University of Cambridge

Master's Thesis

User Authentication for Pico

Author:
Cristian M. Toader

Supervisor:
Doctor Frank Stajano

A thesis submitted in fulfilment of the requirements for the degree of 'Master of Philosophy'

in the

Computer Security Group
Computing Department

April 2014

Abstract

The abstract needs to be written at the end.

Acknowledgements

The acknowledgements and the people to thank go here, don't forget to include your project advisor…

Contents

Abstract i

Acknowledgements ii

Contents iii

List of Figures v

List of Tables vi

Abbreviations vii

Symbols viii

1 Introduction 1
- 1.1 Contribution . . . 2
- 1.2 Prerequisites . . . 3

2 Pico: no more passwords! 4

3 Assessment framework 7
- 3.1 Framework for evaluating web authentication schemes . . . 7
- 3.2 Token based authentication framework . . . 8

4 Design 13
- 4.1 Proposed design . . . 13
- 4.2 Framework evaluation . . . 16
- 4.3 Conceptual design threat Model . . . 19
  - 4.3.1 Unified device . . . 20
  - 4.3.2 Dedicated device with shared components . . . 21
  - 4.3.3 Insecure communication with Pico . . . 23
  - 4.3.4 Shared device with shared components . . . 24
  - 4.3.5 Proposed secure implementation . . . 24

5 Implementation Prototype 26
- 5.1 Authenticator design . . . 26
- 5.2 Threat model . . . 27
- 5.3 Implementation details . . . 27

5.3.1 Main application and services 27
5.3.2 Owner configuration 29
5.3.3 Voice recognition library 29
5.3.4 Face recognition library 29
5.4 Results 29

A Appendix Title Here **30**

List of Figures

List of Tables

Abbreviations

LAH **L**ist **A**bbreviations **H**ere

Symbols

a	distance	m
P	power	W (Js^{-1})
ω	angular frequency	rads^{-1}

Chapter 1

Introduction

As shown by papers such as [] written by Adams & Sasse passwords have become increasingly difficult to manage. They are meant to authenticate an user based on a shared known secret. Although in theory as well as in the past this has worked well, in practice they are no longer a viable scalable solution.

Increasing computation power has made passwords easier to brute force. In order to avoid this, authentication systems started enforcing rules such as a minimum number of characters, having at least 1 numeric character, or at least one special character. Furthermore the number of accounts which require passwords has significantly increased since they first became widely adopted. In order to maintain security and not have the exposure of a password affect multiple accounts, each password should be unique. Furthermore, security experts recommend (and system admins sometimes enforce) that passwords are to be changed regularly with something that is not too similar to the previous one.

All these security restrictions and recommendations are in place in order to make the mechanism theoretically secure. There is no focus on the user which is meant to memorise numerous unique complex passwords. As shown in paper [] most of the times users ignore recommendations that are not enforced by the authentication system or even worse, they write the passwords down and compromise security for usability.

The Pico project [] by Stajano was designed to make passwords obsolete. It is a hardware token which keeps and generates user authentication credentials. This unburdens the user from the dreaded “something you know” transforming it into “something you have”.

This makes the solution scalable with the number of accounts, as well as secure, since the user may no longer choose “weak” passwords.

What differentiates Pico from other token based authentication mechanisms is its locking and unlocking mechanism. Without introducing the need for a PIN or any other known secret, Pico only becomes available in the presence of its owner. This adds security to the threat model when the device is not in the possession of its rightful owner. Currently Pico relies on the concept of Picosiblings [] which are small in the presence of which Pico becomes unlocked.

Although Picosiblings are a sensible solution to unlocking Pico, they purely based on proximity to the device. The purpose of this project is to create a new way of authenticating the user to the Pico device. Pico is defined as a device which unlocks and locks automatically only in the presence of the owner. This is a relatively open concept. How would the device detect presence? The following chapters will reveal the proposed solution.

1.1 Contribution

The project work is strongly related to the Pico project designed by Stajano []. The following contributions have been made in order to provide an alternative to the Picosiblings user authentication:

- We created a framework for assessing token based authentication mechanisms.
- We designed a new way of authenticating the Pico to the user and compare the proposed solution with alternatives.
- We developed an Android prototype which functions as a platform which allows further contributions.
- We analyse the Pico claims according to the framework developed by Bonneau et al [] and determine the impact of the solution on the Pico.

1.2 Prerequisites

Although the focus of the project is creating an alternative concept for the Picosiblings solution, the prototype was developed using the Android Java SDK. Additional knowledge regarding biometric authentication mechanisms and signal processing would also be useful but implementation details are outside the scope of this project.

Chapter 2

Pico: no more passwords!

The project has strong connections with the Pico project. Therefore, the following sections aim to go into brief detail as to what Pico is, how it works, and what its properties are. T

In the paper "Pico: no more passwords!" [**?**], Frank Stajano describes an alternative design to passwords. It is based on a hardware token called "Pico", which replaces all instances where passwords could be used. The reason for designing this alternative authentication mechanism is that passwords are not as usable and secure as they used to be.

Computation power has increased significantly since passwords became popular. This makes passwords easier to break using either brute-force or dictionary attacks. As a result, restrictions were imposed to make passwords stronger, such as requiring at least an upper case letter or a numeric character. Furthermore, in order to reduce the threat of dictionary attacks, authentications systems recommend, if not enforce, passwords to be harder to guess, meaning they should not contain dictionary words. Furthermore, after imposing all these restrictions, it is also recommended to change them regularly. This rule is sometimes enforced in the case of larger companies.

The changes previously described are reasonable from a security point of view, but completely lack user focus. With an increasing number accounts requiring authentication, whether web based or local, passwords become increasingly difficult to remember. Reusing them is not recommended, as compromising one account would lead to multi-account exposure. This leads to having users remember a large number of unique passwords, which is not realistic in practice. Therefore security compromises are made by the user such as writing passwords down, or reusing them, therefore breaking the theoretical security of the password.

The token based alternative designed by Frank Stajano improves the security offered by passwords. It transforms something you know in something you have. This makes the authentication memory effortless. Unlike other hardware tokens, Pico is theft resistant thanks to small devices called Picosiblings. The Pico uses wireless communication with the Picosiblings and becomes unlocked only in their presence. It is also responsible for generating authentication credentials therefore ensuring that they are not weak or reused.

The Pico device uses for authentication purposes two buttons and a camera. The buttons are used for pairing and creating a new account. The authentication process is multi channel. It relies on a visual code created by the app in order to transmit its public key and unique id. The rest of the authentication process is performed over radio communication via nonce challenges to prove that the other participant has the private key corresponding to the public. The authentication protocol is described in detail in paper [**?**].

Authentication between the Pico device and its owner is performed using small devices called Picosiblings. These are meant to be embedded in everyday items that the owner carries around, such as earrings, rings, keys, chains, etc. Communication with the Picosiblings uses short range radio. The initialisation protocol for Picosiblings is called the resurecting ducklings [**?**]. Further enquiries are performed using picosibling pings once every predefined time interval. This tests whether the picosiblings are within the range of the Pico.

The communication between Pico and its Picosiblings is meant to reconstruct the Pico master key. Each Picosibling has a k-out-of-n secret used by the Pico device to reconstruct its master key. Pico tracks each Picosibling based on a timer. If the timer expires and the Picosibling is unable to generate a response, the secret is deleted.

There are two special shares with a longer time out period: a biometric authentication mechanism, and a remote server communication. These are used for special cases such as having your Picosiblings and Pico both stolen. The biometric authentication would only allow the new user to have limited access time, while the remote server communication can be used to lock Pico remotely.

Chapter 3

Assessment framework

The purpose of this chapter is to create a framework for assessing token based authentication mechanisms. Using this framework we can then compare existing solutions. Having the framework as a performance compass we can then continue by designing an alternative to the Picosiblings unlocking mechanism.

3.1 Framework for evaluating web authentication schemes

In the paper "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes" [?] the authors develop a framework for evaluating web based authentication mechanisms. The purpose of the framework is to identify authentication schemes which outperform passwords. The framework is intended to provide a benchmark for future web authentication proposals.

The framework focuses of three classes of properties which are abbreviated as UDS: usability, deployability, and security. Each class contains a set of properties, totalling a number of 25 benefits. A mechanism may either offer, quasi-offer, or not offer a property. Properties which are not applicable to a mechanism are marked as "offered" to simplify the framework.

Using the framework to evaluate 35 password replacement schemes shows that no scheme is dominant over passwords. According to the evaluation, passwords score perfectly in deployability. They score reasonably in terms of usability, excelling in properties such as: nothing-to-carry, efficient-to-use, and easy-recovery-from-loss. In terms of security however, passwords don't perform as well, only receiving points in resilience-to-theft (not applicable), no-trusted-third-party, requiring-explicit-consent, and unlinkable. The full list of properties and their description can be found within the paper itself.

Biometric mechanisms receive mixed scores on usability. None of them have the infrequent-errors property which is a precision problem related to false negatives. More importantly if the biometric data is exposed by malware for instance, the authentication mechanism may not be used by the user any more. They score poorly in deployability due to the additional hardware required. In terms of security they perform worse than passwords. Replay attacks can be used by an attacker using a recording of some sort in order to trick the sensor. They are not resilient to theft, since they require an additional device. The fact that they uniquely link the owner to the recording means that the owner may be linked back to the data, therefore not granting the "unlinkable" property.

The paper notes that the memory-effortless property versus nothing-to-carry is only achieved by biometric schemes. None of the mechanisms manage to fully achieve memory-effortless and be resilient-to-theft. This is due to the fact that most mechanisms replace something you know with something you have.

The authors do not produce aggregate scores or rankings. This is due to the fact that not all properties are equal in importance, but different properties would have different weights depending on the scheme's application domain.

Combining schemes is mentioned as a two factor arrangement. This would result in a mechanism which in terms of usability and deployability would only have the properties which are granted by both schemes. In security however it would have the properties of both mechanisms. As shown in the paper, according to [**?**] the presence of a two factor authentication would lead the user to creating weaker passwords.

3.2 Token based authentication framework

Web based authentication mechanism is initiated locally and performed locally. In contrast, token based authentication is initiated and performed locally, leaving no room for man in the middle attacks or any other 3rd party participation. For this reason, a subset of the properties described in paper [?] by Bonneau et al should also be present in the framework we have developed.

The following list shows what properties of the framework developed by Bonneau et al are relevant to token based authentication mechanisms:

Memory-effortless

Different types of tokens would have different results. For instance the RSA SecurID [] token doesn't require any authentication, while the FIDO (Fast IDentity Online) Alliance [] may use a PIN requiring a known secret.

Easy-to-learn

Token authentication mechanisms may have different learning curves. As an example a CAP reader is fairly easy to use, while a Pico device may prove more difficult for the inexperienced.

Efficient-to-use

Time required by the token user authentication mechanism may differ from one type of authentication to the other. The time required for registering a new user or unlocking the token for its owner should be reasonably short.

Infrequent-errors

The token unlocking mechanism may reject true positives. If the number of false negatives is reasonably low, then the mechanism has this property.

Easy-recovery-from-loss

The user's ability to get another token which uses the same authentication mechanism. Tokens which unlock using biometrics for instance, if not properly secured may lead to the user's inability to use that mechanism again.

Accessible

The ability for all users to use the authentication mechanism. As an example, PINs may be entered by any user regardless of disabilities, on the other hand other biometric mechanisms may not be available.

Negligible-cost-per-user

The total cost enquired by the user in order to use the authentication mechanism.

Mature

It refers to the number of users that have successfully used the mechanism, open source projects based on the mechanism, and any other usage by a third party which did not participate in the development of the scheme.

Non-proprietary

Anyone can implement the token unlocking scheme without having to pay royalties to anyone else.

Resilient-to-physical-observation

An attacker should not be able to impersonate the user after observing them authenticate.

Resilient-to-targeted-impersonation

An attacker should not be able to impersonate the user using knowledge about the user, or previous recordings of his biometrics.

Resilient-to-throttled-guessing

The resilience to an attacker automating a guessing process in order to brute force the unlock of the token.

Resilient-to-unthrottled-guessing

An attacker which only physical access to the token cannot guess the required unlocking resource.

Resilient-to-internal-observation

An attacker cannot tamper with the token in order to intercept user input. Furthermore it is impossible for the attacker to gather the input from within the token's storage.

Requiring-explicit-consent

The authentication mechanism requires explicit consent from the user in order to become unlocked.

Unlinkable

The unlocking mechanism does not generate data which if leaked would compromise the identity of the user. . . .

The properties described above are derived from the original framework presented by [**?**]. Additional details relevant to each property, including when a property is only quasi (partially) satisfied may be found in the original work by Bonneau et al. Some properties from the original work, such as Nothing-to-carry or Server-compatible, do not apply for token unlocking schemes and therefore are not included.

Although the framework by Bonneau et al provides a good base set of properties, a few others are needed in order to fully characterise token unlocking mechanisms. The following list is part of the project's contributions to the overall evaluation framework.

Continuous-authentication

The concept, although mentioned, is omitted from the framework developed by Bonneau et al [**?**]. It stressed a bit more as part of the benefits of Pico by Stajano [**?**]. This is a property belonging to the security category of the original framework. The property is satisfied if, once authenticated, the user remains authenticated to the token for as long as he is in its presence. This is similar to an authentication session with the added property that the session remains active for as long as the user requires it. The property should be satisfied by mechanisms which may re-perform authentication in a non explicit way, leaving the user unaware of the underlying process.

Multi-purpose-unlocking

This is a security category property. If satisfied, the unlocking mechanism may allow for multiple types of unlocking based on the user confidence or identity. This is something that may be characteristic for mechanisms which involve biometrics or accounts with multiple security levels.

Availability

This is an usability property. If satisfied, the user has the ability to use the unlocking mechanism in any circumstance. As an example gait recognition would only function while moving on foot, or a recognizer that restricts access based on pulse would not satisfy this requirement. A mechanism requiring only a PIN on the other hand would work in any circumstance.

In order to demonstrate how the framework works we have assessed a number of token based authentication mechanisms, including the original Picosiblings solution. Results are shown in figure **??**.

At the end of this peace of work a new framework for the evaluation of token unlocking mechanisms was developed. Existing properties have been identified from the literature and added to the framework together with original work. An initial evaluation was made for existing token unlocking mechanisms which will serve as a benchmark for the proposed solution.

Chapter 4

Design

4.1 Proposed design

The design proposed by Stajano [?] requires Pico's locking and unlocking mechanism to be performed without the use of any password mechanisms. Furthermore support needs to be added for continuous authentication. What this means is that Pico needs to detect the presence of the user in a non obtrusive way.

Given these properties, my idea is to combine multiple continuous authentication mechanisms in order to compute a confidence level. If the confidence level is satisfactory then the Pico is notified that the owner is present and becomes unlocked. This would provide a non-obtrusive continuous authentication mechanism that would suit the Pico requirements for satisfying its claims.

Since the process needs to be continuous and non-obtrusive, most of the mechanisms rely on biometric data such as iris recognition, face recognition, voice recognition, gait recognition, and others. Another source of data would be the user's location compared with previous history, and other patterns that give with a certain confidence whether the owner is in the presence of the Pico.

The approach offered by this project is different from simply stating that Pico is using biometric data as an unlock mechanism. The novelty in this design is based on how the data is combined in order to compute the owner confidence level. Each is assigned a different weight based on the level of trust it offers in identifying the user. This doesn't necessarily need to be the precision of the mechanism but it would be a good indicator for choosing a value.

Just as the Picosiblings solution, the Pico may have a wide range of supported biometric inputs. However, not all data is always available or relevant. As an example gait recognition would only work while the user is travelling on foot. Other biometrics such as iris recognition may not always be available based on how the sensors are integrated and carried by the user. For this reason all mechanisms will have a decaying confidence level which decreases in time from the last successful recording.

Let us take for example voice recognition which would be sampled every minute. The current weight of the mechanism is 0 so its output is completely ignored. The next sample is recorded showing a confidence of 70% that the owner is present. Upon this recording, the mechanism weight is updated to it's original value. For the next 10 minutes the owner will be silently reading. Since the mechanism does not manage to identify any voice present, the weight of the mechanism decays. The overall result will be impacted less by the voice recognition mechanism up to the point that the recording will be so old, it will no longer be taken into account, or a new successful identification will be performed.

Each mechanism outputs a probability that the recorded data represents the owner of the token. Upon each recording, this probability is updated using Bayes' Law. This process is also known as a Bayesian update. The equation is described below:

$$P(H|E) = \frac{P(H) * P(E|H)}{P(E)}$$

In the equation above:

- $P(H|E)$ represents the probability of hypothesis H after observing evidence E; this is also known as the posterior probability.
- $P(H)$ represents the probability of hypothesis H before observing evidence E; this is also known as the prior probability.
- $P(E|H)$ represents the probability that the current evidence belongs to hypothesis H.

- $P(E)$ is the model evidence, and has a constant value for all hypothesis.

We will use the "Law of total probability" in order to compute the value of $P(E)$ in order to accurately compute the posterior probability. The formula is the following:

$$P(E) = \sum_n P(E|H_n) * P(H_n)$$

Using this the Bayes' Law equation becomes:

$$P(H|E) = \frac{P(H) * P(E|H)}{\sum_n P(E|H_n) * P(H_n)}$$

Our model however, contains only two hypothesis: the recording of the data either belongs to the user, or not. We can therefore consider $P(H)$ to be the hypothesis that the data belongs to the owner and $P(\ H)$ that the data belongs to someone else. Obviously the value of $P(\ H)$ is $1 - P(H)$ and $P(E|\ H) = 1 - P(E|H)$ Using this information, the rule for updating mechanism probability that the recording belongs to the user becomes:

$$P(H|E) = \frac{P(H) * P(E|H)}{P(H) * P(E|H) + P(\ H) * P(E|\ H)}$$

Now that we have shown how mechanism probability is calculated, and know that each mechanism has a decaying weight based on the last recording time we can continue to calculate the overall confidence of the Pico. This is performed quite trivially using a weighted sum. The following equation shows the process:

$$P_{Total} = \frac{\sum_{i=1}^{n}(w_i * P_i(H|E_i))}{\sum_{i=1}^{n} w_i}$$

The result is then compared with the minimum threshold required by Pico. If the requirement is satisfied, the user is granted access for the current app authentication. Due to the continuous authentication property, the Pico token will continue to ask its authenticator whether the confidence level is still satisfied. Based on the decay rate of the weights and the input data available of the authenticator's mechanisms this will constantly be recalculated.

At some point the confidence level required by Pico might be too high for the authenticator to grant access. As an example the owner will want to access it's bank account after being silent in a dark room for the past hour. Let us say this would require a confidence level of 95%, while the authenticator may only output a 20% confidence that the user is still present. Given the circumstances, an explicit authentication mechanism may be required from the user in order to increase the current confidence level.

Combining explicit authentication with the current design can be performed consistently with the continuous authentication mechanisms. Whenever an explicit authentication is required, the only difference will be the fact that the user becomes aware of the authentication process. They are prompted to pass an authentication challenge (i.e. facial recognition, voice recognition). This would guarantee valid input for the authenticator which may then proceed to compute an accurate score.

4.2 Framework evaluation

We will continue by evaluating the new proposed scheme with the token unlocking framework defined in the previous chapter.

Memory-effortless: Quasi-satisfied

TODO: ask about this!

Easy-to-learn: Satisfied

In order to satisfy Pico's property of continuous authentication, all mechanisms part of the scheme I developed also need to have this property. Therefore the authentication process is non-transparent to the user, and therefore there is nothing to learn.

Efficient-to-use: Satisfied

The authentication data is collected either at fixed time intervals, or is fired during special events. The authentication process however, does not fully depend on recent data. A response may be generated without any recent authentication data. Therefore the time spent by the mechanism to generate a response is immediate.

Infrequent-errors: Quasi-satisfied

Given that the scheme depends on biometric mechanisms, the quality of the errors is as good as the underlying biometrics. If the scheme cannot generate a high enough confidence an explicit biometric challenge will be issued for the user to satisfy. Since the original biometric mechanisms do not have this property, to some extent neither will the scheme I have designed. However, the scheme is combining multiple biometrics results with different score weights based on importance and accuracy. This is much more likely to be accurate, which is why I will mark this as Quasi-satisfied. For a more accurate response, the design needs testing with a high quality prototype.

Easy-recovery-from-loss: Not-satisfied

Token based mechanisms in general do not have this property due to the inconvenience of replacing the token. In our case, the property is also not satisfied. The user would have to re-acquire a new token and reconfigure the owner's biometric data. Furthermore based on the mechanism, such as location settings or gait recognition, the token is likely to require an adaptation period.

Accessible: Satisfied

Due to the fact that the scheme is based on multiple biometrics and location settings, I consider this property to be Satisfied or as a very least Quasi-satisfied. The scheme functions based on available biometrics, without having any predefined solutions. It is highly unlikely that the owner cannot generate any of the available biometric inputs, especially for some such as "face recognition".

Negligible-cost-per-user: Quasi-satisfied

This property depends on the way in which the scheme is implemented. If the implementation is based on high quality sensors embedded in items of clothing and such, then the property is not satisfied. If the implementation reuses sensors that the user already possesses, the the property is fully satisfied as the cost is 0. An example of such an implementation would be an Android application/service possibly using the future Google Glass hardware.

Mature: Not satisfied

This property is not satisfied as the project is at the level of a work in progress prototype. The design is quite fresh and was not implemented by any third party. Neither was is reviewed by the open source community or has had any user feedback.

Non-proprietary: Satisfied

Anyone can implement the scheme without any restrictions such as royalty checks or any other sort of payment to anyone else.

Resilient-to-physical-observation: Satisfied

Since the mechanism is based on biometric data, simple observations from an attacker cannot lead to compromising the user's authentication to the token. The attacker would have no way of reproducing the input through simple observation.

Resilient-to-targeted-impersonation: Quasi-satisfied

Saying that the scheme Quasi-satisfies this property is a bit generous. Each of the mechanisms is vulnerable to a replay attack. An attacker may record one of the user's biometric and replay it as a token input. However, given that the token uses multiple mechanisms, some of which being location based, this is a highly unlikely occurrence. The only vulnerable point would be the explicit authentication mechanisms, which carry a lot of weight.

Resilient-to-throttled-guessing: Satisfied

The amount of throttled guessing required for the user to break one of the biometric mechanisms is far too large for this to actually be a threat.

Resilient-to-unthrottled-guessing: Satisfied

Given that the Resilient-to-throttled-guessing property is satisfied, this property is also satisfied.

Resilient-to-internal-observation: Satisfied

This property does not apply to this scheme.

Requiring-explicit-consent: Quasi-satisfied

The user may opt out from the scheme being active. However, in order for the Pico continuous authentication property to be available, non explicit periodic authentication needs to be performed based on biometric data. This should be viewed as the user consenting for authentication when deciding to use the device. Since there is some uncertainty regarding what explicit consent means in this case, only the property is marked as only Quasi-satisfied.

Unlinkable: Not-satisfied

Just as any of the biometric mechanisms, this property is not satisfied by the mechanism. The authentication data maps uniquely to the owner of the token.

Continuous-authentication: Satisfied

The mechanism was designed with continuous authentication in mind. Data is collected periodically with a confidence weight decaying over time. This allows for the token to be used at any time based on current existing data. The only exception breaking the model would be the explicit authentication mechanisms, but these could only be triggered at the beginning of an authentication process using the token.

Multi-purpose-unlocking: Satisfied

This property is fully satisfied by the authentication mechanism. It allows the token to grant access to different authentication accounts based on the precomputed level of confidence that the owner is present.

Availability: Satisfied

Some mechanisms are not always available even though enabled, especially due to the continuous authentication property. As an example gait recognition while sitting in an office. However, the scheme may use a multitude of mechanisms with the unlikeness that all of them are unavailable. For instance location history may predict with a certain confidence that the owner still in possession of the token. This propery is aided by the explicit authentication mechanism which requires explicit input from the user.

Comparing this score with the other token unlocking mechanisms... TODO!

4.3 Conceptual design threat Model

An accurate threat model on the proposed unlock mechanism must start by analysing the set of assumptions made about the mechanism. From there we can identify available threats and how the scheme can be exploited in order to unlock the Pico without owner permission. Throughout the threat model we will explain how relaxing the initial set of assumptions may change the security outcome. Each model is analysed from an Availability, Integrity, and Confidentiality.

It is important to note that confidentiality is an important category in this evaluation. This is because the device will store sensitive biometric data which is directly linkable to the user. Losing this data, especially in plain-text, would disable the user from ever using the biometric device for which the data was leaked. This is due to the fact that the leaked data could always be replayed, successfully tricking the biometric mechanism.

In each subsection, the model will obviously only introduce issues with the mechanism. Therefore when reading a subsection, the issues are not only those currently presented, but also those from previous subsections that lead up to that point.

4.3.1 Unified device

We will start from the assumption that the unlock mechanism is integrated on the same device with the Pico. The device is assumed to be dedicated and runs no other software. Furthermore, the set of available sensors will also be integrated within the device. Alternatively there may also be peripheral sensors, with no way for an attacker to tamper with the communication to the authenticator.

Availability

From an availability point of view, an outside attacker cannot create a denial of service scenario. Interactions with the device are performed physically, so therefore the device cannot be made unavailable while in the possession of its owner. If the Pico would temporarily lose ownership, from a software perspective it would lock up due to mismatching biometric and location data, but would become available again in the presence of the owner.

Only hardware modification would affect data availability. Simply disconnecting the sensor would not affect the scheme's ability to generate viable results due to the fact that multiple biometrics are used. However an attacker could modify a sensor to output wrong data, tricking it into saying the user is never the owner. This would create a successful denial of service attack path where a few sensors output that the owner is never present.

Integrity

Communication paths are not accessible from the outside and therefore cannot be tampered with in order to modify data. Furthermore the device is not running any other software and is therefore safe from any malware attacks.

Only physical tampering with the device would change data integrity. Modifying one of the sensor's and changing its output to some random data would be undetectable by the mechanism.

Confidentiality

No software access as well as no communication with the outside (i.e. wired communication) means that data is safe as long as the device is with its owner.

If the device were to be lost, Storage data should be kept encrypted, similar to the way Ironkey [] protects its data. Unfortunately an attack path may already be identified which is due to the fact that using this model the decryption key needs to be stored on the device. An attacker which has hardware access could therefore extract the key and decode the data. The original Picosiblings solution circumvented this approach by keeping

4.3.2 Dedicated device with shared components

We will relax the original set of assumptions by saying that the communication path with the sensors is no longer secure. Furthermore the sensors may be shared with other owners, via a wireless communication link for example. Another feasible scenario is that although sensors are located on the same device as the Pico, the Pico application is fully compartmentalised from the outside world.

What we are trying to stress with this scenario is that the sensors are no longer part of a trusted secure box, but are outside and communication with them, as well as their input may no longer be secure.

Availability

Since the sensors are no longer dedicated, other users may access sensor data. Depending on the hardware and software platform supporting the sensors, this may lead to a denial of service attack on the scheme. For example, if the sensors may only have one owner at a time, an attacker may request data from all sensors keeping them locked from the biometric authentication mechanisms. If the system is built in such a way, then there is nothing the scheme could do to prevent this other than keep the sensors constantly locked for itself. However since the model is built on the concept of shared sensors, this might not be a feasible solution.

Furthermore, communication paths are no longer dedicated. Weather the communication channel is radio or pure software, this introduces a new attack path. A "man in the middle" type of attack may be performed where information data from the sensors is dropped and replaced with bad data. This would create a scenario similar to the one in the previous section, but without the need for physically modifying the sensors.

Integrity

Having shared communication paths with the sensors means that data integrity may be compromised from outside. This goal would be achieved in the previous model only by physically modifying the sensors. Furthermore if the sensors are on the same device as the Pico, malware may modify output data leading to unsuccessful mechanism authentication.

Since Pico and the authenticating mechanism are fully compartmentalised from the outside, their communication is still secure. This compartmentalisation however needs to include all types of storage and communication.

Confidentiality

Unfortunately having shared sensors introduces quite a big confidentiality issue. Given that the sensor data required for authentication is shared, nothing would stop an attacker from collecting just as the Pico unlocking mechanism would. This data could then be replayed to the authenticator in order to unlock the Pico.

This is quite a critical issue. An example of feasible attack pattern would be. A peace of malware analyses when the sensors are locked, and makes assumptions as to when the Pico authenticator is locking them. Based on these assumptions the malware then captures sensor data immediately after the lock was released therefore capturing a possibly valid sample of data.

A more elaborate peace of malware could detect patterns such as time intervals or events that trigger sensor locking. Knowing these patterns it could therefore lock the sensors and gather data just before the Pico authenticator would, and then trick the authenticator by sending it a replay or possibly modified data.

Yet another scenario in these circumstances would be to send the Pico authenticator constant bad data and anticipate the trigger of an explicit authentication request to the user. By locking the sensors at that key time the peace of malware could acquire a high quality data sample. Since most of the mechanisms used by the scheme are biometrics, that data sample would represent permanent damage to the user, as an authentication mechanism using that type of biometric could be replayed in any circumstance.

Since the Pico unlocking mechanism is fully compartmentalised, access its storage is secure and therefore any stored credentials are fully protected.

4.3.3 Insecure communication with Pico

This is a special case model which assumes that Pico and the authenticator we have developed are communicating over an insecure channel. The only thing this case needs to consider is the communication between the two participants.

Availability

Integrity

Confidentiality

4.3.4 Shared device with shared components

We will relax the model even more in order to better fit reality constraints when implementing the mechanism. In this model, Pico and its authentication mechanism reside in a computing model with shared storage resources. The security of Pico and its authenticator may only be as good as the underlying OS. In order to have a meaningful use-case scenario.

Availability

Integrity

Confidentiality

4.3.5 Proposed secure implementation

A secure proposed implementation is viable using an Android telephone running a TrustZone enabled ARM processor available in ARMv6KZ [] and later models. This device would essentially be divided into two "worlds": the normal world running the untrusted Android OS, and the trusted world running a small operating system written for TrustZone. Both operating systems are booted at power up. In addition the TrustZone OS loads a public/private key pair which is inaccessible from Android.

Ideally Pico would be implemented with its authenticator within TrustZone. This would essentially guarantee complete separation from a memory perspective leaving any sort of malware attack impossible via memory.

Persistent memory is however required in order to store data for each individual biometric mechanism used in the authentication scheme. Unfortunately this type of memory is not protected by the TrustZone OS and constitutes a way for a third party to attack the scheme. However, we could use the TrustZone OS key pair in order to encrypt biometric data on disk. Even though this data is available from Android it would be fully confidential. If properly stored within Android, the OS may even protect its integrity from outside attacks.

Let us consider however that the Android OS has been completely compromised by the attacker and is therefore “hostile”. Under these circumstances data confidentiality can still be fully guaranteed. The TrustZone public key could still be used in order to encrypt the biometric data before writing it to disk. Attacks from a memory perspective may only be performed by modifying data stored on disk. This may only lead to a denial of service for the owner, but not a confidentiality breach.

Let us briefly discuss any issues using the availability-integrity-confidentiality framework.

Availability

Only plausible attacks are denial of service through deleting biometric cache files from disk. This would require constant reconfiguration for the Pico scheme, making the Pico unavailable.

Integrity

Data integrity may only be altered from cache files on disk.

Confidentiality

No known attacks on data confidentiality other than capturing sensor data just as the authenticator would. However this would be possible with or without the Pico being present.

Chapter 5

Implementation Prototype

Thus far we have developed a new Pico authentication scheme and assessed it using our own token unlocking framework. We then have performed a threat model from an availability, integrity, and confidentiality perspective and have suggested the safest implementation which would be as feasible as possible for the user to adopt.

In this chapter we will described design and implementation details for the prototype of the proposed scheme. The implementation platform will be the Android OS, which uses a Java based SDK for application development.

5.1 Authenticator design

The user authenticator for Android is designed to work as a bound service called UAService. Periodically the service outputs to registered Pico clients the status of the authentication process. Any application may be a client as long as it registers with the service. Furthermore, explicit authentication update requests may be performed by the Pico client.

Since different authentication mechanisms require different update periods, we have chosen each mechanism to be represented by an independent service. This allows for more flexibility such as periodic sampling with different intervals. Another feasible use case for example would be performing voice recognition based on the first few seconds of an outgoing or incoming call. This would require a service that is triggered by a PHONE_STATE intent.

Each authentication mechanism service is started and managed by the UAService. Communication between the UAService and each authentication mechanism is enabled through intents. Using this communication link, requests can be made from each individual authentication mechanism in order to get the current confidence level. This value is equal to the probability that the owner is present, multiplied by the weight carried by the mechanism. Given that each mechanism runs as an independent service, weight decay may easily be performed using an AlarmManager or simply a function which is called periodically within the authentication thread.

Either periodically UAService gets the confidence level and weight from each mechanism. It then calculates the overall result. If the result is above the threshold requested by the Pico client, a "Message" is passed back saying that Pico should unlock. Otherwise a negative result is returned, letting the Pico know it should be locked.

5.2 Threat model

5.3 Implementation details

5.3.1 Main application and services

The user authenticator for Android is designed to work as a bound service. According to the Android documentation a bound service exposes functionality to other application components and as well as external applications. It is developed as a regular service which implements the onBind() callback method to return an IBinder. The service lives only as long as a component is bound to it. The service implementation class is called UAService.

The UAService is a central node in the application. It is a bound service for any Pico client which wishes to register for events. Furthermore, it binds any authentication mechanism that is available, enabling it for authentication.

The UAService periodically broadcasts intents to registered clients saying if the Pico should be locked or unlocked. The following interface is exposed to available Pico applications through the "what" parameter of the "Message" class:

MSG_REGISTER_CLIENT

Used for registering a client. The "Message" should have as the "arg1" parameter the level of confidence required for unlocking. This value should range from 0 to 100. Any values outside these limits will be truncated within the range.

MSG_REGISTER_CLIENT

Used for any application to unregister as a listener for this service. No additional parameters required.

MSG_GET_STATUS

Used by any application when an authentication request is needed. Although the service periodically broadcasts to its registered clients what is the authentication status, explicit requests may also be performed using this "Message".

UAService interacts with AuthMech objects in order to communicate with an authentication mechanism. Each object is responsible for interfacing the communication with an authentication mechanism. A valid authentication mechanism service needs to extends the AuthMechService abstract class which defines a standard way of communication with the UAService.

Each AuthMechService is programmed as a bound service. UAService binds these services through AuthMech objects. Each AuthMechService exposes the following message passing interface:

AUTH_MECH_REGISTER

Used for registering the UAService service as a client to the AuthMechService.

AUTH_MECH_UNREGISTER

Used for unregistering the UAService service as a client to the AuthMechService.

AUTH_MECH_GET_STATUS

Used by the UAService in order to request the authentication confidence from the AuthMechService. The value will be returned in the arg1 parameter of the Message passed.

5.3.2 Owner configuration

5.3.3 Voice recognition library

5.3.4 Face recognition library

5.4 Results

Appendix A

Appendix Title Here

Write your Appendix content here.