads in the GUID number. If the scanner is used on the QR code that is not a GUID number. It will not register it. in order to stop accidental scanning.

In order to send the packets, to the server with the token, the user will have to provider their fingerprint. If the fingerprint sensor is not setup, the user will be asked to set it up. If the user does not have a sensor on their smartphone, they will be asked to use their PIN. If they do not have a PIN they will be asked to set it up. It goes without saying that if the user is not, it will not work.

```
if (!fingerprintManager.isHardwareDetected()) {

    mParaLabel.setText("Fingerprint Scanner not detected in Device");

} else if (ContextCompat.checkSelfPermission(this,
Manifest.permission.USE_FINGERPRINT) != PackageManager.PERMISSION_GRANTED) {

    mParaLabel.setText("Permission not granted to use Fingerprint Scanner");

} else if (!keyguardManager.isKeyguardSecure()) {

    mParaLabel.setText("Add Lock to your Phone in Settings");

} else if (!fingerprintManager.hasEnrolledFingerprints()) {

    mParaLabel.setText("You should add atleast 1 Fingerprint to use this Feature");

} else {

    mParaLabel.setText("Place your Finger on Scanner to login");
```

When the fingerprint sensor is pressed and is correct, the packets are then sent to the server. To achieve authentication, the token that the client sends must be the same as the one on the server.

In the code for sending the packet. There is a class call SendPacket(). When the fingerprint reader is pressed. A thread of this class is created and this sends the token to the server.

```
BluetoothDevice device = pairedDevices.iterator().next();
new Thread(new SendData(device)).start();
```

6.5 Keystroke Dynamics

The second and third factor of this system is the password and the keystroke dynamics, Fortunately, there does not need to be any implementation with the password because the password that is being used is the password that is registered with the Windows system. Because of this, there does not need to be any specific storing for it. The implementation of the keystroke dynamics will happen in the *KeystrokeDynamics.cpp* and *ICredentialProviderCredential.cpp* classes. In the *ICredentialProviderCredential* class there is a function called *SetStringValue().* This function runs every time the keystroke is pressed during the login session. This will be used to get the switch key time.

Here in the function, there are a double variable called duration, which gets the time from the clock. Then the *getKeystrokes()* function is called to store the key switch times.

```
// Sets the value of a field which can accept a string as a value.
// This is called on each keystroke when a user types into an edit field

HRESULT CSampleCredential::SetStringValue(DWORD dwFieldID, _In_ PCWSTR pwz)
{
        HRESULT hr;

        duration = (clock() - start) / (double)CLOCKS_PER_SEC;
        keystrokedynamics.getKeystrokes(duration);
}
```

When the user clicks enter, the times of the keystrokes are calculated and compared to what is stored about the user. This is done through the *getAverage()* function. If average times of the keystrokes are not above the percentage threshold, the user will fail to login and the object will be created again.

```
if (!keystrokedynamics.getAverage(keystrokedynamics.KeySwitchTime))
{
        keystrokedynamics = KeystrokeDynamics();
        return E_FAIL;
}
```

6.6 Facial Recognition

The facial recognition factor is used in two different stages in this system. The first stage is when the user is logging in from a powered-off system and logging in from the system being locked.

This factor was originally going to be implemented using C++ with the OpenCV library. Since this led to many technical difficulties with C++ being so strongly typed. Python then replaced it. With programming in Python, basic image processing tasks could be implemented with much ease compared to C++.

The first step of the facial recognition is first adding the user's face to the system. Going forward, it must detect the user's face. The begins with reading in the frames in grayscale.

```
image_gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

Then the classifier is read in which was created using the Viola-Jones algorithm,

```
self.classifier = cv2.CascadeClassifier(xml_path)
```

The classifier then detects where the face is and returns the face co-ordinates.

The user runs the setup and awaits approximately one second and the database gets five images of the user's face. During this process, the faces are edited then cropped into an elliptical shape for better recognition. By doing this, it disregards any background light that could distort the image. After this stage, the images are normalised. The faces are then added to a file to be stored for when the recogniser starts.

```
normalize.append(cv2.equalizeHist(image))
```

This changes the intensity of the image and makes the facial features appear more clearly. An example is in figure 21.

Fig. 19 Normalisation

(OpenCV, n.d.)

Once the recogniser starts. It first loads in the images that it had stored from the setup. It first initialises the classifier and Local Binary Patterns (LBP) algorithm:

detector = FaceDetector('haarcascade_frontalface_alt.xml')

recognizer = cv2.face.LBPHFaceRecognizer_create()

After this, the frames are loaded in to be trained and converted into LBP.

recognizer.train(images, np.array(labels)).

Now, the images from the webcam are loaded in and converted into LBP. The result is then predicted and a number of confidence is returned.

Confidence = recognizer.predict(face_img)

If the confidence if below the threshold, the recognition was a success.

The face is then loaded in to be tracked.

tr.track(frame, face)

6.7 Object Tracking

After a successful user login, the system will pass the image of the user's face into the tracking algorithm. First, then algorithm initialises the median flow tracker, then it uses three different states to track the face.

tracker = cv2.TrackerMedianFlow_create()

1. CHECKING

Check to see if there is a face present: If the length of the face is greater than zero. There must be a face to track. Then it initialises the co-ordinates and the points of interest.

```
if TrackingState == CHECKING:

    faces = userFace

    # If there is a face .. track it
    if len(faces) > 0:
        x, y, w, h = faces[0]

        # put co-ordinates in Region of interest
        TrackingROI = (x, y, w, h)


        # Changing state
        TrackingState = INIT
    else:
        return 1
```

2. INIT

Initialising the location of the face: If there is a face, initialise the tracker or go back to the checking state.

```
elif TrackingState == INIT:

    ok = tracker.init(frame, TrackingROI)
    if ok:
        TrackingState = TRACKING
    else:
        TrackingState = CHECKING
```

3. TRACKING

Here it is tracking and updating the region of interest (ROI) of the face. Then constantly updates the faces. If the face is no longer in sight, the while loop breaks.

```
elif TrackingState == TRACKING:

    ok, TrackingROI = tracker.update(frame)
    if not ok:
        break
```

7 System Validation

7.1 Testing

This section covers the testing and the performance of the authentication factors.

7.2 Facial recognition

Siblings that look similar	Pass
Identical Twin	Fail
Non-Identical Twin	Pass
Under Extreme lighting conditions	Fail

Fig. 20 Facial Recognition Testing

As seen in fig. 22. The failed attempts were identical twins. Thermal cameras or iris scans fix this issue. The reason for this is every iris is completely unique and to capture this the camera need to be very high resolution and unfortunately the camera that was used was not. Also, a thermal camera could be used to solve this issue. This shows the heat of a face and this is also unique in everyone.

7.3 Tracking

The tracking algorithm was very successful after multiple tests. When the user first authenticated into the machine, there was three people also sitting down next to the user. The tracking algorithm completely ignored the other people and when the left the camera view the computer locked instantly.

Another test was then done by running the tracker for 30 minutes. During this time a 10-minute 1080p YouTube video was watched and Facebook was browsed for 20 minutes. During this test there appeared to be no slowness from the system but the fan did get loud from time to time.

But unfortunately, it does drain the battery. I battery report was submitted by running “powercfg /batteryreport /output "C:\battery_report.html” in the command line. The tracking started on a full charged system. After 30 minutes the battery fell to 87%. While watching a 10 minute YouTube 1080p video and browsing Facebook without having the tracker on only lost 5 % battery power.

7.4 Keystroke Dynamics

The overall results for keystroke dynamics were quite mixed. An experiment was done with five people. The account owner had five attempts to enter their password for the algorithm to get the typing patterns. The first password was “everything1” and set as percentage threshold of 80%. Each user got 10 tries to spoof the password and four out of the 5 people could spoof it. Due to the password length being so small the easier it is to crack the authentication.

After, the same test was run again but with the password “instituteoftechnology”. Each user again had ten attempts and were unable to spoof it. This was due the password length being high. But unfortunately, there was a false rejection rate of 50% at a threshold of 70%.

7.5 Demonstration

In this section, the main features of the system will be demonstrated.

7.6 Token Authentication

First, this is the android app the client that connects to the server. In fig. the image on the left displays a message asking the user to login and the image of the right displays the message after it successfully scans. In fig. 23 the image of the right displays the message after a failed login attempt and the image of the left shows the use of the QR code scanner reading in the token.

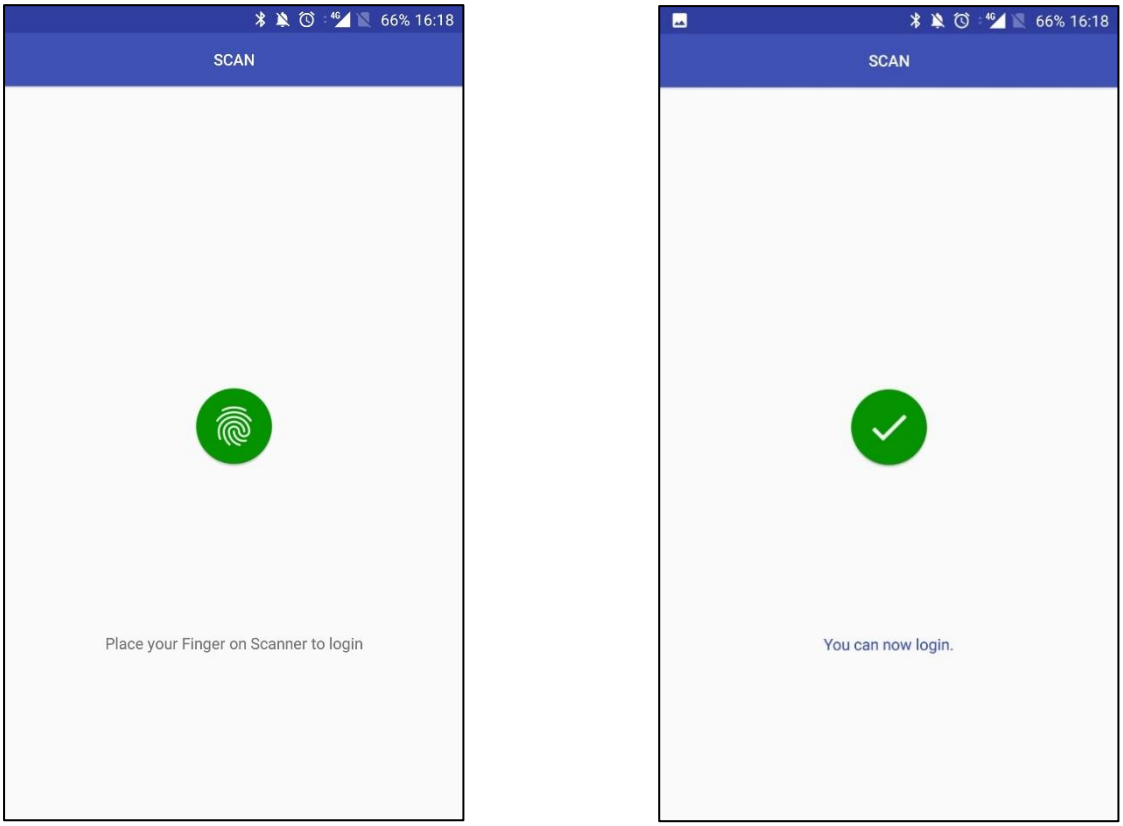


Fig. 21 Android app

Fig. 24 has the token authentication process on the server side. A console window is displaying in front of the credential provider.

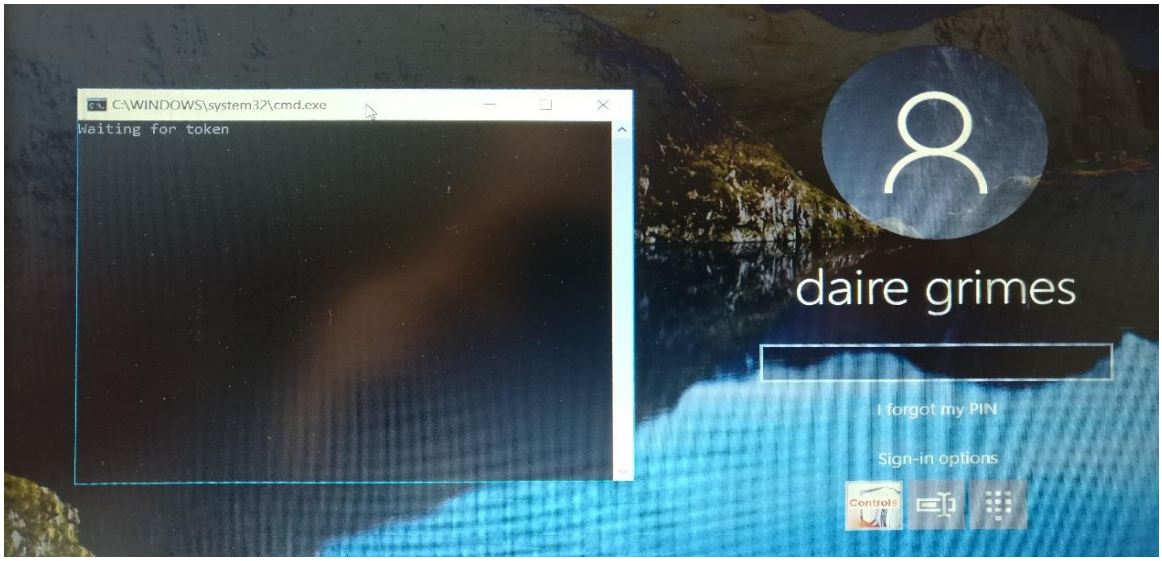
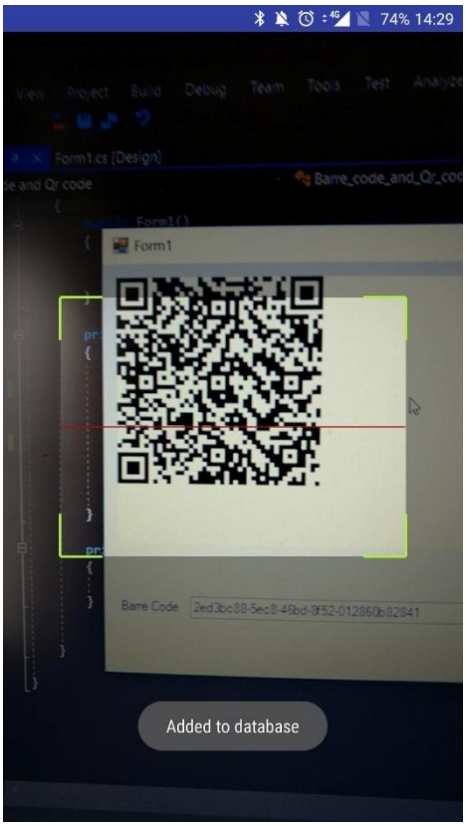


Fig. 22 Token Authentication at login



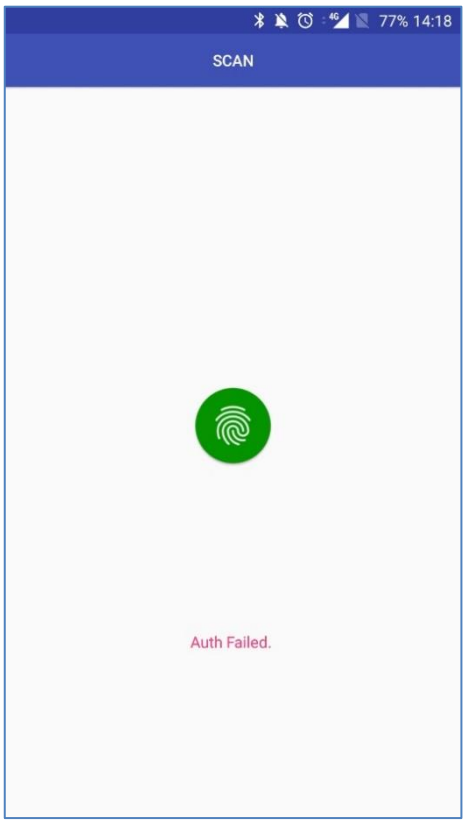


Fig. 23
Scanning a QR code and Failed fingerprint attempt

8 Conclusion

8.1 Introduction

This section gives an overview of the project, the project plan, future work, strengths and weaknesses and everything that was learnt during the implementation of the project.

8.2 Project Plan

The initial objective of this project was to create a multi-factor authentication system for a Windows 10 environment. This included multiple diverse factors such as facial and fingerprint recognition and keystroke dynamics. But due to hardware limitations, the fingerprint factor had to be dropped. Most fingerprint sensors come with their own SDKs and because of this limitation, access to the actual image of the print was not possible. To solve this issue, the fingerprint sensor was swapped for a smartphone with a fingerprint sensor. This involved creating an app on the smartphone and using it to login through token authentication via Bluetooth Low Energy. But due to information online about Bluetooth Low Energy being scarce, regular Bluetooth was used instead.

Spoof and liveness detection was also going to implemented with facial recognition through asking the user to move their face during authentication. But this involved advanced image processing skills and this could not be implemented. An alternative method that was used to solve this problem was using an infrared webcam. But due to more hardware limitations, this had to be dropped. The infrared webcam that was purchased would only run under specific circumstances and would not run with the OpenCV library. It would only run with Microsoft's built in facial recognition software and Universal Windows Platforms (UWP). This platform was created specifically for making applications for the Windows Store and accessing the infrared sensor was unavailable due to security concerns. Microsoft does not allow third-party libraries like OpenCV to use this. To fix this solution, another more expensive infrared webcam would needed to be purchased with its own SDK, but this was not possible due to time concerns. Then finally, liveness and spoof detection was going to be implemented through visible light webcams. A blink detection algorithm, was then going to be implemented but did not happen due to time constraints.

Token authentication was first going to be created in C++ for the server on the system. But due to the difficultly of creating this server in C++, the language switched to C# because it has more sufficient libraries like 32feet.net. With this library, implementing a Bluetooth server was a third the size in lines of code that in C++. Also, sending data to this server was a lot easier.

The client was created in Android Studio using java. Most of this was implemented except for storage of the token in a database. During the initial setup of the authentication, the user would read the token from the server using a QR code. The scanner was implemented on the app but the database could not be created due to shortage of time.

For Keystroke Dynamics, there were problems due to Windows 10 limitations. One of these was acquiring the hold down time of the user when the password was typed. The reason for this was not being able to access to the user input of the process. This could not happen because of security concerns. To combat this issue, a key logger was used, but it was not able to run during the authentication process due to Windows security. Another method was tried to run a C++ script on a thread during the authentication but it would not work due to the system crashing. Additionally, for the Keystroke Dynamics, the initial implementation was going to have a SVM classifier for authentication. This had to be swapped for an average-based classifier to save time.

For the Credential provider (CP) and the account lockout threshold, could also not be implemented in due time. This would have involved the user being able to choose how many times the authentication would have locked the system due to many failed attempts. If the threshold would have been reached, the user would have to then enter in an eight-digit PIN to enter the system to re-enter the system and re-do the setup process for the authentication. This too could not be completed in time.

For the facial recognition, not everything was implemented. This was due to not being able to run Python file as executables. The Python file was going to be run in the credential provider after the entering of the password. Also, for the tracking of the face. This was only partially implemented. A correction algorithm was conceived to run after the tracker lost the face. This would have happened after sudden face movements. Unfortunately, due to the short timeframe, it could not be implemented.

For data storage and encryption, an SQLite database was to be implemented. This was going to store all the user's biometrics data and would have encrypted and hashed the values. Also, the forget your phone feature was not implemented.

The final problems with the system were in the initial setup for the authentication. There was going to be a user interface where the user could easily add their credentials for authentication. But unfortunately, this also was not finished.

8.3 Project Strengths

Although, did not make it to implementation, there still is some strengths about this project. The facial recognition algorithm works successfully except for under extreme lighting conditions. But overall, the system is sufficient. Additionally, the tracking algorithm works perfectly after the recognition is completed. Immediately after it recognises the face, it transitions to the tracking state smoothly.

Adding extra factors of security to Windows is also a massive strength. Initially, it was unknown that this could be completed due to Windows being a proprietary software but after many failed attempts, there was success in completing this task. After fully understanding how Windows authentication worked. Adding the keystroke dynamics and the token authentication with the smartphone was then a was relatively straightforward.

8.4 Project Weaknesses

Unfortunately, there were some weaknesses with this system. The main one was time management. This resulted in in a lot of the projected goals not being completed as mentioned section 8.2. The reason for this was not using the time allocated for this system wisely. Too much time was spent on some parts and this resulted in other parts not being finished. For example, too much time was used in researching the Windows credential provider and facial recognition, while the storage of the user's data was not completed.

8.5 Future Work

For future work it would be important to finish these features. Mainly to add the database on the system. A regret of this project was not implementing the database sooner. Also in the future, the reliance on external libraries could be used less. Unfortunately, for the facial recognition and the Bluetooth features, extra help was needed to complete these. Finally, as more users switch from traditional personal computers to touch screen devices. A multi-factor authentication system could be implemented on a smartphone device or a tablet display. This would be something that would be every interesting for the future.

Bibliography

32.feet.NET (2017), *Personal Area Networking for .NET*
Available at: https://github.com/inthehand/32feet
Accessed: 19/11/2016

Android (2017), *Apple Vs Android—A comparative study 2017*
Available at: https://android.jlelse.eu/apple-vs-android-a-comparative-study-2017-c5799a0a1683
Accessed: 19/11/2016

Ankit Srivastava, S. M. A. S. N. S. P. B. T., 2017. *A survey of face detection algorithms,* Pune: Computer Department .

Alun Jones (2011), *Starting to build your own Credential Provider*
Available at: https://blogs.msmvps.com/alunj/2011/02/21/starting-to-build-your-own-credential-provider/
Accessed: 19/11/2016

Arwa Alsultan, K. W., 2013. *Keystroke Dynamics Authentication: A Survey of Free-text Methods,* Reading: ISSN.

Chaney, M., n.d. *Recognizing the face-value of facial recognition technology.* [Online] Available at: http://campbelllawobserver.com/recognizing-the-face-value-of-facial-recognition-technology/

Feher, C, Elovici, Y, Moskovitch, R, Rokach, L, Schclar, L,
User Identity Verification via Mouse Dynamics, Department of Information Systems Engineering, Ben-Gurion University of the Negev, Beer-Sheva, 84105, Israel

Gleason, L., 2016. *Wireless Communications in Modern Mobile Technologies.* [Online] Available at: https://dzone.com/articles/wireless-communications-in-modern-mobile-technolog [Accessed 27 02 2018].

Glenn Greenwald, E. M. a. L. P., 2013. *Edward Snowden: the whistleblower behind the NSA surveillance revelations,* Hong Kong: The Guardian.

Gupta, R. J. a. G., 1990. *Identity authentication based on keystroke latencies,* s.l.: Communications.

Hao-Yu Wu, M. R. E. S. J. G. F. D. W. F., 2012. *Eulerian Video Magnification for Revealing Subtle Changes in the World,* s.l.: s.n.

Herh, M., 2016 . *Non-contact Fingerprint Recognition.* [Online] Available at: http://www.businesskorea.co.kr/english/news/money/16238-non-contact-fingerprint-recognition-contactless-fingerprint-recognition-tech-drives

Klitsie, M., 2015. *Eulerian motion magnification during robot-assisted surgery*, TWENTE: s.n.

Marasco, E, Ross, A, 2015, *A Survey on Anti-Spoofing Schemes for Fingerprint Recognition Systems*, West Virginia University, Michigan State University

Mallick, S., 2017. *Object Tracking using OpenCV (C++/Python).* [Online] Available at: https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/ [Accessed 15 03 2018].

Mariusz Rybnik, P. P. K. S., 2009. *User Authentication with Keystroke Dynamics using Fixed Text,* Bialystok: IEEE.

Mejda Chihaoui *, A. E. W. B. a. C. B. A., 2016. *A Survey of 2D Face Recognition Techniques,* Tunisia: University of Sfax.

Manpreet Bagga, D. B. S., 2016. *Spoofing Detection In Face Recognition : A Review,* Belagavi: IEEE.

methodologies, I. f. r. A. c. r. o., 2014. *Reza Shoja Ghiass, Ognjen Arandjelovi, Abdelhakim Bendada, Xavier Maldague,* Université Laval: ELSEVIER.

Microsoft, 2016. *Credential Provider Technical Reference.* [Online] Available at: https://www.microsoft.com/en-us/download/details.aspx?id=53556

Microsoft, n.d. *Credential Providers in Windows 10.* [Online] Available at: https://msdn.microsoft.com/enus/library/windows/desktop/mt158211(v=vs.85).aspx

M. Rybnik, M. T. a. K. S., 2008. *Keystroke Dynamics Based System for User Identification,* s.l.: s.n.

Mariusz Rybnik, P. P. K. S., 2009. *User Authentication with Keystroke Dynamics using Fixed Text,* Bialystok: IEEE.

Mukesh D. Rinwa, B., 2015. *Face Recognition System,* s.l.: International Journal of Advanced Research in Computer Engineering & Technology.

Neves, P. F. d., n.d. *Multifactor Authentication using Smartphone as Token,* Lisboa: s.n.

OpenCV, n.d. *Histograms - 2: Histogram Equalization.* [Online] Available at: https://docs.opencv.org/3.1.0/d5/daf/tutorial_py_histogram_equalization.html [Accessed 03 01 18].

OpenCV (2017), About Available at: https://opencv.org/about.html Accessed: 19/11/2016

PGina (2014), *How pGina Works* Available at: https://github.com/pgina/pgina/wiki/How-pGina-Works Accessed: 19/11/2016

Phaetto (2017), *Windows Credential Provider Available* at: https://github.com/phaetto/windows-credentials-provider Accessed: 19/11/2016

Pin Shen Teh, 1. A. B. J. T. a. S. Y., 2013. *A Survey of Keystroke Dynamics Biometrics*, Lincoln: Hindawi Publishing Corporation.

P. Jonathon Phillips, P. J. F. K. W. B. R. W. V. B. J. G. G. W. Q. M. P., 2011. *Distinguishing Identical Twins by Face Recognition*, Santa Barbara: IEEE.

Prototype (2016), What is Prototype model- advantages, disadvantages and when to use it? Available at: http://istqbexamcertification.com/what-is-prototype-model-advantages-disadvantages-and-when-to-use-it/ Accessed: 19/11/2016

Python (2016), Why learn Python?
Available at: http://www.bestprogramminglanguagefor.me/why-learn-python
Accessed: 19/11/2016

Reza Shoja Ghiass, O. A. H. B. a. X. M., 2013. *Infrared Face Recognition: A Literature Review*, Dallas: IEEE.

Salil P. Banerjee, D. L. W., 2012. *Biometric Authentication and Identification using Keystroke Dynamics: A Survey*, Clemson University: CiteSeerX.

Satran, M., 2017. *Windows Unlock with Windows Hello companion (IoT) devices.* [Online] Available at: https://docs.microsoft.com/en-us/windows/uwp/security/companion-device-unlock[Accessed 13 02 18].

Sasikumar., P. K. a. M., 2010. *Recognising Emotions from Keyboard Stroke Pa*, s.l.: Inter-national Journal of Computer Applications.

shopeemart, n.d. *WHAT IS A PROTOTYPE MODEL?.* [Online]
Available at: https://shopeemart.weebly.com/sdlc.html
[Accessed 15 11 17].

Statista (2014), Number of smartphone users worldwide from 2014 to 2020 (in billions)
Available at: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/
Accessed: 19/11/2016

Subrat Kumar Rath, S. S. R., 2014. *A Survey on Face Detection and Recognition* , Odisha: School of Computer Engineering, KIIT .

Warwick, A. A. K., 2013. *Keystroke Dynamics Authentication: A Survey of Free-text* , Reading: s.n.

SQLite (2017), About
Available at: https://www.sqlite.org/about.html
Accessed: 19/11/2016

Warwick, A. A. K., 2013. *Keystroke Dynamics Authentication: A Survey of Free-text* , Reading: s.n.

Yu, Z. J. a. T., 2011. *On Mouse Dynamics as a Behavioral Biometric for Authentication,* Raleigh: s.n.