# Possible Attacks on Match-In-Database Fingerprint Authentication

Jadyn Sondrol
sondr037@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

## Abstract

Biometrics are used to help keep users' data private. There are many different biometric systems, all dealing with a unique attribute of a user, such as fingerprint, face, retina, iris and voice recognition. Fingerprint biometric systems, specifically match-in-database, have universally become the most implemented biometric system. To make these systems more secure, threat models are used to identify potential attacks and ways to mitigate them. This paper introduces a threat model for match-in-database fingerprint authentication systems. It also describes some of the most frequent attacks these systems come across and some possible mitigation efforts that can be adapted to keep the systems more secure.

***Keywords:*** biometrics, match-in-database, template, minutia, FAR, FRR, threat model, spoofing attack, denial-of-service attack, replay attack, trojan horse attack

## 1 Introduction

According to the Global Markets Insights, the global biometric market is anticipated to exceed 70 billion dollars by the year 2027[1]. Since biometrics are used in banking companies, law enforcement, and in the vast majority of mobile phones, about 80% of US citizens have used some kind of biometric technology. *Biometrics* help to secure a user's information by recognizing that user via one or more of their unique features, such as fingerprint, retina, or facial scans. Biometrics use an individual's unique attribute, and because of that there is a new risk of that attribute being stolen and used elsewhere. For example, creating an artificial copy of one's fingerprint would allow an intruder to access any information on one's phone. That is why biometric systems need to be very secure.

This paper discusses security of fingerprint-based authentication. In section 2 we describe all of the modules of match-in-database fingerprint authentication and the processes behind it. Section 3 introduces a threat model that is used to find a system's susceptibility to any attacks or breaches. Finally, section 4 details what the most common attacks are and explains the mitigation methods that should be implemented to prevent these attacks. Because match-in-database fingerprint authentication is widely used, there is a need for a threat model. This threat model is used to describe all of

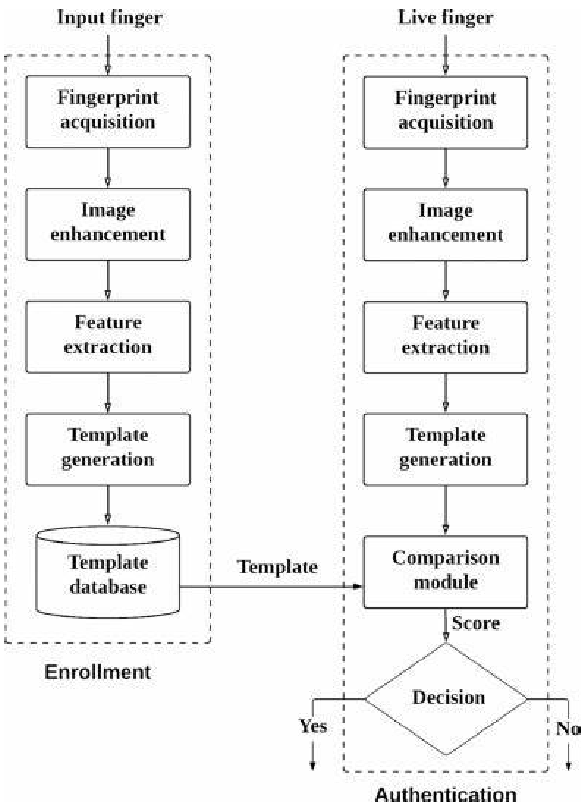

**Figure 1.** Match-in-database biometric system model. The input finger describes the enrollment phase and the live finger describes the authentication phase [5]

the most frequent attacks made on this system, categorize them by threat level, and provide some mitigation techniques. All of the attacks listed (Spoofing, Denial-of-Service, Replay, and Trojan Horse) can be prevented by numerous mitigation efforts which are implemented into the current systems.

## 2 Fingerprint Authentication

Biometric systems can be generally classified into two categories, match-in-database and match-on-device systems. The match-in-database, or MiD, fingerprint authentication system is the most popular biometric system. As shown in Figure 1, *MiD* systems use a remote database to store a user's encrypted template. A *template* is a digital portrayal of a

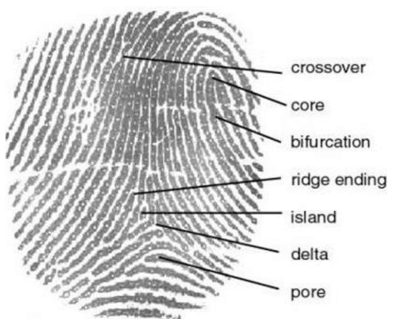

**Figure 2.** Characteristic features of a fingerprint [7]

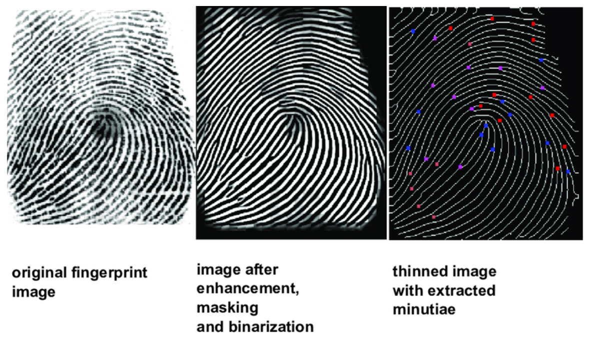
original fingerprint image
image after enhancement, masking and binarization
thinned image with extracted minutiae

**Figure 3.** Data stored in minutiae based templates before encryption [9]

fingerprint. Then when a user scans their finger, the feature enhancement module extracts distinct patterns in the fingerprint to generate another template. The template is sent to the remote database where it is compared to other stored encrypted templates. This is unlike the match-on-device systems where the templates are compared on the device itself.

Fingerprint authentication occurs in two distinct phases. The first is the enrollment phase, where an individual and their fingerprint are first introduced to the system, as shown in Figure 1 under the input finger. This is when a user puts their finger on a sensor located on an input device. Various techniques of image enhancement are then used to improve the image's quality.

Image enhancement is used because both environmental and physical factors can decrease the quality of the image. The physical factors may include sweat, a finger wound, the amount of force applied to the scanner, etc. Environmental factors like temperature, humidity, and the durability of the scanner also play a role in the quality of the image.

The characteristic features of a fingerprint, such as the ridges (dark lines), and valleys (white spaces), are unique to every individual in the world. Fingerprints are made up of lakes, ridge bifurcations, ridge endings, etc. as shown in Figure 2, but what it really comes down to is the minutiae. A *minutia* is a small characteristic feature of a fingerprint used for authentication. Most commonly a bifurcation, the point where the ridge ends and it splits in two or bifurcates, is used [4, 5]. The feature extraction module then finds the minutiae from the fingerprint and highlights them, as shown in Figure 3 on the right, so they can be used later in the comparison module.

A template is a recording of the minutiae and is consistent between readings of the fingerprint. For example, in the minutiae based approach, the coordinates of all the bifurcations are stored in a template. A template is then generated and encoded (turned into binary strings) for transition into the template database, where it is stored until authentication or identification occurs.

The authentication phase is when the user is verified or identified, as shown in Figure 1 under the live finger. Fingerprint verification is checking if a user is who they say they are, meaning to verify that the fingerprint matches the registered user they claim to be. Fingerprint identification is matching the live finger template to a template stored in the input finger template database. This means that the template is compared to all of the other templates in the remote database [2]. However in practical uses the templates can be sorted in various different ways in order to lessen the amount of comparisons.

There are three different ways to store the data in the template to use for comparison. The first is the *minutiae based* approach. This approach stores the x and y coordinates of all the minutiae, the direction of the minutiae angle, and the minutiae type, which is usually a bifurcation. The *pattern based* approach preserves the pattern of the ridge structure, meaning whether the fingerprint is an arch, loop, whorl, etc. The last is the *image based* approach. It stores the unaltered impression of the fingerprint from the scanner. The most commonly used approach is minutiae based.

For the minutiae based approach the image of the fingerprint is binarized (converted into black and white) by tracing the ridges of the finger. This image is then thinned so that the each ridge is exactly one pixel wide [5, 6]. The minutiae locations are extracted, then encoded (converted into binary strings), and stored in the template to be used for comparison.

The comparison module then checks to see if the live finger template matches with a template in the database. A match is perceived if the similarity score surpasses a certain threshold. This threshold differs from system to system depending on level of security necessary. The set threshold is based on the *false accept rate (FAR)* and the *false reject rate (FRR)*.

The false accept rate states the chance that the system will accept an intruder or non-user. The FAR is calculated by taking the number of non-users that had a similarity score over the threshold divided by the number of true negatives

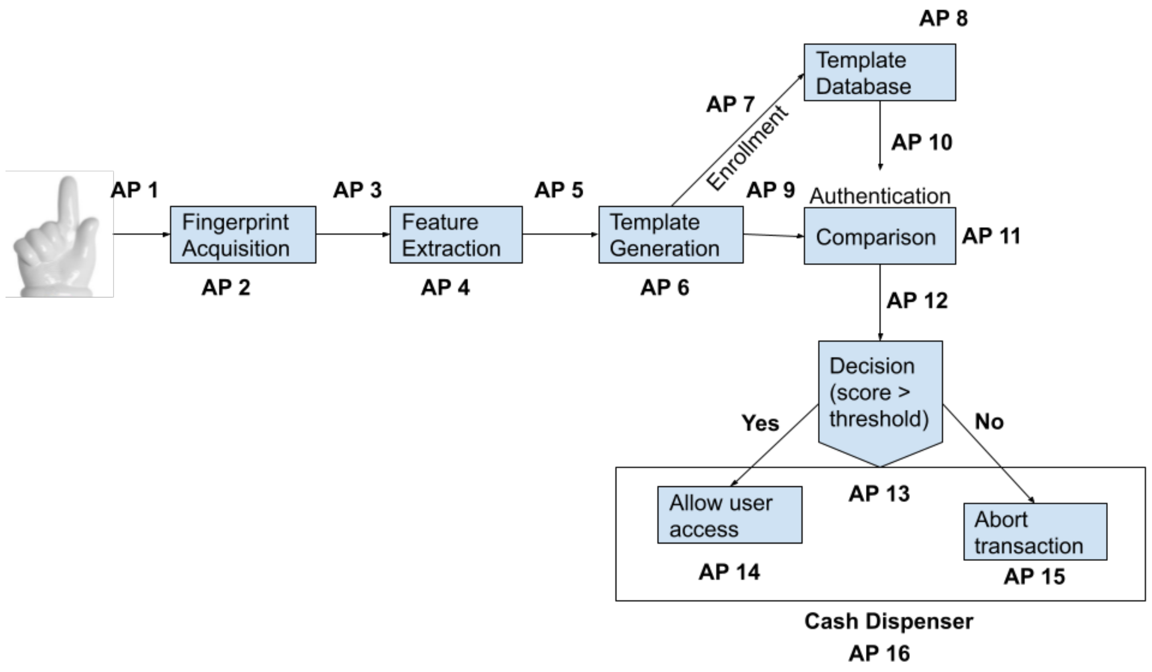

**Figure 4.** Modified threat model of match-in-database fingerprint authentication [5]

and false positives added together. True negatives are the amount of non-users that were not allowed into the system, and false positives were the amount of non-users that were allowed access into the system.

The false reject rate determines the probability that a system will deny a registered user. The FRR is calculated by taking the number of registered users that had a similarity score under the threshold divided by the total number of true positives and false negatives added together. True positives are the amount of users who were able to get into the system and false negatives are the amount of users who were not able to get into the system.

Each system's threshold differs as the level of security differs. A high degree of security would cause a high FRR and a low FAR. Conversely a lower degree of security would cause a low FRR and a high FAR. Both the FAR and FRR are measured on a scale from 0 to 1. There is some give and take between this relationship. Having a high FAR would guarantee that no false users could enter, but in turn it would also not allow many real users to gain access either. Similarly having a very low FRR would allow all registered users to get in, but would also let many non-users in as well.

## 3 Threat Model

*Threat models* are used to show where and how a system is vulnerable. To create a threat model the biometric system is examined step by step in order to find its susceptibility to any attacks or breaches.

The first step is to diagram the system being designed. Using this design, potential weak spots in the system can be discovered. These weak spots are where potential attacks could occur within the system. Specific attacks will be matched to these attack points in which they could occur. These attacks will be categorized by threat level, either high, moderate, or low, depending on the ease of the attack. The threat model will also list any mitigation methods that would prevent these attacks from being successful.

These models are best created continuously as it can help detect problems before they actually occur. Overall, threat models are systematic and structured so that they can detect threats in the software development cycle. There are many different threat models for fingerprint authentication. The one we are going to describe was proposed by Joshi, Mazumdar, Dey. [5] and then modified slightly for simplicity.

### 3.1 Proposed model

This threat model shows 16 probable attack points (labelled AP) at 7 components. In Figure 4, the lines between the modules show the communication channels where the information is sent from one module to another. The boxes represent the components of the system. The components match the steps of the match-in-database system model (Figure 1) with the attack points added. The proposed model is very thorough and shows 16 probable attack points with over 30 different attacks described. In this paper we will describe a few of the most common attacks and how to try to mitigate them.

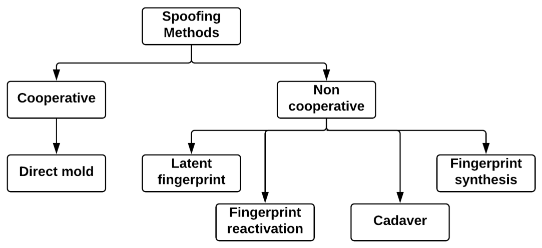

**Figure 5.** Spoofing methods used for creating fabricated fingerprints [5]

## 4 Types of Attacks

Biometric attacks are classified into direct and indirect attacks. Direct attacks describe the attacks to the scanner or cash dispenser itself. In a direct attack an intruder may try to introduce a forged biometric template to the input sensor. This attack requires no knowledge of the biometric system itself. However indirect attacks involve gaining access to a component of the system and providing forged data at some internal step of authentication. These attackers need to have precise information (about the modules themselves) in order to execute an indirect attack.

### 4.1 Spoofing attack

In the first attack point from Figure 4, a spoofing attack could occur. In a *spoofing attack* is a direct attack in which an attacker would present false biometric data while claiming to be an enrolled user. Spoofing may be the simplest way to gain access to the system. An attacker doesn't need to know any information about any of the encryption or comparison components within the system. There are various ways to create a fabricated fingerprint, as shown in Figure 5.

In the cooperative method, an enrolled user helps to produce a fabricated fingerprint. The enrolled user shares an imprint of their own fingerprint with the intruder. Then the intruder uses this imprint to manufacture a direct mold using a type of plaster. This direct mold will allow anyone to enter the system.

The non-cooperative method is more difficult as the intruder must first get the imprint of an enrolled user on their own. This is most frequently achieved by the intruder gathering an impression of a user from a different place such as a door, table, etc.

A *latent fingerprint* is an invisible impression of a fingerprint made by the oil secretion of a user's fingers. An attacker can revive a latent fingerprint from a user using cyanoacrylate glue or powder in order to make the imprint visible. By using powder and a brush someone can get the impression of the fingerprint and then transfer it to a clear substance, such as tape, to get the imprint. The attacker could then place the tape on their finger and present it to the scanner to gain entry.

*Fingerprint reactivation* techniques take the impression of a latent fingerprint from the surface of the sensor and then revive it so that it can be used again. These techniques include putting a water filled plastic bag, sweeping with graphite powder, or even breathing over the input scanner to produce a usable fingerprint to resubmit into the system. Then the attacker would add some pressure to the scanner, such as a clothed finger, to allow the sensor to read the print.

An intruder can create an artificial fingerprint by using a dead fingerprint of a *cadaver*. This would only be possible if an enrolled user had died. *Fingerprint Synthesis* attempts to reconstruct a fingerprint from a stored template. This would require access to the template database in order to acquire a stored template. A hill-climbing attack could assist this technique. An attacker would need to know the similarity score and then input a fingerprint. The image would then be modified slightly over and over until it achieves a reasonable similarity score and is accepted into the system.

Although anti-spoof methods can be either hardware or software based, the hardware implementation can be costly due to needing new hardware components for the input scanner. These new additions will allow the system to use fingerprint properties such as skin resistance, temperature, or electrical conductivity, in order to determine if a fingerprint is real or fake. Adding these new parts create a new problem: confidential information may be revealed through patterns of electromagnetic waves or power consumption. Therefore the software-based approach is more preferred. Software-based approaches can be either static, using only one snapshot or dynamic, using multiple snapshots.

The static method is pore-based, perspiration-based or uses texture properties. There are many unique pore-based techniques, such as determining the quantity of pores on a finger, using the distance and spacing between the pores, or seeing the pore distribution. A fake fingerprint is claimed to have fewer pores due to the different falsification steps. High resolution is needed to accurately find the pores within a fingerprint [3].

The perspiration-based technique uses the fact that sweat is unevenly dispersed along the ridges of a real fingerprint. Many spoofed fingerprints show a high level of conformity throughout the print. A falsified print generally does not contain any sweat pores since they are very small and hard to include. This means the shading of the print is less varied in a fake print. However, these features can be difficult to read because it depends on the pressure applied by the finger to the scanner.

Texture properties include the form, smoothness and position of the fingerprint. Variables like ridge strength and ridge continuity are calculated to determine if a fingerprint is real. A spoofed finger's texture is far more coarse than a real finger and can be measured by using a technique of multi-resolution texture analysis as well as inter-ridge frequency analysis [8]. Each of the techniques alone do not work at a consistent rate so they are most commonly used together.

The dynamic method captures multiple snapshots of the fingerprint within 2-5 seconds of the finger being on the scanner. These snapshots are analyzed to acquire dynamic features such as perspiration and ridge distortion. Perspiration is caused by sweat pores on one's finger and produces a scan with darker sections near the pores that have given off sweat recently. Therefore the separate snapshots would contain different shading around the pores. A spoofed finger's snapshots would produce nearly the same image over and over.

Ridge distortion is another dynamic method. Using multiple snapshots, a real finger would move slightly or apply a different amount of pressure on the sensor. This deformity is analyzed by examining the snapshots at a faster rate per frame while the finger is moving.

The risk factor for spoofing attacks is high because of the ease of creating an artificial fingerprint. Also many commercial devices do not implement anti-spoofing techniques, and so are very hard to detect.

## 4.2 Denial-of-Service Attack

Attack point 2 from Figure 4, shows a possible attack on the input device itself. This is a direct attack since the intruder doesn't need to know anything about the makeup of the system itself and is attempting to damage the physical biometric scanner. In a *denial-of-service attack* a malicious individual causes a device to be unavailable to the intended users by electronically sending numerous requests, essentially overloading it. If the scanner can't take a fingerprint as an input, the application is unable to let a user into the system.

A denial-of-service attack can also occur at attack point 16, from Figure 4. Bombarding the biometric controlled application with countless authentication requests will prevent the system from being able to grant access to any enrolled user. The system would be forced to begin exception handling, making the biometric system unusable.

To combat this, many systems that use biometrics tend to keep an eye on the scanner using a video camera or security guards, so that they can observe any attempts by a user or an intruder. They can also use rugged devices, which are devices that perform the same as the consumer device but are also designed to withstand any unusual conditions, to prevent the system from being unusable.

The Denial-of-Service attack has a low threat level due to the fact that the intruder can't get into the system and therefore is unable to steal any of the information within the system. The attack simply forces the system to be inaccessible and is more of an annoyance if anything. There is no valuable information an attacker can get from this attack, so attackers are less likely to attempt this kind of attack on the input device. However, companies still want to try to prevent this since it causes their system to be unusable.

## 4.3 Replay Attack

Replay attacks could occur on the input of the feature extraction module (attack point 3) from Figure 4. *Replay attacks* maliciously use a valid data input from a previous user by repeating or delaying the attempt. A non-enrolled user will watch (on the system itself) a user input their finger on a scanner and is able to repeat or delay this input so that they can enter into the system. This is an indirect attack due to the input of the forged data to the feature extraction module.

A liveliness test can also be added to the scanner, then the result will be forwarded onto the next module. The next module will check if the data came from the input scanner. This test checks to see if a user is real or fake by prompting the user or using an algorithm to check. An active liveliness test will simply ask the user to follow a line on the scanner, e.g. swipe across the screen. Fingerprint action is very hard to replicate as real fingers show uneven pressure within the print. A passive liveliness test uses various algorithms in order to see if is has been falsified. Similar to the static anti-spoof methods, these algorithms will look at the pore distribution, examine the texture and inspect the amount of ridge distortion within a fingerprint [8].

Attack point 9, from Figure 4 is also susceptible to a replay attack. Attack point 9 describes the communication channel between the template generation and the comparison modules. The intruder introduces an intercepted fingerprint template of a user into the channel. The template would then be read as input in the comparison module.

A challenge/response method can be applied to the system in order to mitigate this attack. This method examines all of the modules within the system as one transaction. This way a query system coming from a secured transaction sever can be used. Once a user places their finger on the scanner, a query is sent from the server. For example, the query could be a random image pixel value that is sent to the input scanner and the feature extraction module. The transaction server will compare the two pixels, if the are the same the process will continue. Otherwise the process will be aborted as an attempt to bypass the modules has been detected.

This helps mitigate a replay attack because the intruder will have access to the pixel value from the previous user's input. However the server will ask for a different pixel's value for every attempt. So the attacker will have to send the previous pixel value to the server, therefore the pixels will not match and the process will be terminated.

Another way to try and mitigate this attack is to add a global clock to each module. Adding this clock would allow each component to put a timestamp on all outgoing traffic. Then the next module can verify the timestamp from the incoming data to make sure that the information from the previous module is accurate. If the verification fails, the process will stop and throw away the current data.

Implementing a one-time password reduces the chance of a replay attack being successful. A one-time password is usually a set of numbers generated by a computer to add an additional layer of security. This password will be sent to your phone, tablet or email at the time of a log in attempt. Then you will be prompted to type in the password to the system itself. Making a user type in this password every time they sign in does not allow an attacker to re-use a fingerprint without knowing this password as well.

The replay attack has a moderate threat level due to the fact that this attack is to the communication channels between modules. It is challenging for an attacker to gain access to the channel but if they do, all of the information being protected would be accessible.

### 4.4 Trojan Horse Attack

A Trojan Horse attack could occur on attack point 4 from Figure 4, also known as the feature extraction module. A *Trojan horse attack* is an indirect attack, where an imposter embeds malware onto the system. It can be activated right away or it can lay dormant in the system until it is activated. Once the Trojan is triggered it can modify, copy or delete any information from the feature extraction component. This Trojan can create a premade feature set and send it into the communication channel which is then sent to the template database.

This type of attack can also transpire on attack point 11, which is the comparison module. This module looks at two fingerprint templates, one from the database and one from the live finger. It produces a similarity score by comparing the templates. The Trojan can attack either a template or the similarity score in order to get into the system. Either of the templates can be modified in order for the fingerprints to be similar enough. Or the similarity score threshold could be lowered so that almost any fingerprint would be sufficiently alike.

To mitigate this attack a trusted biometric system (TBS) can be implemented. A TBS uses mutual authentication, which means that each component makes sure that the input information really came from the previous module. Therefore if both ends of the communication channel are verified, then no information within the channel could be altered in any way. This works in both directions in order to secure the channel. If one or both of the modules are not verified, the process will be stopped and the user will have to start the authentication process all over again. Implementing this does increase the computation power and the time it takes, but it also severely decreases the chance a Trojan horse attack would be successful.

Another way to prevent a Trojan horse attack would be to add code signing or new hardware components. Code signing would be implemented into every component, so that every output would be signed by the previous module. The next module will first check if their input is signed by the past component. If it has been signed the process will continue, otherwise the attempt will be discarded. Specialized tamper-resistant hardware could be applied to the communication channels to prevent the modification of information. However the hardware addition is very costly and therefore less popular.

This is a moderate threat level because once a Trojan is embedded into the system it is somewhat easy to get into the software and modify it. However the difficult part is getting the Trojan into the input device. Since the input scanner is external and usually in a public setting, outside eyes, such as a security guard, may be present to stop this addition to the system.

## 5 Conclusion

Biometrics are used all over the world to protect our private information from any malicious individual. Currently, match-in-database fingerprint authentication is the most widely used biometric system due to the quick and accurate verification process. Because of its popularity, the need for a threat model is essential. This threat model is used to describe all of the most frequent attacks made on this system, categorize them by threat level, and provide some mitigation techniques.

The attacks listed in this paper were from various attack points from Figure 4. The spoofing attack uses various techniques to create an artificial fingerprint and introduce it to the biometric sensor. The denial-of-service attack simply makes either the input scanner or the biometric controlled application inaccessible for any valid user. A replay attacker uses a repeated or delayed authentication attempt in order to gain access to the system. The Trojan horse is embedded into the system and can be activated at a later time. This Trojan can delete, copy, or modify template data so that the similarity score is above the set threshold and the attacker can gain entry.

All of these attacks can be prevented by numerous mitigation methods which are implemented into the current systems. Overall, lots of research has been done on biometric systems and they are found to be secure. This is why many companies will be incorporating more biometric systems within their devices to keep users' information private.

## Acknowledgments

I would like to thank my advisor, Elena Machkasova, and Dylan Cramer for all the help with the research and development process in writing this paper.

## References

[1] 2022. *Biometrics: Definition, use cases, latest news.* https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/inspired/biometrics

[2] Saad Bin Ahmed, Muhammad Imran Razzak, and Bandar Alhaqbani. 2016. The Minutiae Based Latent Fingerprint Recognition System. In *Proceedings of the International Conference on Internet of Things and Cloud Computing* (Cambridge, United Kingdom) *(ICC '16)*. Association for Computing Machinery, New York, NY, USA, Article 49, 9 pages. https://doi.org/10.1145/2896387.2896434

[3] Heeseung Choi, Raechoong Kang, Kyungtaek Choi, and Jaihie Kim. 2007. Aliveness Detection of Fingerprints using Multiple Static Features. *World Academy of Science, Engineering and Technology* 2 (01 2007).

[4] Markus Dürmuth, David Oswald, and Niklas Pastewka. 2016. Side-Channel Attacks on Fingerprint Matching Algorithms. In *Proceedings of the 6th International Workshop on Trustworthy Embedded Devices* (Vienna, Austria) *(TrustED '16)*. Association for Computing Machinery, New York, NY, USA, 3–13. https://doi.org/10.1145/2995289.2995294

[5] Mahesh Joshi, Bodhisatwa Mazumdar, and Somnath Dey. 2020. A comprehensive security analysis of match-in-database fingerprint biometric system. *Pattern Recognition Letters* 138 (2020), 247–266. https://doi.org/10.1016/j.patrec.2020.07.024

[6] K. K. H. Karunathilake, A. R. M. Shahan, M. N. M. Shamry, M. W. D. S. De Silva, Amila Senarathne, and Kanishka Yapa. 2021. A steganography-based fingerprint authentication mechanism to counter fake physical biometrics and trojan horse attacks. In *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 0286–0292. https://doi.org/10.1109/IEMCON53756.2021.9623240

[7] Esraa Naamha and Abdul Monem Rahma. 2017. *Fingerprint Identification and Verification System Based on Extraction of Unique ID*. Ph. D. Dissertation.

[8] Ram Prakash Sharma and Somnath Dey. 2019. Fingerprint Liveness Detection Using Local Quality Features. *Vis. Comput.* 35, 10 (oct 2019), 1393–1410. https://doi.org/10.1007/s00371-018-01618-x

[9] Wioletta Wójtowicz. 2014. A Fingerprint-Based Digital Images Watermarking for Identity Authentication. *Annales UMCS, Informatica* 14 (10 2014). https://doi.org/10.2478/umcsinfo-2014-0008