Luis Filipe Machado Malhadas

Bachelor of Science

Perceiving is Believing

Authentication with Behavioural and Cognitive Factors

Dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Computer Science and Engineering

Adviser: Henrique Domingos, Assistant Professor,
NOVA University of Lisbon

Examination Committee

Chairperson: Doutora Sofia Carmen Faria Cavaco
Members: Doutor Antonio Eduardo Cardoso Pinto Dias
Doutor Henrique João Lopes Domingos

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

December, 2016

Perceiving is Believing

Authentication with Behavioural and Cognitive Factors

This document was created using the (pdf)LaTeX processor, based in the "unlthesis" template[1], developed at the Dep. Informática of FCT-NOVA [2].
[1] https://github.com/joaomlourenco/unlthesis [2] http://www.di.fct.unl.pt

To my grandparents Adelaide Morais Machado and
António de Morais Machado.

Acknowledgements

The author gratefully acknowledge the guidance and support provided by Dissertation adviser Henrique Domingos.
The author thanks as well for everything (words can not describe well enough), to his Mother Maria Teresa M. Machado.

Abstract

Most computer users have experienced login problems such as, forgetting passwords, loosing token cards and authentication dongles, failing that complicated screen pattern once again, as well as, interaction difficulties in usability. Facing the difficulties of non-flexible strong authentication solutions, users tend to react with poor acceptance or to relax the assumed correct use of authentication procedures and devices, rendering the intended security useless. Biometrics can, sort of, solve some of those problems. However, despite the vast research, there is no perfect solution into designing a secure strong authentication procedure, falling into a trade off between intrusiveness, effectiveness, contextual adequacy and security guarantees.

Taking advantage of new technology, recent research on multi-modal, behavioural and cognitive oriented authentication proposals have sought to optimize trade off towards precision and convenience, reducing intrusiveness for the same amount of security. But these solutions also fall short with respect to different scenarios. Users perform currently multiple authentications everyday, through multiple devices, in panoply of different situations, involving different resources and diverse usage contexts, with no "better authentication solution" for all possible purposes. The proposed framework enhances the recent research in user authentication services with a broader view on the problems involving each solution, towards an usable secure authentication methodology combining and exploring the strengths of each method. It will than be used to prototype instances of new dynamic multi-factor models (including novel models of behavioural and cognitive biometrics), materializing the PiB (perceiving is believing) authentication. Ultimately we show how the proposed framework can be smoothly integrated in applications and other authentication services and protocols, namely in the context of SSO Authentication Services and OAuth.

Keywords: Authentication, Security, Multi-Modal, Multi-Factor, Multi-Mode, Biometrics, Services.

Resumo

A maioria dos utilizadores de computadores já experiênciaram problemas de acesso tais como esquecerem-se da palavra chave, perderem os cartões ou 'dongles' de autenticação, falhar outra vez o padrão do ecrã, bem como dificuldades de interacção e usabilidade. Face ás dificuldades da não-flexibilidade das soluções de autenticação forte, utilizadores tendem a reagir com fraca aceitação ou relaxando o suposto correcto uso do procedimento de autenticação e aparelhos, tornando a segurança pretendida inútil. A biometria, de certa forma, resolve alguns desses problemas. No entanto apesar da vasta pesquisa, não existe uma solução perfeita para desenhar um procedimento de autenticação forte seguro, caindo num equilíbrio entre intrusividade, adequação ao contexto e garantias de segurança.
Tirando partido das novas tecnologias, propostas recentes de pesquisas em multi-modelo, cognição e orientadas ao comportamento têm procurado optimizar o equilíbrio no sentido da precisão e conveniência, reduzindo a intrusividade para a mesma quantidade de segurança. Mas estas soluções também ficam à quem no que diz respeito a cenários diferentes. Os utilizadores realizam correntemente múltiplos acessos de autenticação todos os dias, através de múltiplos aparelhos, numa panóplia de diferentes situações, envolvendo recursos diferentes e diversos contextos de uso, sem "melhor solução de autenticação"para qualquer propósito.
A framework proposta completa a recente pesquisa em serviços de autenticação de utilizadores com uma visão larga sobre os problemas envolvidos em cada solução de autenticação, com vista à usabilidade combinam-se e exploram-se os pontos fortes de cada método. Será então utilizada para prototipar instâncias de modelos dinâmicos e multi-factor (incluindo novos modelos de biometria comportamental e cognitiva), denominando-se assim a autenticação PiB (Percecionar é Acreditar). Em suma, mostramos como se pode facilmente integrar a framework proposta em aplicações entre outros serviços e protocolos, nomeadamente no contexto de serviços SSO e OAuth.

Palavras-chave: Autenticação, Segurança, Multi-Modelo, Multi-Factore, Multi-Modo, Biometria, Serviços.

Contents

Chapter 1

Introduction

In most (if not all) computer systems nowadays, user authentication is considered as a fundamental building block and a primary line of defence of computer security. It is also the basis for most types of access-control principles and user accountability systems. In the IETF RFC 4049 [19], authentication is defined in general as the process of verifying an identity claimed by or for a system entity. The authentication property is instantiated in two different steps of an authentication process: the identification step and the verification step.
Identification involves proofs of a claimed identities, with identifiers corresponding to unique digital representations of non ambiguous naming entities, establishing also the identification of principals as subjects of access-control policies. Verification involves the corroboration information as verifiable proofs of correct bindings between the entity and the claimed identity.
According to the above reference definition authentication always involve the process of determining whether someone or something is, in fact, who or what is claimed to be. Despite the direct concern of the authentication requirement as the basis to establish access-control and auditing activities, there are structural differences between such services as materializations of different security properties [49]. Authentication, like confidentiality, integrity, availability, access-control and accountability, are all fundamental security properties that are combined in the security approach of computer systems, in the reference terminology and conceptual basis in the main security standards and frameworks (ex., [47], [43], [41]). As a fundamental property, authentication is the basis for other security services, such as: establishment of secure communication channels, cryptographic

key distribution protocols or establishment of network based security associations between principals, regarded as naming entities acting as endpoints at different abstraction levels of any secure network protocol.
In the context of the thesis authentication is particularly focused on the user-authentication abstraction level. At this level, authentication is related to user-authentication guarantees established by user authentication services, based on correspondent building block mechanisms and authentication factors, as believing means by which a user provides the correct claimed user identity representation to the system. In summary, user level authentication is the means of establishing validity and correctness assumptions on user names and identification claims [49], to be able of reporting what events took place and who or what triggered them (in the context of User-Audibility or User-Accountability services), as well as of significance the application of a policy that restricts what actions can a user do at a certain point of time and space (as the Authorization dimension) [44].
A security policy is used to define what security means for a system and can have multiple purposes [4]:

Confidentiality
: Many government organizations and military enforce such policies in order to maintain information secret.

Integrity
: Banks, insurance companies and others care more about keeping information correct and precise than its confidentiality [36].

Hybrid
: Is a mix of confidentiality and integrity (e.g. access control systems).

Composition
: It's often useful to combine security policies and some are modelled to allow just that.

An example of a system application of a policy is an access control, which is responsible for the verification of a principal's identity and permissions at a secure context [49].
From the above definitions, although authentication and access-control are closely coupled fundamental properties, the results of user authentication processes are orthogonal to different access-models for user operations on authorized system resources. In this we include MAC, DAC, RBAC or ABAC control models and management policies.

Mandatory Access Control is defined by a policy modifiable only by system administration.

Discretionary Access Control is similar to *MAC* but permissions are transferable and not only by the administrator.

Role Based Access Control can be seen as a generalization of the previous, given that can implement both. Permissions are assigned using the notion of role, which can aggregate several users.

Attribute Based Access Control defines a different paradigm that relates the entity to the resource the action and the environment through the use of attributes, becoming the most flexible of the access controls.

Another key concept in the security realm is the notion of trust, as something that is attributed. Being trusted by another principal means that such principal has been given that property of integrity, on the other hand the principal might not be worthy of the attributed trust (said not trustworthy) [30]. Because trust can be broken assurance plays a key part in building a trustworthy system. Assurance is the guaranty that a principal applies its security requirements [1], [44]. In order for assurance to be provided evidences need to be obtainable. An evidence may take many forms, such as: credentials, logs, biometric, tokens, ...
But a new problem arises, as how to present these assurances without compromising ones privacy or allowing someone else to impersonate the presenter. The use of cryptography is almost mandatory to achieve secrecy and anonymity, may it be for messaging or safe keeping of information. It's a vast field of research that dates from 1500 B.C. by ancient Assyrian merchants with simple substitution cyphers (replacing one symbol by another) [42].

Transposition consists on the scrambling of the symbols position.

Substitution consists on the switching of one symbol by another.

Asymmetrical previously the key to cipher is the same to decipher, but with asymmetrical cryptography the ciphering is done with one key and the reverse process with a different key, reducing the key exchange problem.

1.1 Security Issues in User Authentication

There is a very long list of threats to authentication procedures, specially because the applied methods have not changed much in practice over time and most of

the requirements are for one to remember a password, which is many times a very short simple sequence of characters with an almost obvious relation to the user [44]. Among them it can be found (better summarized at table 1.1):

Phishing which is an act of deception to make others relinquish their credentials, most commonly by impersonating some authority requesting the user credentials either for confirmation or updating.

Social Engineering is an act of deception that tricks people into breaking their security procedures. Usually by influence to security relaxation or plain unfamiliarity.

Exploiting is the set of actions towards exploring system design flaws to bypass the security procedures or disabling them.

Denial of Service is a particular attack where one or more principals keep authenticating in order to overwhelm the system disabling its ability to reply to more clients who will end up not even knowing if the server still exists.

Impersonation is the act of claiming the identity of another principal. If some other principal is enrolling the identity of an unregistered principal is called Identity Squatting.

Over the Shoulder a particular attack in which it's attempted to see the user password by peeking over his shoulder. It's of particular interest for biometrics, that are often kept or introduced in plain sight

Human error is the act of losing the credentials or just lack of knowledge in security engaging in high risk behaviours.

1.2 Authentication Factors

In the more current approach, user authentication services involve the adoption of different types of authentication factors that can be validated by system targets, such as passwords or pass-phrases (used as a shared secrecy-factors), token-based secrets stored on physical memory cards or USB dongles (as a way to locally store secrets in a physical device the user possesses), smart cards (as processing devices that can store and process secrets or cryptographic keys in the context of supported cryptographic functions and protocols, with additional guarantees for proofs of user possession processing capabilities and biometry, associated to

Attacks	Authentication	Threads	Typical Defenses
User Device Target (Client)	Passwords	Guessing, exhaustive search	Large entropy; limited attempts; long passwords; robustness checking techniques.
	Tokens, Dongles	Exhaustive search	Large entropy; limited attempts; theft of object requires presence.
	Static Biometric	False match (difficulties in combining false negative/positive tradeoffs)	Large entropy; limited attempts.
Server Side Target	Passwords	Plain text theft; dictionary/exhaustive search	Hashing; large entropy; protection of password database.
	Token	Pass-code theft	Same as password; use of Dynamic OTP (One-time pass-codes).
	Biometric	Template theft	Capture device authentication; challenge response.
Eavesdropping, theft, and copying of user authentication factors	Passwords	"Over-the-Shoulder surfing"	User diligence to keep secret; administrator diligence to quickly revoke compromised passwords; multi-factor authentication.
	Token; Beacons	Theft; counterfeiting hardware; substitution-attacks; non-physical protection	Multi-factor authentication; tamper resistant/evident token; location verification strategies.
	Biometric	Copying (spoofing) biometric	Copy detection at capture device and capture device authentication.
Authentication Factor Replay Issues	Passwords	Replay stolen password response	Challenge-response protocol.
	Token	Replay stolen pass-code response	Challenge-response protocol; OTP.
	Biometric	Replay stolen biometric template response	Copy detection at capture device and capture device authentication via challenge-response protocol.
Intrusion Threats	Password; Token; Biometric	Installation of rogue client or capture device	Authentication of client or capture device within trusted security perimeter.
Denial of service Issues	Password; Token; Biometric	Lockout by multiple failed Authentications	Multi-factor with token.

Table 1.1: Some Potential Attacks, Susceptible Authenticators, and Typical Defences [49]

proofs of static physical properties of the user, as a human being.
Those classic authentication factors are today supported by well known technology in the context of specific authentication services and applications, with a well known roadmap of advantages and drawbacks, as well as trade-offs in different issues, when used as strong authentication factors. To maximize possible advantages, to overcome the drawbacks, or to optimize trade-offs, classic factors can be combined as independent factors, in a multi factor authentication strategy. Examples of simple combinations (for example in the case of two-factor authentication approaches) usually involves password based authentication and one of the other factors as a complementary proof of authentication, when strong authentication guarantees are required. Later we will discuss the trade off issues in the context of the related work against the vision of the dissertation objectives.
Also, with the emergence of mobile computing systems, complementary approaches to the classic authentication factors, looked as context-aware authentication factors, began to be considered as extended factors for additional proofs of authentication and access control. In this we include location aware authentication and time frame verifications, with particular interest as factors for attribute based access-control models [49], [23], [2], [24], [20], [17].
Despite the interest of this new vague of context-aware authentication schemes, particularly focusing on verifiable and trusted dynamic location attributes from devices in the context of mobile network technology, in the scope of the dissertation, we are particularly interested in other possible approaches for multi-factor

authentication services, using new authentication factors that have been envisaged in the more recent research on user authentication.

1.3 Authentication Evolution

Even though humans age and can change their appearance we have been identifying each other based on our looks, sounds smells and other aspects of ones physiology since ever. Biometrics is the automated measurement of biological or behavioural features that identify a person [4]. A principal can't get his biometrics stolen or lost, it can't deny them and are mostly impossible to fake (extremely hard to get mimicked), which makes biometrics a very interesting method of authentication.
With technology enhancements is now possible to diversify the authentication methods towards a sensory perspective. Sensors became more reliable, smaller as well as cheaper they are starting to appear on smaller and portable devices, such as mobile phones. Even for developers these technologies become easier to use, easier to combine, allowing people to move towards a perceptive authentication philosophy. Where one no longer needs to remember his password but instead needs to fulfil a set of requirements that only it can fulfil. This is based on the premise that all humans are distinguishable among themselves, although technology has not yet evolved to get such level of accuracy it is possible to increase it by combining multiple models [45].
Every authentication method has its pros and cons, and choosing the correct one for any context is a trade off task, so it comes easily that combining multiple authentication factors will enhance the robustness of the authentication procedure, e.g. using a token with a PIN (ATM card) makes the token safer to loose because it's useless without the PIN. So the procedure would become *something you have*, *something you know*, *something you are and something you do*.
Multi-modal authentication systems are like many people trying to guess the total number of marbles in the jar, no one will likely be right but their average will be pretty close if not correct. Such a system can also take advantage of each modality best features, but they face many challenges like, how is each peace of evidence collected, how much noise does the evidence contain, how to merge all evidences, false positives and false negatives, how much intrusiveness is required, ...

1.4 Objectives and Contributions: the PiB Approach

By designing a multi-modal dynamic context aware authentication system, one should be able to mitigate noise, bias, imperfections, accidental errors, impersonation attacks and complexity for the user when authenticating. For such, it's projected a framework that provides abstractions and mechanisms to cope with and manage the multiple models, contexts and resources. To illustrate the advantages, the smooth integration of mechanisms in the framework approach for extended authentication services is a relevant issue. In this we include, namely: cloud-web based OAuth authorization services, with verifiable authentication factors on remote applications using access-control leases requiring continuous pervasive authentication proofs. while noise is removed from context either by restricting the allowed authentication models or by balancing their relevance against each other, exploring multi-factor authentication in complementarity.

1.4.1 Objectives and Contributions

The dissertation proposal addresses the above considerations in the PiB (Perceiving is Believing) approach, a proposal for the design and implementation of a multi-modal dynamic context-aware authentication framework.
The main contributions in this objective are:

- Design a framework able to extract environmental information that is categorized and formatted into contexts which in turn are modelled to produce and process authentication proofs.
- Easy extension of new authentication methods, e.g., invisible authentication models.
- Example integration with established, well known authorization protocol, OAuth 2.0, making it easily pluggable in current authentication providers.

For the experimental validation of the above contributions, we implemented and tested the PiB framework in a controlled environment, where some evaluation metrics were collected and analysed for validation, addressing the following questions:

- How to evaluate the performance of the proposed system, comparing with other authentication factors and methods, for example, in the context of well-known authentication protocols and services?
  In this question we include:

1. The concerns related to the comparison of the performance indicators when we use our framework to combine PiB factors complementarily to conventional factors in concrete authentication services, namely OAuth v2.0
2. The concerns on the adoption of the proposed framework, to be smoothly integrated in current applications and other authentication services and protocols, namely in the context of SSO (Single Sign On) authentication services.

- How can we validate the proposal in the context of the complementarity of multi-factor authentication methods, performing the validation of multiple authentication factors evaluated in hierarchical sequences?
  For example, for the case of two-factor authentication (using passwords and face-recognition), if a second factor (a continuous face recognition) fails to see the correct user for some time window, then first step must be dynamically re-activated and users must be forced to insert their password to proceed to the second factor evaluation again.
- How can we evaluate the impact in the adoption of our proposal in the development or upgrading process of different applications?

In answering the above questions, we need to evaluate the impact of the solution in terms of specific analysis criteria and metrics. We focused our analysis in the following metrics:

Time observing the latency of authentication processes to fully execute correctly and completely the authentication procedures.

Memory observing the memory consumption in system operation.

Energy observing the impact in energy consumption, particularly relevant when we use mobile and resource constrained devices.

Impact on application-development when the proposed PiB framework is adopted to develop new or to update existent applications.

1.5 Report Organization

The rest of the document is as follows: Related Work (in chapter 2), which is an *exposé* of the relevant similar research followed by the Critical Analysis summarizing the contrast between the current research and the proposed framework,

the PiB Model (in chapter 3) further explains how the PiB framework is designed, how it is composed and how it operates, followed by the PiB development and prototype (in chapter 4), summarizing the implementation details, along with the technologies used, ending with a discussion on the usability of the framework in the context of software development processes; the evaluation and experimental assessment of the implemented framework is then presented (in chapter 5), discussing the evaluation observations and validation analysis. Finally, we conclude the document (in chapter 6) summarizing the concluding remarks, addressing some complementary comments, open issues and future work directions.

Chapter 2

Related Work

Authentication is a deeply studied and worked problem that has, an yet to be invented perfect solution. With the open use of new emergent technology and the evaluation of new adversary models and threats in new contexts of use opens the opportunity for new research directions in the topic. The concept has always been an intrinsic part of our species due to all the differences that provide looks and character, we are able to identify each other visually and behaviourally, but in computer authentication the problem is one of representation and assumptions that can be exploited.
So by devising authentication in factors, there are:

- Something you know
- Something you have
- Something you are
- Something you do
- Something you perceive

The relation between the context at which the factor of authentication is used affects the credibility of the system and by having the system perceive the context, it will be able to fine tune the best models [17]. Perceiving is Believing (PiB) is the augmentation of user authentication with the perception of it is purpose and environment.
The remaining of the chapter will discuss the main ideas behind each factor with

reference to published work as well as introduce new concepts in biometrics, authentication modes and context awareness, discussing identification recovery and the public acceptance of the contemplated authentication methods.

2.1 Classic and Biometric Authentication Factors

Classic factors refer to the well know and widely used factors such as knowing a secret (password), possessing an authorization (smart card) and being of a certain physiological characteristic such as a fingerprint. All those factors have been in use for a long time and subjected to extensive research and where some prove to have high maintenance or have an expensive implementation, others prove to be of low assurance (easy or guessable passwords) or cumbersome to use.
Authentication systems based in biometrics have been in development for many years and have inspired much research, becoming nowadays ever more popular due to mobile devices and the reduction of resources required to process the big amounts of data that are characteristic of these systems.
Such systems take their input information from sensors and compare that processed information with templates that have been previously arranged to determine if the user can get access or which kind of access it is allowed [44]. The issue with any biometric system is the decision boundary (see figure 2.1), trying to balance the strength of the security provided with the usability. If that threshold is more on the imposter curve then the system will accept more often intruders, if more to the right then will become increasingly annoying for the genuine user denying him access more frequently.

2.1.1 Something you know

In a ever growing digital world, where personal and private information is spread all over the *cloud*. The need for a safe access to information has never been more present, specially with the possibility of quantum computing breaking many of the cryptographic ciphers in usage.
Something you know, most commonly it is a simple memory challenge such as remembering a sequence of characters (also know as password). A password based memory challenge is composed of two parts:

Identifier

A public label that is uniquely associated to his profile. It can be used in a discretionary access control.

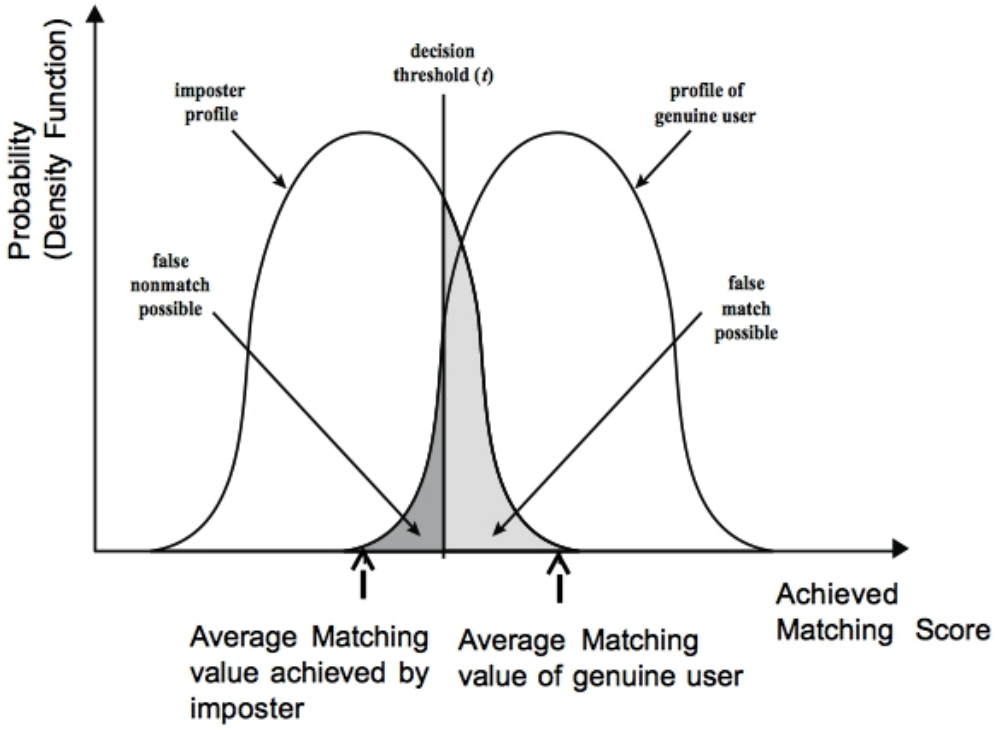

Figure 2.1: Equal Error Rate

Password

A sequence of characters, commonly hashed for comparison.

Human memory is flawed and degrades over time, in particular if the user keeps accumulating different accounts with different passwords, eventually it is near impossible to know which is from what account. People get old and careless about security, share their passwords or write them down loosing them, some use one single password for everything, others just use a very simple password. Even with all the research that has been done to try and tackle these problems either by slightly changing the authentication methods used, by trying to enforce policies (e.g. forbid the use of common passwords, periodic mandatory password change) or by the use of cryptography (e.g. hashing a password with salt) but they either delay the problem or just solve a part of it [49].
Challenge-response is a general purpose authentication method to verify knowledge of a secret different from a value in an attempt to solve the insecure channel problem. There are many examples of such systems from salted one-way hashed passwords, Kerberos, Zero-Knowledge till key agreement systems. A different approach to the same problem could be by using Matrix based passwords which coupled with a virtual keyboard randomly masking the entered codes can virtually span the matrix life time indefinitely. All these fail to even consider the

environment at which the authentication takes place, forcing users to use cumbersome tokens to access it is account.
A recognition-based system, instead of a recall-based, such as the user/password scheme, is said to aid the user while preventing it from having insecure behaviours (e.g. harder to share its password) [9]. *Déjà Vu* is a system designed to take advantage of the human episodic memory (which is the memory of autobiographical events) that allows humans to recall images easier. The system randomly selects m images, among which are n images belonging to the user chosen set (with $m-n$ decoy images). This method makes it very difficult for the user to share its password because it is highly complex to describe, but at the same time is easy to recall the correct images for the user [9], or is it? It presents a relaxation of the security considering that there should be an equilibrium between the number of images (password length) and the recall ability of the user, plus there is no contemplation of contexts as well (e.g. blind users will not be able to use it at all).

2.1.2 Something you have

While authentication systems became more complex, knowing something became knowing a lot of some-things. A token-based authentication is the sort of authentication where the user possesses a token that needs to be presented to the system, that in turn is going to validate such object as valid and with the proper access level.
Something you have, allows for the complexity of system access to increase potentially without the user perceiving it making the information more secure, but doesn't address the weakest link in the security chain, the user itself. Tokens are either a simple memory card that holds the information and requires special reader for usage rendering them expensive, or a smart card that is equipped with a microprocessor and is capable of performing calculations for cryptography, which doesn't decrease the cost. They can also be connected requiring physical link in order to work, or disconnected such as *RFID* devices, but they might be cumbersome to carry (e.g. when in a beach), can be lost or get damaged preventing access to the system and the special readers are not compelling either. To overcome the readability problem there are USB connectable devices (a.k.a. USB Dongle), which use a widely available connection standard. Of particular interest are also the mobile phones and tablet devices that can, in theory, be used in the form of a smart token.

2.1.3 Something you are

If taking metrics that aren't immediately changeable like iris patterns or finger prints. These are usually stored as a vector of its features and is later used to compare against the provided identification.

Finger Print

It is a widespread typical example of a static biometric method, available already in many cellphones, computer keyboards, among other devices and used mostly by law enforcement and justice. A set of feature distances and numbers of the ridges and valleys are stored [44],[49].

Iris Print

It is currently possible to scan the iris with common low cost cameras and even mitigate intrusiveness [3] while collecting a stable texture consisting of crypts, furrows, corona and freckles [46].

Retina Scan

Harder then an iris print due to the need of infra-red light for the capture of the unique blood vessels cross-sections distances.

Palm Print

A derivative of the finger print but with the advantage of easier visibility of the pattern due to its size. Collects principal lines, wrinkles and palm texture from low quality source and ridges, singular points and minutiae points from high quality source [46].

Facial Recognition

Due to its natural use by almost every human it is a topic of interest for research and carries a high public acceptance. Collecting 2D and 3D metrics of the human face is a very complex task that can be defrauded with the use of glasses, cosmetics, surgery or even time [46], [4].

D.N.A.

For precision and undeniable proof there's always the code that truthfully identifies almost each and every thing alive (unless they are twins). It is also a very expensive and still a lengthy process although not intrusive in terms of usability. It is characterized by a sequence of 3 billion bases of four different types (in the human case): Adenine, Guanine, Cytosine and Thymine.

2.1.4 Multifactor Authentication

Multi-Factor authentication is on the rise, with many starting to provide two-factor authentication (e.g. Google), it is an easy step until more factors are counted in, but they haven't. Some argue that multi-factor introduces redundancy and therefore reduces usability. Every authentication method carries advantages and disadvantages and the best way to exploit the advantages while mitigating the disadvantages is by aggregating the multiple sources of authentication [8]. The aggregation can be achieved by a multitude of methods, voting strategies, Bayes Factoring [14], in sequence, policies, ... see figure 2.2.

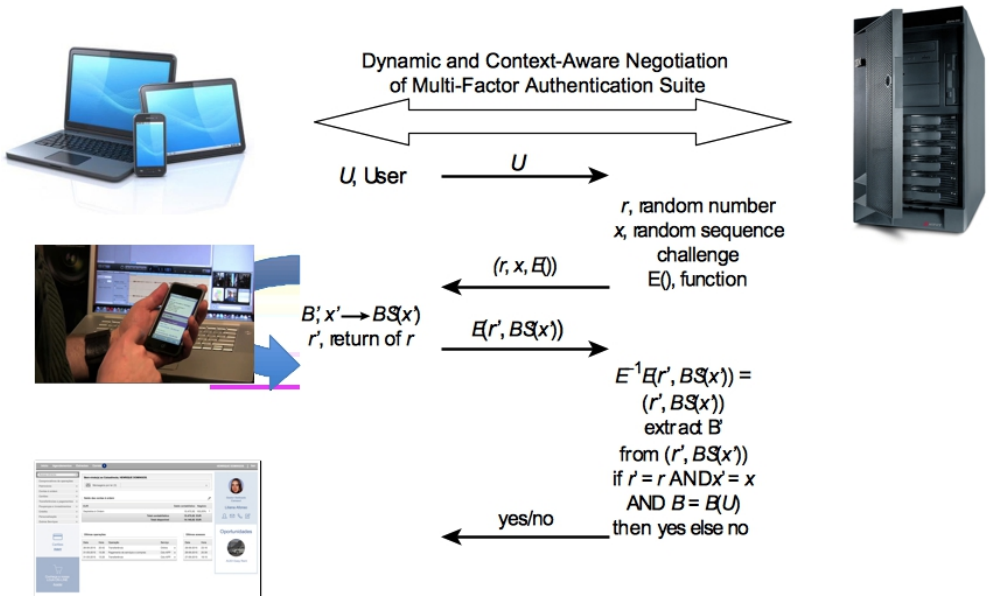

Figure 2.2: Multifactor Context-aware Authentication Schema

2.2 Behavioural and Contextual Factors

Life has been developing authentication systems for as long as interaction among different parts has been occurring. Mammals have been performing authentication since they are born, bounding immediately with their progenitors by smell or memorizing their looks, voice, actions, ...

2.2.1 Something you do

If the metrics are modifiable or specific user behaviour such as voice pattern or signature. These are highly dependable on what is being measured and how for the identification.

Voice Print

Also know as Speaker verification uses either a set of utterances that need to be spoken on a static context for the verification or under a dynamic context, the collected features can be: Spectrum, Glottal pulse features, pitch, energy, duration, rhythm, temporal features, phones, idiolects, semantics, accent, pronunciation, ... providing over 90% of success rates [46]. Keystroke

Gait Traits

Allows for recognition from greater distances by using human motion features techniques such as Moving Lights Displays, Maximal Principal Gait Angle, Motion history image, Hough Transform, Scaled Prismatic Model [25]. Similarly to Face recognition can have a very low intrusive level, and therefore has a widespread acceptance by the public. It is also a biometric that can easily be sensed by mobile devices with inertial measuring units [18].

Keystroke Pattern

Well studied biometric extracted from features such as key hold time, inter-keystroke times, duration of a keystroke, . . . [21].

Handwriting Pattern

Signature is a static example of handwriting recognition, extracting features like signature shape, pen position, pressure, pen direction, acceleration, length of strokes, tangential acceleration, curvature radius, azimuth [46]. It has been under use for a very long time, but is easily falsifiable.

Gestures Patterns

May it be touch-screen gestures or hand gestures these can be used to identify a person and are not hard to extract [33]. If used with enough features it is a reliable authentication method that is adaptive, computationally efficient and trainable [7]. Further more if coupled with touch gestures separating palm from fingers this authentication method can be used in a pattern lock scenario enhancing its resilience against attacks [38].

Mouse-based challenges

Even though keystroke techniques presents better results, mouse usage presents potential for authentication tracking events such as clicks, movement, wheel, durations, angles, distances, silence (lack of activity), drag&drop, curvature and stroke tangent, velocity, angular speed, acceleration, ... [6].

User Habits

Daily habits like buying freshly baked bread or going to work every week day in a schedule following the same route may be used as a biometry specially due to its extensive potential features. It is one of the least intrusive, if not the least, authentication methods considering that there's no need for special interface or any new input from the user making it even more prone to securing its phone. Apple already provides an API for dynamically configuring the authentication process according to the phone's location [34], in [39] rhythms are used arguing that a person can use its preferred music as the input, other sources of information may include browser or phone call history, calendar, . . . [40].

2.2.2 Multimodal Biometric Factors

The idea on implementing multimodal biometrics is similar to the multifactor but by making use of multiple modalities instead of factors, in particular if there is already in place a biometric system, the same information source is used (sensors) to be processed by multiple algorithms and applied a fusion technique. All with cost and scalability trade-offs.
Typical multimodal system may have one or more modes of operation [37]:

Serial executes modalities in a sequence restricting each time the possibilities but not requiring all the information at once.

Parallel executes modalities at the same time but requires all the information to be available at start.

Hierarchical executes modalities in an hybrid method. Particularly useful when many classifiers are to be fused.

[27] Uses fusion at the scoring level to identify users according to their palm print and face image. [12] Uses another fusion technique to continuously aggregate keystroke dynamics, mouse movement and stylometry. [22] Proposes a bi-modal system that is able to authenticate users with a mobile in a noisy environment through a Gaussian mixture of inter-session models, merged by multi-fusion algorithms. Other methods include sparse trees or similarity preserving hashing, ... No approach leaves space for the environment to be included, some may try to adapt to poor conditions but not in a dynamic fashion, rendering them inflexible.

2.2.3 Something you perceive

People react differently to the same context because everyone experiences the world differently (e.g. a lumberjack looks at a forest as a resource but a camper would see it as leisure). Perception is defined as a identification, interpretation and organization of sensory information in a context or environment. An example of application is the targeting of advertisements, but towards a security context is of particular interest applied in identification and peer authentication. It is a relatively new area of biometrics that draws inspiration from psychology and neuroscience and aims to capture a cognitive signature of the individual broadening the context at which biometrics can be applied (e.g. while playing on-line game). BioCatch is an application developed by a start-up company that makes use of cognitive challenges in order to spot intruders and other attacks with an accuracy of over 80% in parallel with other behavioural biometrics [50], [29].
Some behavioural biometrics might be considered cognitive due to the influence of some cognitive processes in behaviour, others, in literature, restrict cognitive biometrics to those from bio-signals only. Cognitive biometrics are here presented as a solution to such influence (which may actually be noise), i.e., where a gait analyser would fail to authenticate a physically tired user a cognitive system would not because the cognitive process being measured is not muscle dependent. These systems carry however an higher level of complexity and in some cases a large intrusion degree, e.g., many of these systems depend on brain signals being captured which is possible through the use of electrode caps (figure 2.3) attached eventually to a computer interpreting them through a series of cables.

Stimulus-Reaction

Similar to the challenge-response mechanism, the authentication process takes place by extracting features from the reactions of the central nervous system to stimuli such as image, audio or video observation [35], but can also include taste and smell or touch.

The reaction can be measured by electroencephalogram (EEG), electrocardiogram (ECG), electrodermal response (EDR), blood pulse volume (BVP), near-infrared spectroscopy (NIR), electromyogram (EMG), eye trackers (pupilometry) and hemoencephalography (HEG) [35].

Image Recognition Puzzles

BioCatch uses a PIN entry screen that forces the user to select the numbers and place them at a certain location, enabling it to extract cognitive features such as the time it takes to find the numbers, which ones are selected first,

Figure 2.3: Electrode Cap with 256 electrode slots.

among others [50]. The objective is to explore the visual memory of the user.

Visual search

One interesting challenge is the mouse courser disappearance, in which people will uniquely take different times until they start the search for it, they will start the search in different ways as well and take different approaches [29], exploring the decision making abilities of the human brain. Searching for a target in an image and track eyes movement and pupil dilatation [13] is another example.

Working memory and information processing speed

Humans are capable of reasoning, problem solving and learning from mistakes by using working memory, that can be explored despite its decay.

Automatic Processing and Priming Effects

A variation of the Stimulus-Reaction explores the Priming effect, where the stimulus is designed to trigger specific reaction, such as a picture of a spider will trigger a feeling of fear but not all will react with the same

amount of fear, people even have different reactions to fear, where some might flee others will fake death and remain static. All this happens in the sub-conscious mind triggering muscles and hormones involuntary with different times, quantities and structures [13].

2.2.4 Fusion of Multimodal and Multifactor Authentication

There are many proposals on fusion techniques of models or factors used in authentication, focused on decision/score level, feature level or low level (sensors and information sources), in particular for biometrics whose measures are filled with noise and imprecisions.
[48] Proposes an expert fusion at the decision level based on similarity algorithm between the template and the claim. [26] Proposes a differential evolutionary algorithm that scores according to an adaptive weight-exponent fusion technique which is parametrized with the help of an heuristic. [12] Developed a distributed fusion scheme for sensor level that combines local decisions by minimizing the global Bayes risk assessment of the sensors. Other example are voting strategies, averaging, ... [37]
The problems being addressed are with respect to:

Decision Problem The moment at which the multiple information sources produce a result. If they disagree how should they be mixed towards a single result.

Load Problem How many models or how many factors are to be used, from the available set.

Cost Problem If it can be achieved with cheaper devices, or if it targets specific environments that require unique, secret settings.

And despite the vast choice of fusion methods there is no best solution, not even a standard for which developers may guide while implementing multimodal or multifactor authentication systems.

2.3 Authentication Modes

Different methods of authentication have different impacts, i.e., it is not very practical to keep one's finger at the sensor to be able to use the device therefore a finger print is not well suited for a continuous mode of authentication. On the other hand, if it is used to authenticate the proper owner of a firearm or maybe of some

vehicles it might make sense to use fingerprint as a continuous authentication method. Different methods for different reasons imply different modes and each mode has its own advantages, but they are most complementary among each other having a trade off between complexity and intrusiveness. A continuous mode of authentication may be more intrusive than a one-shot mode of authentication were the user has to introduce his credentials only once, compared to constantly (every other time window, the least). Unless the user is constantly observed and tracked on a controlled environment allowing the system to identify specific traits that uniquely identify each user. Such system executes an invisible authentication method, one that requires no user interaction (that is perceivable) and is the least intrusive but of very high complexity and reduced extensibility.

2.3.1 One-shot Authentication

In this mode of authentication one's credentials are verified once and if authentication is successful then access is granted until next iteration. The use of static biometric measures are best suited for this type of authentication. Usually it is a simpler implementation and with better performance results because of the one to one comparison nature of the authentication process (principal claiming an identity by matching credentials or fulfilling a challenge), but if authentication fails then the user needs to repeat it, and it could be doing this *ad infinitum*, if is the incorrect user it could go on until a false accepted arises, or enough practice has been achieved. A barrier or limit needs to be imposed in order to mitigate this problem.
Another approach may comprise the complexity escalation of the challenges proposed.

2.3.2 Continuous Authentication

In a continuous model the authentication process takes place "uninterruptedly" until an imposter is detected, which can be achieved by taking multiple readings along a fixed length of time and extracting the features from there (e.g. by average) or take a single sample of a window of time and compare the snapshot against the template. Due to noise and irregularities of sensor readings continuous authentication needs to handle with care how to react to authentication fails therefore a thresholds or tolerance is required before failing [10]. This method is more intrusive for the user than one-shot, so it needs to be designed with care. Despite it is of an higher processing complexity and the trade off between the

time to identify an intruder and its false acceptance rate it is widely used [12], [28].

2.3.3 Invisible Authentication

Drawing inspiration from [29], their invisible challenges pose a new perspective on authentication modes. The key concept is the unawareness of the user that it is being authenticated, so a face recognition can operate as an invisible authentication if the user needs not to interact with the system to be recognized (e.g. a casino trying to identify card counters) as opposed to a system that requires you to hold the camera or sit in a certain place to be recognized. So information used in the authentication can be extracted from continuous and cognitive biometrics and is the least intrusive and most complex of designing and processing.

2.4 Multimodal and Multifactor Frameworks

Although the concept is actually old and has been somewhat explored, e.g., automatic bank machines from where one draws money by using a card (token) and a code (password) is a two factor authentication method, it has not been generalized to a standard or hardly extended beyond the integration of three factors. Some focus on problems such as modalities or factor fusion [15], [48], [12], while others focus on usability problems such as intrusiveness.
An interesting approach to the integration problem is found on [11], who developed a system (CYOA) that allows users to chose authentication models and is implemented making use of the know authorization protocol OAuth. While accommodating multiple models of authentication, CYOA relays heavily on the users choices who tend to relax in order to achieve easier usage, and lacking other forms of configuration is of arguable adaptability.
On the other end is CASA [17] (Context Aware Scalable Authentication) a system that automatically selects the authentication models to be used according to a naive Bayes classifier to access the risk given a set of factors that don't require user interaction. Stripping the users of almost all control makes it a black box and as such may get knowledgeable users suspicious of its workings.
There is no framework or infrastructure able to balance the required security with the users wishes for easy usability recurring to multiple models or factors of authentication (much less both), amounting almost always to a decreased comfort of the authentication procedure.

2.5 SSO Authentication Backends and Identity Management

Single sign on systems, as well as Federated Identity management aim at centralizing who holds the user credentials with the intent to reduce possible leacks while user confort is enhanced with less accounts to remember credentials for.

2.5.1 Single Sign On Systems

Single Sign On (and Single Sign Out) systems aim at reducing the need for the authentication moments to occur, and once authenticated by a peer, then all trusted partners accept the same user without question (or an authentication), just by presenting the certificates obtained at the authentication moment. Usually to access a resource held by a third party, which also translates in an enhanced security in the sense that the user information is no longer spread over the cloud and he can actually memorize a small set of username/password pairs of widely trusted entities, on the other side, the certificates must be managed with care not to be leaked and if the trusted partner gets attacked then many peers may get compromised as well.

2.5.1.1 OAuth

Oauth is an authorization framework and related protocol, enabling a third-party application to obtain limited access to WEB services, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the service, or by allowing the third-party application to obtain access on its own behalf [16].
OAuth has "de facto" become a highly influential protocol due to its visible swift and wide adoption in the industry and in well known large scale web services and internet cloud computing providers. The initial objective of the protocol was specifically oriented to serve the authorization needs for users and web applications, but the protocol has been significantly re-purposed, re-targeted and evolved over the years as a standard adopted by the major identity providers, and more recently Facebook, Google, Amazon, and Microsoft, have re-purposed OAuth for user authentication and developers have re-targeted OAuth to be used as native authentication and access-control services in mobile platforms. The abstract flow (as observed in figure 2.4) is composed by the following procedure:

1. A thrid party application requests the user to access a protected resource.

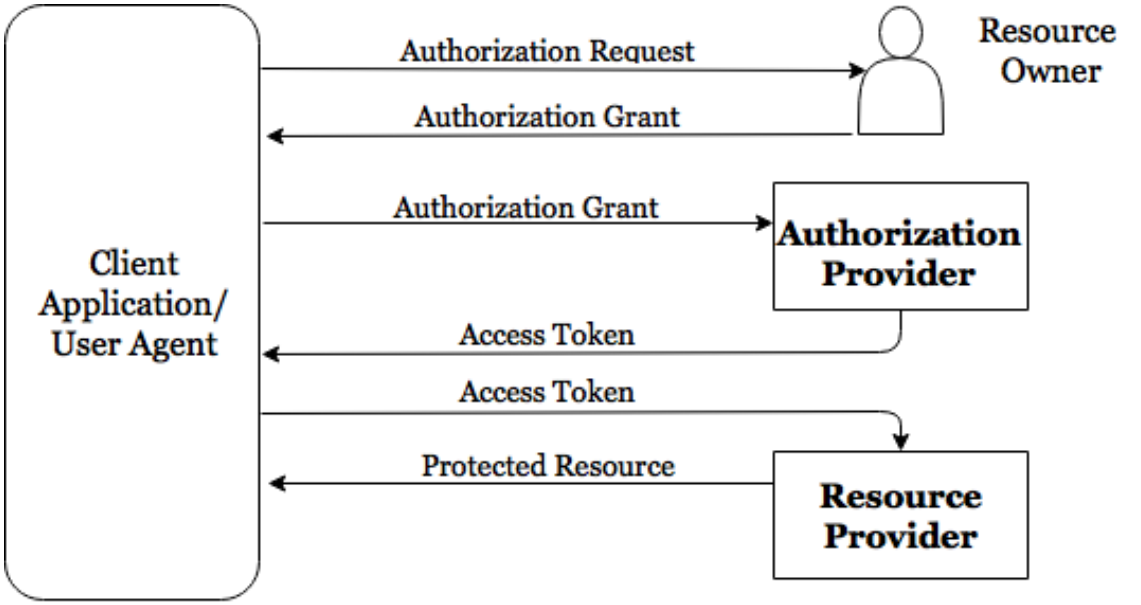

Figure 2.4: OAuth Abstract Flow

2. User grants, or not, permission. If not then the process terminates here.

3. If user grants permission then the application requests the authorization provider by presenting the permission grant from the user.

4. The authorization provider, if all checks out, gives the application an access token.

5. The application requests access to the resource provider by presenting the access token.

6. The application is granted access to the resource.

2.5.1.2 SAML

Security Assertion Markup Language is a XML-based language for the exchange of security information, namely authentication claims, in a distributed environment interconnecting on-line principals, acting as identity providers or service providers. The SAML assertions convey authentication proofs about the involved subjects, as statements about the subject issued and verifiable by authoritative entities, possibly implemented in the context of a Single Sign On solution (SSO). A current strong interest is today dedicated to the relevance of SAML as a well-acceptable extensible standard to deploy complex authentication policies based on multiple authentication elements aggregated in the authentication-claim of subjects or users, in a very generic scope [32], scaling beyond the FIM (Federated

Identity Management) approach [49].
SAML is developed by the Security Services Technical Committee of OASIS [32] [5] and evolved from 2001 until today (with a on-going agenda of new extensions, not only from the industrial community, but also in the interests of academic and research communities).

2.5.2 Federated Identity Management

Federated Identity Management (FIM) is a relatively new concept dealing with the adoption of common identity management services and functions across multiple authentication domains. The FIM is particularly oriented to act as an orthogonal, generic and scalable identity management solution, application independent and neutral to a possible extensible set of attributes representing user identifiers and associated proof of authentication elements. The central concept between the FIM approach is the aggregation in a Single Sign-On verification system of different conjugated elements, which is particularly interesting to be explored as a conceptual base for the purpose of the present dissertation, namely:

Peer-entity authentication as a verifiable association or binding of authentication proofs and identities in logical connections providing the required binding confidence.

Authorization describing and granting access-control policies for specific end-services or registered resources.

Accounting as a process for logging access and authorization.

Provisioning as enrolment process for subjects as end-naming entities (such as users, processes, services, computers, devices or any abstract identifier).

Work flow automation as a declarative way expressing data movements and correct information flows in the context of authentication and access control protocols.

Delegated administration as the support for Role Based Access Control (RBAC) models to grant permissions.

Password or secret-sharing synchronization supported as a process for SSO or RSO (Reduced Sign-On) function, enabling a user to access any resource in a Federated Identity Domain, to access all the remote resources.

Self-Service Passwords Reset as a function enabling end-entities to modify identification credentials for possible use as future claims.

Federation as the architectural model where authentication and permissions are passed on from one entity to another entity, usually across multiple identification boundaries, thereby reducing the number of required proofs of authentication, or as possible chains of incremental authentication proofs.

2.6 Critical Analysis

From the previous overview it can be seen the potential and flexibility of aggregating the many authentication procedures, of the authentication methods in a new dynamic enhanced multi-factor authentication approach poses as an opportunity to increase the resilience of the authentication against both attackers and users. While reducing usage complexity and therefore making more user friendly as well as promoting a secure behaviour. It is worthy of note that no system completely perceives its environment in order to override the defined authentication procedure and optimize it towards an increased usability and effectiveness.
Although there is much research done and already some commercial application of some of the methods presented, even with aggregation of multi-modal or multi-factor, none (to the best of my knowledge) is a complete extendible multi-factor multi-modal multi-mode context aware type of system.
Coupled with the comprehensive approach to the authentication problem, there are the following contributions:

Environment Management

After proper identification of the environment in which the authentication takes place it is intended to chose the appropriate combination of authentication methods. This should take into consideration ambient noise, available sensors, user preferences and authentication history. Perceiving contexts with resources and evaluating their validity.

Extensibility

Considering that authentication can take place at any time, for many reasons, then it is important that it can actually occur so the actual method used is unrestricted, given that peers agree. The ability to complement the implemented methods with new ones (e.g. novelty branch of Cognitive Biometrics) is not seen unrestricted in literature. If it can be perceived it could

be implemented.

Integration

Integrating the implemented framework on other platforms serves both as proof of concept and serves the adaptable nature of the proposed framework, enabling it to be used over a wide range of devices.

2.6.1 Context Awareness

One way of removing noise from recordings is by taking samples from the "silent" environment and use them to ignore similar frequencies from the authentication phase, that is the idea behind context awareness, i.e., by distinguishing contexts at which multi-factor and multi-modal authentication takes place makes possible to adapt dynamically the importance attributed or even possible to be used, as well as predict changes in the expected measures. If a developer wishes to assure that the principal is looking at the requested info, and that it really is him, then it may draw the a manager to only accept a context in which a photo is collected from a camera resource. It is also possible to take into consideration user preferences about privacy, e.g. the user may be willing to use its habits as an authentication factor for a company login but not for an email account. The user may be coerced into authenticating, and the authentication should fail in such context [30]. These scenarios are not contemplated in a broad dynamic relational authentication scheme, even though many of the referenced implementations take into consideration instances of usage cases, where noise and bias are considered.

2.6.2 Identification Recovery

Every now and then the user looses its token or forgets its password and it is as if its identification is lost, so every system provides an alternative method of authentication, often a token sent through a secondary channel (such as an email or sms) that is valid for a short period of time. With continuous authentication methods it is common practice to revert to a classical authentication procedure once the continuous authentication deems the current user as an imposter or the user himself changed its metrics. For biometric systems it needs to be possible for the user to re-enrol its biometrics or update them if they are to be used for a long time, but in the case of perceptive then it might be under short periods of time (one may change opinion often).

Despite its apparent importance it is not a very researched subject and therefore there are very few approaches on how to recover one's identity.

2.6.3 Public Acceptance

Comfort applied to usefulness, is the main goal of user friendly devices. Inserting a password or pattern every time the screen goes off is at least annoying, causing an interruption of the objective task, even promoting the relaxation of security towards easier patterns and passwords. So it is easy to see the advantage of some of the introduced authentication methods but with the increased performance of these methods came an increase of personal information provided to unknown systems that can potentially be stolen or misused. What for some people does not pose a privacy issue for others may be a sensitive question.
The issue is more relevant over biometrics research, and most studied implementations have some measure of user acceptance, but for the focused research there is no user impact study that comprise multi-models along with multi-factor authentication systems.

So as it has been referenced, the authentication realm is a vast field of knowledge that has been among us for as long as life exists. Populated by many different approaches, technologies, issues and solutions that are, in most cases, not combined just because no one has yet tried. It comes easy to see that the aggregation of authentication factors, models and modalities coupled with an higher level structuring of the parts, such that, they can take advantage of each other strengths may sacrifice some performance but just marginally when compared to the security and usability gains.

Chapter 3

PiB Model

As seen previously no system has yet accomplished a full integration between usability, efficiency, adequacy and security guarantees, which is understandable due to the nature of the problems that were tackled. With this understanding the objective is to design a prototype of an authentication system, that when an identity is claimed it is able to verify the credentials and adapt to the environment, may it be in whichever mode or combination of them, sequential or not, but dynamically adapted to the context in which is occurring while accountability is provided and both user and system have some control over the procedure.
How, when and with what is defined in the remaining of the chapter.

3.1 Framework Overview

The authentication process may be devised in 3 steps: claiming an identity, fulfil a challenge, accessing the resource.
For an identity to be claimed it needs to be compared to some existent list of available identities which in turn implies that information needs to be stored or kept persistently and securely. Then the challenge which defines the inner mechanics of the process itself, being executed in an environment, that adapts to context (during the day, noisy room, ...) as well as in a hardware context (mobile phone, personal computer, ...) or a security context where policies are defined or maybe sessions are considered as well. Although combining all the information may be at times a mater of perspective, the possibilities start to emerge when such perspective over the issues are to adapt and complement instead of restricting

the process. Only after the challenge has been successfully solved the access can be granted, otherwise a different behaviour should take place, maybe a failure message and a new form, but it could also imply a new model to be executed with different authentication procedure or even maybe there is a threshold for failure as it happens in the case of many biometrics. Figure 3.1 further illustrates the process flow.

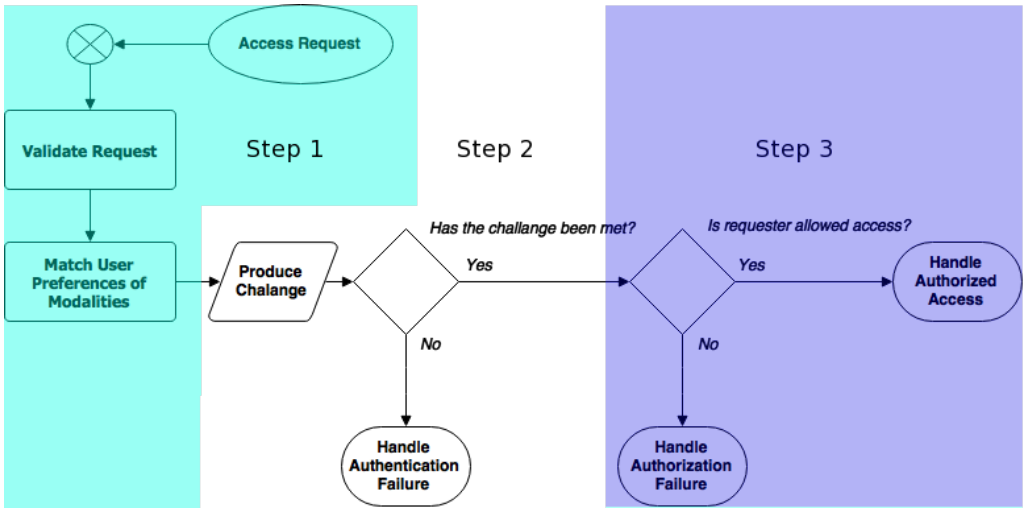

Figure 3.1: Authorisation Flow Chart

In order to break the identified steps into manageable problems the PiB framework is split in three basic modules:

Information Storage is provided by means of a knowledge base with an ontology designed by the *Ontology Engineering Group. Departamento de Inteligencia Artificial. Facultad de Informática, Universidad Politécnica de Madrid*, the mIO! Ontology Network is aimed to gather multiple other ontologies in order to represent context where information such as location, time, device, schedule can all be combined [31]. A relational database would have worked as well. But this choice allows for an extension point, along with extra information that can be inferred, from the current state of affairs.

Information Assembly aims at gathering information on the devices in which it runs (e.g. auditing of available sensors, memory, ...) as well as preferences on which authentications methods to use or how to use them or not, providing an abstraction over the hardware and preference management mechanisms to all authentication parties.

Core System is responsible for handling the authentication process. Transforming the sensory inputs, as well as received credentials, according to the

model for all considered contexts and preferences into authentication declarations, that ultimately allow, or not, the access to the resources. Therefor the authorization flow may be represented within this module.

3.2 Back-End Information Management

The used approach to represent and store the collected information is supported by the semantic web framework for Java, Apache Jena. Due to its persistence of information capabilities it is possible to use the framework similarly to a relational database while providing an extensibility point for new ways to combine the collected information and reason about it producing new, possibly unforeseen information. Its feature for transactions is also an enhancement towards performance, allowing a higher scalability, in particular for systems where multiple users are considered.

3.2.1 Ontological Model

To semantically represent the collected knowledge the *mIO! ontology network* is used and is defined as an interface ontology to gather the multiple composing sub-ontologies to produce semantic information about complex contexts [31]. It unites the following concepts as ontologies: Device, Environment, Interface, Location, Network, Provider, Role, Service, Source, Time and User (see Figure 3.2).

Device The composing ontology represents knowledge modelling devices, their components, charging modes, ...

Environment Models information about conditions such as humidity, temperature, noise, ...

Interface Represents knowledge about types, modalities and characteristics of interfaces.

Location The location ontology contains concepts that describe spatial units, entities, location coordinates, ...

Network Containing descriptions of network topologies, operators, administrators, price, ... underlying is the *Delivery Context* ontology.

Provider This ontology is composed of a wide range of concepts aimed to describe the provided services or their aggregation.

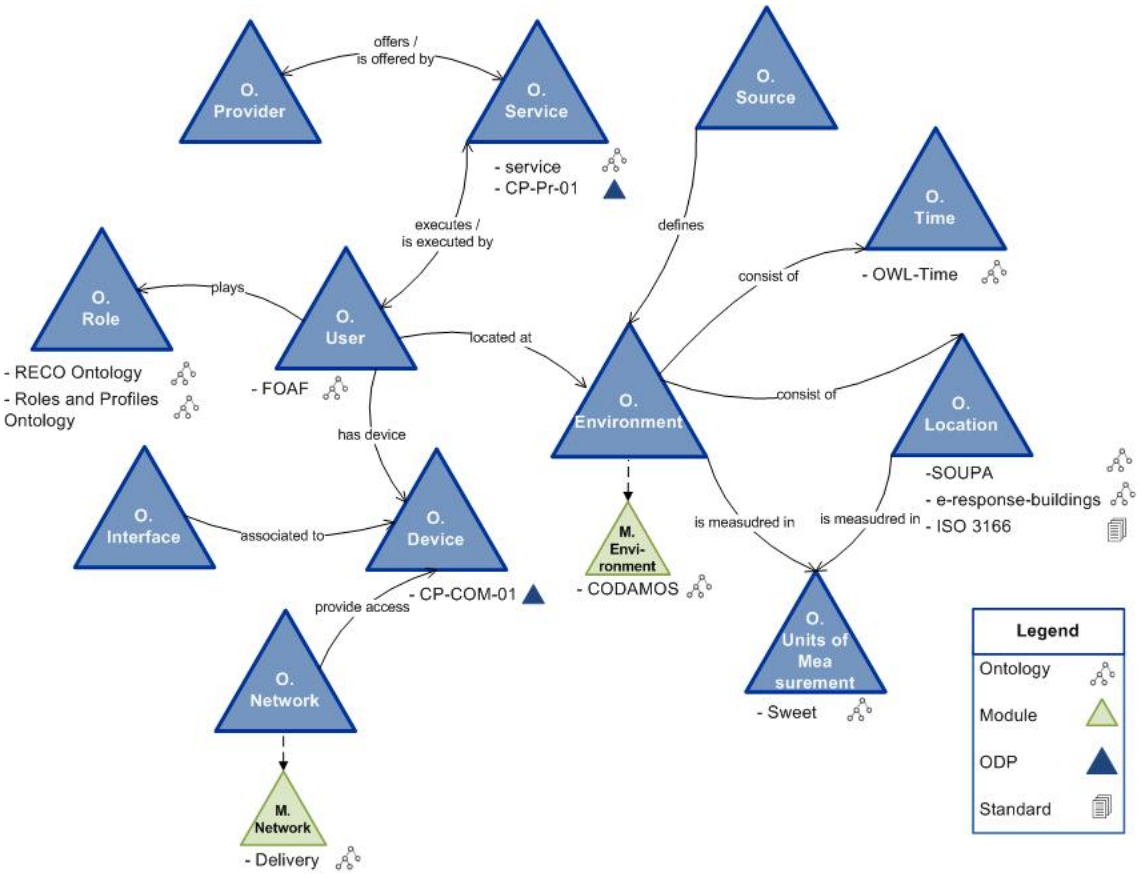

Figure 3.2: mIO! Ontology Network Conceptual Model

Role Represents information about preferences, profiles, roles, ...

Service Aiming to model knowledge such as: services provided by organizations or devices that use services. This is a more complex ontology that is able to describe the many functionalities of any service.

Source As in source of context information. Users, devices, services and groups of them may act as sources of information.

Time Models the knowledge about temporal units, entities, intervals, ...

User The operating ontology is Friend of a Friend (FOAF), modelling names, employment status, skills, and much more knowledge about users, groups or organizations.

3.3 Information Assembly

For complex applications, such as the proposed prototype, it is extremely difficult to account for every possible preference the user may consider. Gathering

all authentication mechanics in one big set of configurable preferences would also be a task users would greatly discard. In order to mitigate such problem, some could come with default values, but even then some of these default values could prove incompatible with deployment environments specificities. The implemented prototype, due to its limited set of mechanisms, uses mostly *static variables* or *enumerations* to define those default values.
Probing for the hardware can solve some of the configurations issues as well. Accounting for all possible combinations of hardware is another daunting task, and not in the scope of this dissertation. So, in order to reduce the problem, there are abstract formalism in place to trigger hardware operation, gather and format the expected information that should later be used for the authentication process. More specifically a Factory making use of Java reflection mechanism, producing hardware probing objects that will map the devices to workable well defined objects. By ignoring what hardware, and the internal mechanics of the hardware it is possible to extend towards maybe any type of sensor or hardware resource, allowing then the developer to adapt the library to their needs.
The other perspective on collecting information concerns the environment awareness that dynamically affects the authentication mechanisms used, e.g., forcing more steps, choosing a more reliable authentication algorithm or simpler because you are within the boundaries of a safe localization. All of the information that can be gathered is done by interpreting sensors output in the core system and acting according to them. These readings may or may not be stored for historical usages, knowing where you were in the past may still influence the present, e.g., consider a regular scheduled job for someone who always takes the same route back and forth every Monday to Friday, if phone localization is too far from his route it would immediately enhance its authentication requests. The information in focus is represented trough resources.

3.4 Design and Architecture

One of the main objectives is the modularization of the parts to be exchangeable, extendible, with new algorithms, approaches and ideas, i.e., replacing the ontological knowledge base with a relational data base or a NoSQL. Authentication managers are execution environment dependent, they can be working in remote pairs, and are designed to be interfaces that are implemented according to its requirements. The same applies to hardware detection and interaction. On the last layer of the abstract level architecture lays the core concepts for this proposal

(see figure 3.3).

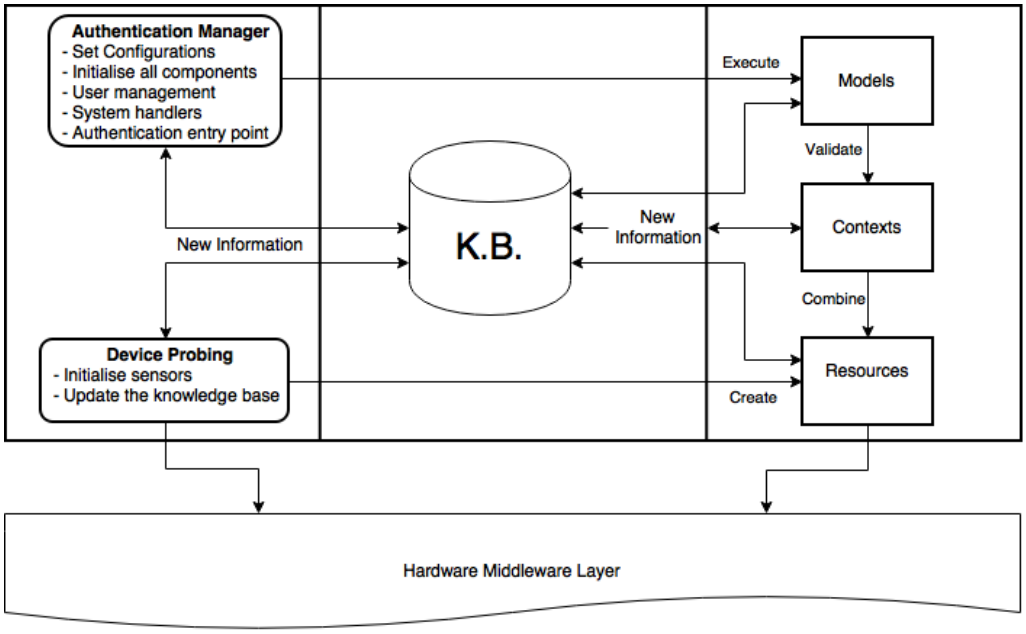

Figure 3.3: PiB Abstract Level Architecture Design

3.4.1 Core System

In order for the framework to overcome all the difficulties of authenticating a principal we propose some concepts to suggest a representation of the parts, components and other entities involved in the process of creating an authentication.

Resource

A resource encloses the consumable information collected about hardware and high level entities such as call history, or system folder. The resource object is by default uniquely characterized by a String object, and requires implementation of an initialization method that defines the starting point of the resource, along with an update method that will update the information within the resource.

Context

The context refers to a set of resources available in an environment or conditions and how and when to use them. Consider any web-camera as an image input resource, that may be filming at night, i.e., in a low light environment. In such a context the camera resource should turn a flash light in order to get a better picture. So context defines the usage of the resources. Contexts are build-able and updated-able as well.

Model

Authentication procedures are defined by this abstraction. Identifying the proper context, and defining the steps to take in order to authenticate an entity.

Manager

Used to make the parts work, i.e., getting and changing the preferences, trigger the execution of models, communicate with the knowledge base, creating and removing users, entities, ...

Decouple of the system parts: probing and management, knowledge base, core (models, contexts and resources) allows for the introduction of different types of data storage such as an encrypted relational database, as mentioned previously. Despite the used knowledge base providing a notification mechanism to get updated information from it we facilitate further the changeability, with another notification mechanism, following the observer pattern for its design, is used for model feedback to the managers, and can then act according to the user preferences.
All components are overridden with concrete implementations. Models evaluate the perceivable contexts, which in turn will read current state of resources providing the them with the necessary information to produce a higher level state, that the model uses to judge if the entity matches with the claim. Consider a mobile device equipped with a frontal camera and a microphone (these are the resources), then the system could have a model with a noise level context and a ambient light context, these contexts have each his own resource (the camera for the ambient light and the microphone for the noise level). Now the noise level context is valid if the microphone registers noise only up to a level, the ambient light context is valid only if there is a certain amount of light. The model could then proceed evaluating other contexts, e.g., a matching face context could be designed using the previous device and system where maybe it is configured to accept only one single face, followed by the voice match context, or even only the one that has the best chance, such as, if too dark then use voice match context and if too noisy use face match context. Going even further, contexts may even be composed, e.g., a context that validates other contexts according to location such that if at home validates less contexts than if at work. This can also be achieved with a model that triggers contexts according to a location mapping.
The atomic object for the core framework are the resources who represent any unit or quantifiable structure holding information, such as a web camera or a

microphone, but can also be a stream or just a string of text. This implies that they can also change across time and that contexts need to be notified of such.

3.5 Framework API

The proposed framework has for base a minimal set of classes that form the basis for the architecture presented in the previous section and are further illustrated on the framework API UML diagram (see 3.4). The *AbsInformationStorage* abstract class defines a base for handling the storage of the information, how to retrieve it and how to act on its changes. By extending the previous abstract class (and serving as a layer between the knowledge base and the framework) is the singleton class *KBLayerWrapper*, that implements a set of notification handler methods which are triggered from the knowledge base framework and that in turn update the related resources values within the PiB framework. This allows for the possibility of creating other methods of knowledge gathering that may be managed within the same knowledge base and having the new knowledge affect the PiB framework directly. Another pro may be the possibility of inferencing policies. On the other end of the API UML diagram can be seen the *IAuthenticationMan-*

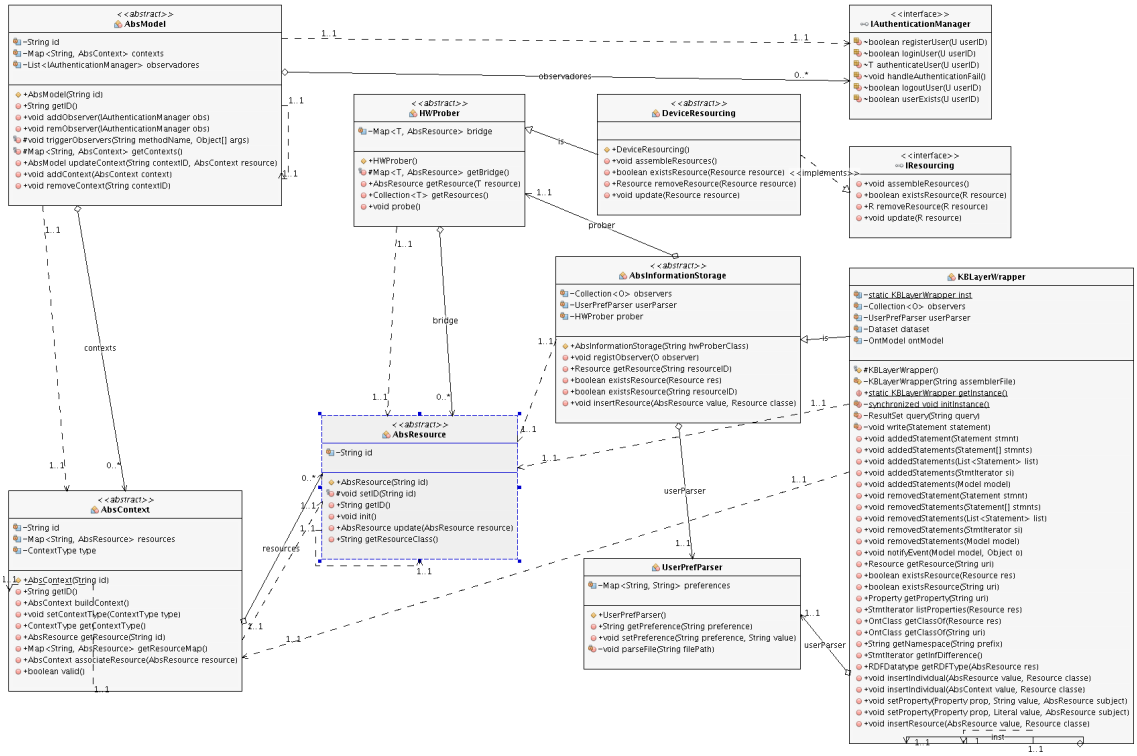

Figure 3.4: Framework API UML Diagram

ager interface defining the methods to be implemented for the entry point of the

PiB framework. Implementing such interface requires an instance of a concrete implementation of *AbsInformationStorage* in order to operate properly, along with a set of models that extend *ExecutableAbsModel*, who extends *AbsModel* and implements *Runnable*, a Java interface that makes an object executable by a *Thread*. This structuring of the architecture allows for models to be static or snapshots, containing properties such as assurance level, and used as variables to be passed around between managers. *ContinuousBaseModel* is an example of a starting point for a continuous model extension that implements the *void run()* method, as well as all methods inherited from *ExecutableAbsModel* super class, from which it stems. Such basic model only validates the known contexts in an infinite loop, triggering the known observers for invalid contexts as if authentication failed. Managers who observe such model will have to implement the *void handleAuthentication-Fail()* method, where complex behaviour could steam from such as: if GPS related model fails it may ask the user if it is travelling and temporarily disable that model or it may trigger a different model that is designed to recover from the failed authentication state (similar to a lost password procedure).
Contexts and resources are respectively defined by *AbsContext* and *AbsResource*. Despite the later being so simple it is the unit for the system, used everywhere and extended to represent accurately the contained information. While the previous is further extended to suggest a tree structure that restricts the application of the context as well as enhances the context with higher level mechanisms to handle the perceivable reality, e.g., a context of type *StaticAbsContext* defines a context with a non-update-able set of resources as opposite to the *DynamicAbsContext* that forces an *AbsContext update(String,AbsResource)* method.
At the centre of the schema is an *HWProberFactory* factory class that can create extensions of *HWProber* by using the reflection mechanism of java, at the method *HWProber getHWProber(String)* (see figure 3.5). This factory design pattern is further explored throughout the proposed framework to instantiate models, contexts and resources masking the details of initialization and construction of those objects.
All classes are separated through the package structure, depicted at figure 3.6, in which the *oauth* package holds the client class for the PiB OAuth service, the *info* package holds all classes used for probing the devices along with user preferences parsing, static functions and variables, plus the *core* package holding all classes related to the abstractions interfaces and their implementations.

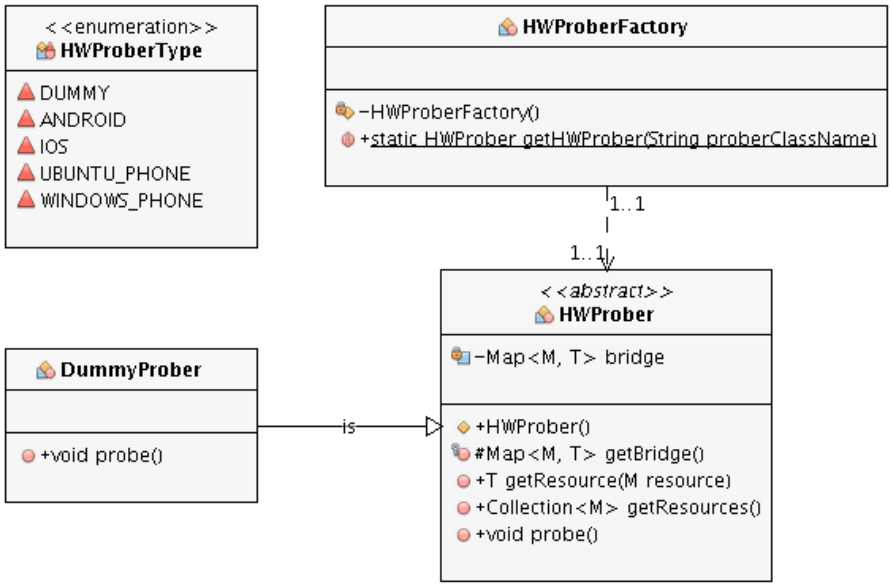

Figure 3.5: PiB Prober Detail UML Diagram

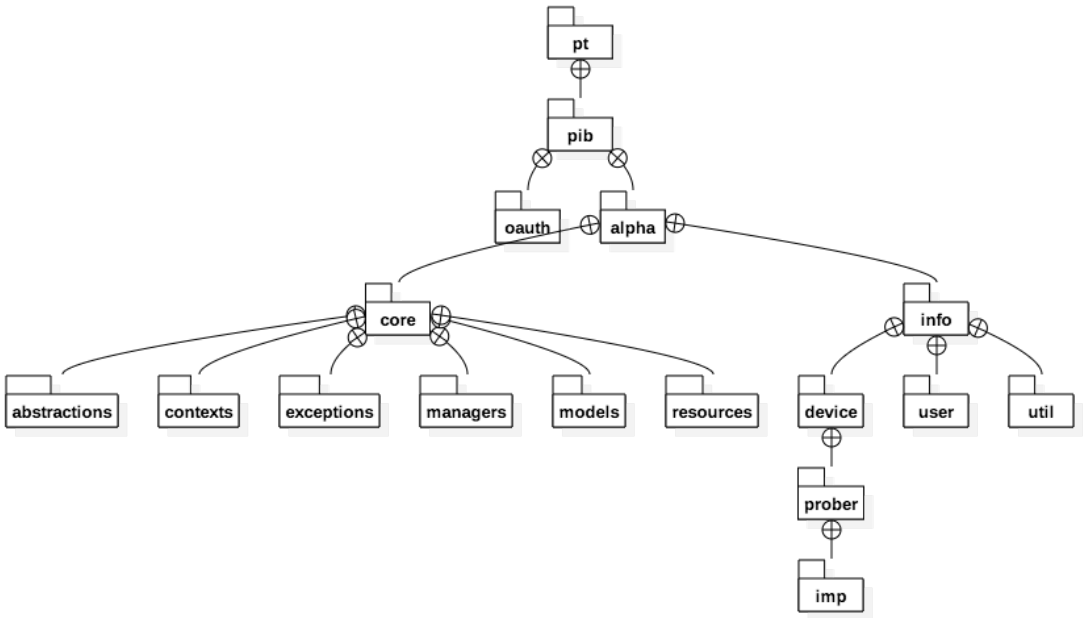

Figure 3.6: PiB Package Structure

3.6 Programming Model and Application-Level Support

Due to the simple basic concept the framework is equipped with some extensions to serve the purpose for which the abstractions are intended and to serve as examples and suggestions on how the *Perceiving is Believing* methodology is

envisioned. The intended use for the framework should cover, or be extensible to cover, all authentication methods executed in a complex environment that may or may not influence the method. So a set of resources representing sensors or information sources are implemented along with contexts, a few models and a couple of managers with different philosophies about handling authentication failures, or with different requirements for model initialization. The proposed type hierarchy is illustrated in figure 3.7. The models may be executed in one-

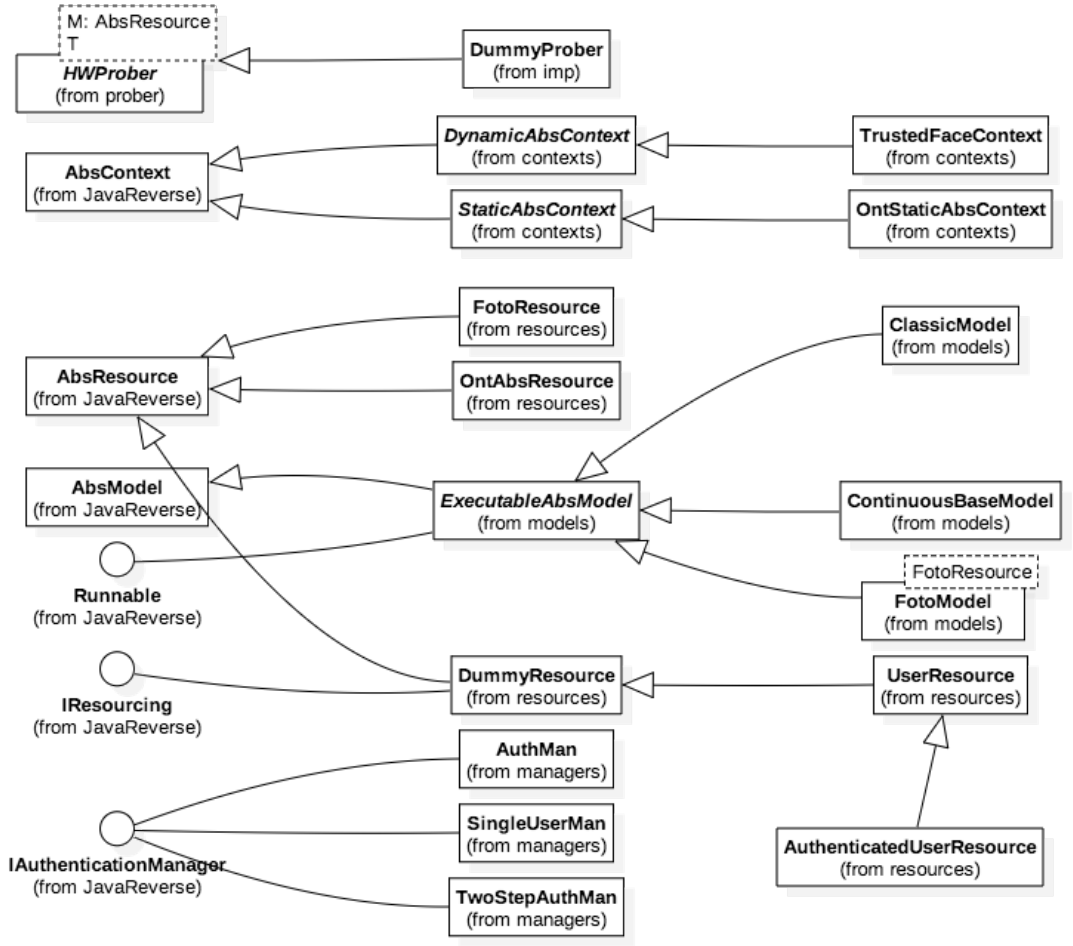

Figure 3.7: PiB Type Hierarchy

shot, continuously or invisibly (as mentioned in section 2.3) and only on one of these modes. Managers are the starting point, when and how models are executed. As such they are defined by an interface that may be a part of a larger structure as an OAuth authorization service (see section 2.5.1.1). Instead of taking a password for the claimed user it handles a photo and a puzzle or an audio recording and the localization history along with the browsing history.
Some methods introduce an information overhead to be exchanged between client

and server for the authentication to take place, and despite server authority some of the processing may take place on the client side the key processing must occur at the providers side. So a client sends information of what models are to be executed, along with their resources (associated to the contexts of the models, if any) values. Given that models on client and server are the same, the associated contexts within the models are the same, but the values of each context resource may differ requiring them to be updated by the client on the server. Another example for the server differing from the client is when a model needs to be generated on the server side, i.e., consider a model that asks the user to select a subset of images related to a subject. The full set of images to select from are to be sent from server after model selection for user selection.
An example of local application of the framework is as a layer on the android login screen. Instead of locking the screen based on a time out and allowing interaction if a pattern is matched, it can check if its location is within certain historical bounds before locking the screen, and if not at an expected location then demanding a face match or a voice match. This example's authentication manager would be implemented by a *LoginActivity* where the associated models could be extensions of *AsyncTask*.

So on this chapter, are introduced the concepts of the PiB authentication framework along with an overview of the modules composing it. Split into three major modules (Information Assembly, Information Storage and Core) working together to produce a dynamic process able to properly identify a principal claiming some identity through the combination of multiple factors, models and modes, validated over different contexts with access to different resources. Culminating with the suggestion of possible integrations with other software technology.

Chapter 4

Prototype Development

The developed library is used in a web service environment listening for OAuth 2.0 Authorization requests, that trigger the authentication procedure, if it has not yet happened or if the model requires it, creating a session. Acts as an OAuth authorization service, providing an authentication interface for client applications, such that they may access resource servers, through the presentation of an access token (see 2.5.1.1).

4.1 Integration with SSO Backend Services

Integration of the PiB framework within the OAuth, is achieved by extending the OAuth authorization service, in particular by implementing an Authorization Endpoint. The authorization service is executed remotely by a trusted peer which provides the authentication platform, that in our case shall be running a version of the PiB framework. This authentication interface is presented by a servlet engine, that listen for HTTPS requests in an extended OAuth format, keeping the interface compatible with standard OAuth protocol.
The OAuth Authorization request are extended to shorten the communication needs. Initially the resource owner triggers the client application that builds a standard Authorization request to the Authorization Endpoint but extended with a list of sets of the authentication models available or willing to execute by order of preference. On the server side the request is processed by the Oltu library and if all requirements are fulfilled then the PiB platform verifies the list and assembles the first set that is fully available and that verifies the local configuration

of policies. If no set of models is available then an error message is sent. So to authenticate the user the selected models are executed and if they successfully authenticate the user then an authorization token is issued.
In order to authenticate the user the authentication interface will have to handle multiple input formats to gather the necessary information in a safe channel. These formats should be defined and handled within the model(s) being used, but to standardize an initial input (and if client is confident of the request submitted) it may be sent along with the initial request (extending the standard OAuth protocol), and passed on to the PiB framework that will handle the arguments for the agreed models. Other approaches could contemplate a side channel, e.g. the Token Endpoint could lease short lived access tokens for models in continuous authentication modes, that coupled with volatile authorization tokens and lack of refresh tokens forces new authorization requests to be issued for the authentication process to be triggered again. In such cases an optimization, for trusted clients, is to answer the authorization request with the access token passing the request directly from the Authorization Endpoint to the Token Endpoint (without forgetting the authentication step).
The proposed PiB authentication process is triggered and managed by a PiB Authentication Manager which is defined by an interface, well implementable by the servlet class that handles the user login. The decoupled design makes the implementation easily integrated into other standard OAuth web services. Another approach could be, having a singleton PiB Authentication Manager to process the requests in a centralized design. Figure 4.1 further illustrates the proposed approach:

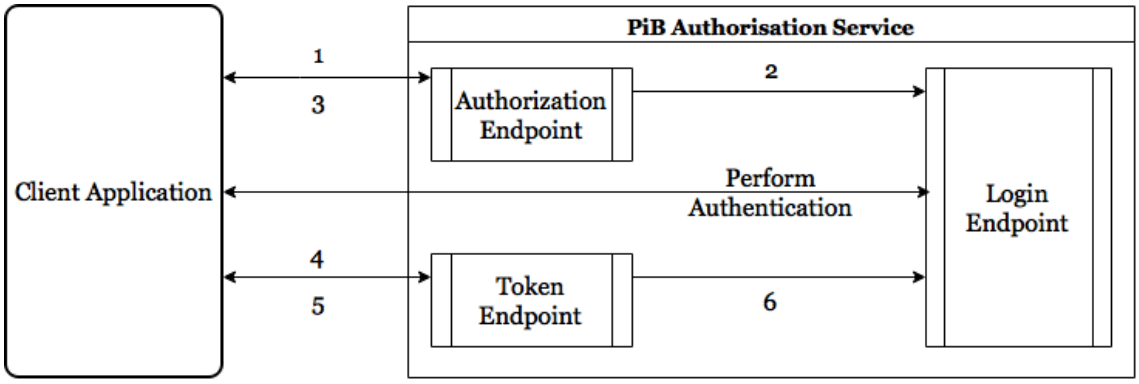

Figure 4.1: PiB OAuth Extention Integration Model

1. A PiB extended OAuth Authorization Request from a client application is sent to the Authorization Endpoint.

2. After proper validation of the request, the Authorization Endpoint redirects the client to the Login Endpoint if the claimed identity does not match the possible existing session identity. Otherwise an authorization token is issued.

3. The authorization token is delivered to the client.

4. An access token request containing the authorization token is sent to the Token Endpoint.

5. Token Endpoint verifies the validity of the handed token along with the session identity, replying with an access token to the client.

6. If the access token has expired or is no longer valid, for any reason, and there's no refresh token then the client gets redirected back to the Login Endpoint to perform the Authentication step.

The Authentication step is handled within the Login Endpoint that is an implementation of an Authentication Manager Interface containing a set of models validating their contexts. In a closer focus, the PiB Framework Authentication Managers differs from client to server but both will operate with the same algorithms to validate the contexts with the exchanged resources that update the contexts. In figure 4.2 can be observed the main components for each, client and server, authentication managers. Showing a servlet handler answering incoming requests that may trigger models to validate contexts that were updated with the request's included resources. The authentication manager needs to be designed dependent on the environment which is to be used, but all will have a collection of models to be used. Models need special care while implementing remote authentication cases because they will be responsible for the data in transport. User detailed information of features or other personal/private characteristics may be captured by a model and sent across the internet. Even though the problem of the communication channel is not the focus of this dissertation an SSL layer is to be used at all times for the communication between client and server.
The client wanting to authenticate by a model needs to provide the target model and the resources that will be composing a context that both models know and therefore are validated by the same process.

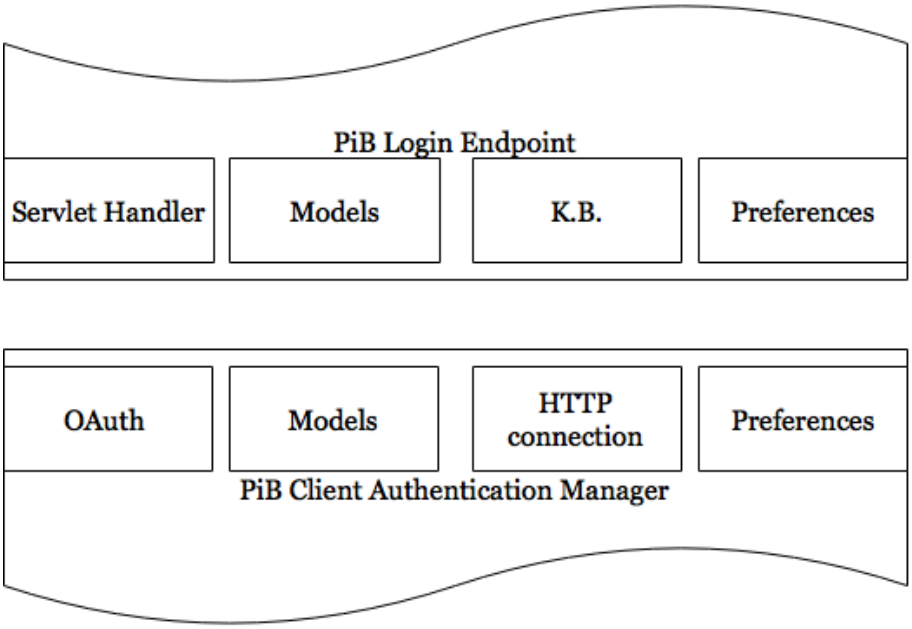

Figure 4.2: PiB OAuth Client Server Model

4.2 Technology

The implementation is coded in JAVA, and the runtime platform is required to provide a servlet engine communicating over HTTPS. Due to the aggregating nature of the proposal it makes extensive use of the following libraries:

Jena 3.0 Ontological engine with a Transactional DB. Used for access and management of the collected information.

Apache Oltu 1.0.2 OAuth 2.0 implementation with authorization and resource servers. The base framework for the communication layer.

OpenCV 3.0 Image processing framework, providing face tracking and recognition features.

Webcam Capture 0.3.11 Open source library providing broad spectrum support of webcams.

JSON 20160810 Provides JSON standard simple implementation.

Some libraries are easier to replace, e.g. JSON library is a lightweight implementation mostly used for the serializing and de-serializing of credentials on the login step of the SSO as defined by the OAuth standard. But others are deeply used and may have classes defined around them, e.g., without the OpenCV library the facial recognition model won't work at all.

4.3 Prototype Implementation

OAuth is extended at the login step implemented by an extension of a servlet (*LoginEndpoint*) mapped at *'/login'* that serves as the authentication manager implementation and attends to formatted login requests with parameters defining credentials or choices of authentication managers that may operate in different modes or with different policies. Once the request is received and processed (meaning credentials are valid) a session is created associating the authenticated user to it, after which the user is redirected to the *TokenEndpoint* where the rest of the standard OAuth protocol will be carried out. OAuth actually defines two endpoints for the authorization service: Authorization Endpoint and Token Endpoint. Both implemented for this prototype and respectively mapped to *'/auth'* and *'/token'*.
Prototype implementation doesn't make use of the full OAuth features, due to the flexible design of the PiB framework some requisites are not possible to be fulfilled. OAuth defines the authorization grant request to be presented from the authorization server directly to the resource owner, and this is usually achieved by redirecting the user agent to a login form that is to be submitted back holding the credentials (username and password). The PiB library carries a basic headless client for the implemented prototype to be tested anywhere, and also because the framework extends beyond the username/password realm predicted by OAuth there is no redirect to a form page but instead it is assumed to be communicating with a dedicated client aware of the needed information ahead of time, also shortening the communication time consumption and other simplifications as it is to be expected with any prototype aiming at a proof of concept.

4.3.1 Authorization Endpoint & Token Endpoint

The authorization endpoint is the servlet answering to authorization request with authorization tokens whose validity terminates after usage (see 4.1). Authorization tokens are volatile because the prototype focus on authentication and OAuth defines the authentication step to occur only at the authorization endpoint for the login step (see figure 4.1) which forces the login to occur on every usage.

Listing 4.1: Authorization Endpoint Sample Code

```
OAuthIssuer oauthIssuerImpl = new OAuthIssuerImpl(new MD5Generator());
HttpSession session = request.getSession();
String[] metodos = request.getParameterValues("methods");
String[] args = request.getParameterValues("args");
OAuthAuthzRequest oauthRequest = new OAuthAuthzRequest(request);
UserResource currentUser = (UserResource)
    session.getAttribute("currentSessionUser");
if(currentUser==null){
  response.sendRedirect("/login");
}
else{
  if(currentUser instanceof AuthenticatedUser){
    String redirectURI = oauthRequest.getParam
      (OAuth.OAUTH_REDIRECT_URI);
    if(!redirectURI.startsWith(redirectURI)){
      // Bad redirection... silently fail
      return;
    }
    else{
      String authorizationCode = oauthIssuerImpl.authorizationCode();
      OAuthResponse resp = OAuthASResponse
        .authorizationResponse(request,HttpServletResponse.SC_FOUND)
        .setCode(authorizationCode)
        .location(redirectURI)
        .buildQueryMessage();

      //save message in session
      session.setAttribute("auth_code", authorizationCode);
    }
  }
  else{
    return; // Not recognized user... it shouldn't reach here!
  }
}
...
```

Listing 4.2: Token Endpoint Sample Code

```
...
OAuthTokenReques oauthRequest = new OAuthTokenRequest(request);
validateClient(oauthRequest);

String authzCode = oauthRequest.getCode();
if(authzCode!= null && !authZCode.isEmpty()){
  String accessToken = oauthIssuerImpl.accessToken();
  String refreshToken = oauthIssuerImpl.refreshToken();

   OAuthResponse r = OAuthASResponse
                .tokenResponse(HttpServletResponse.SC_OK)
                .setAccessToken(accessToken)
                .setExpiresIn("3600")
                .setRefreshToken(refreshToken)
                .buildJSONMessage();

  response.setStatus(r.getResponseStatus());
  PrintWriter pw = response.getWriter();
  pw.print(r.getBody());
  pw.flush();
  pw.close();
}
else ...
```

Despite the volatile authorization the authentication lasts while the session holds the authenticated user, and the tokens associated to an expiration date, that are verified by the OAuth framework invalidating the tokens once such expiration has occurred, consequently invalidating the session. If there is a session user associated and it is an authenticated user holding valid authorization token, then it is allowed to get redirected to a recognized token endpoint that will issue standard OAuth2.0 access tokens.

The authorization endpoint is also responsible for registering new users (which is currently achieved through a PUT request), while redirection accepted URIs are a server defined configuration for the moment. Resource servers will also need to validate the tokens provided by their clients, which is the issuing token endpoint responsibility and because authorization tokens are ephemeral they are not validated (they're not even present on the requests).

4.3.2 Login Endpoint

The login step from the OAuth is where the PiB framework is integrated, and the entry point for the authentication process to actually occur. At this stage if there is no session then one is created to hold an unauthenticated user, followed by the the PiB authentication process. If the user provided valid credentials it gets authenticated and the session is updated with the authenticated user redirecting the client to the authorization endpoint.

Listing 4.3: Login Endpoint Sample Code

```
HttpSession session = request.getSession(true);
UserResource currentUser = ((UserResource)session
  .getAttribute("currentSessionUser"));
if(currentUser==null){
  try {
    OAuthAuthzRequest oauthRequest = new OAuthAuthzRequest(request);
    if(oauthRequest!=null){
      String clientID = oauthRequest.getParam(OAuth.OAUTH_CLIENT_ID);
      if(clientID!=null){
        UserResource unUser = new UserResource(clientID);
        unUser.setProperty(KBValues.HAS_ID_INFO,
          oauthRequest.getParam(OAuth.OAUTH_PASSWORD));
        session.setAttribute("currentSessionUser",
          authenticateUser(unUser));
      }
    }
    else {
      response.setStatus(HttpServletResponse.SC_BAD_REQUEST);
    }
    }
    ...
}
else if(currentUser instanceof AuthenticatedUserResource){
 String authStatus = currentUser.getProperty("authenticationStatus");
 if(authStatus.compareTo("authenticated")!=0){
  ...
 }
}
...
```

The authentication is processed by the manager, at the overriden method *AuthencticatedUserResource authenticateUser(UserResource)* as exemplified at 4.4. Where all expected models are triggered in a hierarchical fashion and if all conditions are verified then an authenticated user is returned.

Listing 4.4: Authentication Manager Sample Code

```
@Override
public AuthenticatedUserResource authenticateUser(UserResource userID) {
 boolean authenticated = true;
 for(ExecutableAbsModel model : this.models_queue){
  if(model instanceof ClassicModel){
   authenticated &=  ((ClassicModel)model)
    .isAuthenticated();
  }
  else if(isDual && model instanceof ServerFacialReconModel){
   authenticated &= ((ServerFacialReconModel)model)
    .isAuthenticated();
  }
 }
 if(authenticated){
  return new AuthenticatedUserResource(userID);
 }
 return null;
}
```

4.3.3 PiB Oauth Client

Because the prototype is, or may be, in a remote execution environment a client needs to exist for communication to occur. The headless design makes the client to be easily plugged in most environments and if the models do not require user interaction then renders the prototype almost invisible (excluding preferences). For the client to be usable with the PiB prototype, it must implement the authentication manager interface with a subset of models compatible with a subset from the service. This manager should perceive the models as remote models whose contexts are to be shared serializing the resources to the server within OAuth requests. 4.5 shows an example of a first request from the PiB client application to authorization endpoint with redirect on success to the token endpoint and extra parameters defining an authentication manager and the credentials for the claimed identity.

Listing 4.5: Client Authorization Request Example Code

```
...
  OAuthClientRequest
    .authorizationLocation(authEpURL)
    .setClientId(userID)
    .setResponseType("token")
    .setRedirectURI(tokenEndPointtURI)
    .setParameter("authManager", "default")
    .setParameter(credential.getID(),credential.getValue())
    .buildQueryMessage();
...
```

After the request the authorization endpoint will redirect to the login endpoint at which point the provided credentials will be evaluated and on success the client gets redirected to the provided *tokenEndPointURI* were a token is to be issued to access a resource as in the standard OAuth protocol. The client may as well request tokens directly to the token endpoint

Listing 4.6: Client Token Request Sample Code

```
...
  OAuthClientRequest
    .tokenLocation(tokenEpURL)
    .setGrantType(GrantType.AUTHORIZATION_CODE)
    .setClientId(userID)
    .setClientSecret(credential.getString())
    .setRedirectURI(resourceEndPointURI)
    .setCode(token)
    .buildBodyMessage();
...
```

4.4 API

The implemented prototype is designed to be used remotely but there are many applications where local authentication is indispensable among users and devices, and the proposed framework is well extensible to accommodate new models, new contexts and new resources managed by different authentication managers. As such, if the aim is a standard implementation of PiB with the OAuth 2.0 protocol one only needs to instantiate the *PiBOAuthClient* object that contains the necessary methods to register a user, build authorization and token requests, make and send the requests to the appropriate endpoints handling the login. Execution

of the client instance culminates with an access token that can be sent to any resource server that recognizes the token endpoint as a valid issuer.

4.4.1 Web API

Login requests to the PiB OAuth service follow the format defined at 4.1, where the suggested argument values are mapped to the models expected arguments.

$$/login?username = U\&method = M\&args = Arg_1, Arg_2, ..., Arg_N \tag{4.1}$$

Standard access is performed in all similarity to standard OAuth requests (as exemplified at 4.2), plus the extra fields to choose the authentication method, depending on what is available at the server. Although there are some mandatory parameters for authorization requests, not all need to be used at the moment, e.g., the *client_id* parameter is pointless at the moment, because only one client is prepared to handle the PiB framework.

$$\begin{aligned} /auth?username = userTest\&method = default\&response\_type = token \\ \&redirect\_uri = http://localhost/PiBAuthServer/token \\ \&client\_id = PiBClient\&args = teste \end{aligned} \tag{4.2}$$

response_type
: Defines the type of response expected. May be of one of the following values: code or token (see [16])

redirect_uri
: Who is to process the next step for the OAuth protocol. Usually contains the Token Endpoint URL.

client_id
: The application identification.

method
: Refers to the authentication method to be used (only one possible value: *default*)

Requests to the token endpoint are just as defined by the OAuth protocol, with the particularity that all will have expiration date associated according to the performed authentication, e.g., so if a continuous model is defined then the expiration should not be over some seconds while if a classic model it can last indefinitely.

4.4.2 Software API

Here is presented a subset of the implemented framework classes and methods. They compose both high and low level classes to provide an overview of the framework and the development philosophies. The main modules abstractions are: *IAuthenticationManager*, *AbsInformationStorage*, *AbsModel*, *AbsContext*, *AbsResource*.
As previously mentioned *IAuthenticationManager* is the interface defining minimal authentication manager functionality, as to be called by the PiB client for example. The interface is parametrized with the user type and contains method signatures for registering, authenticating, logout, verify user existence and handling authentication failures.
The remaining abstractions contain also some previously implemented basic methods along with objects to hold implementations of the containing PiB abstract structures, e.g., *AbsModel* contains a mapping of contexts to be validated along with a list of observers implementing the *IAuthenticationManager* interface that are to be notified on model changes. Extensions of *AbsModel* need only to define how they update their contexts having all observer registration, triggering and context management (add, remove, get) already provided. Following an incremental design, or specialization, of functionality, models are further extended as executable models who force implementation of the *Runnable* interface as either continuous models, one-shot models or invisible models (where user interaction is not allowed). The *ClassicModel* defines an one-shot executable username and password model of authentication with the following methods:

ClassicModel(String id)

ClassicModel constructor, taking as parameter the model unique name identifier.

boolean isAuthenticated()

Returns true if the model is in an authenticated state, i.e., if the last attempt of the user to authenticate was sucessful. In other words, it is the volatile variable used to transport the result of the thread execution that is validating contexts.

Map<String,CredentialResource> getCredentialMap()

Returns a mapping of the credentials needed to fulfil the classic model challenge.

void run()

Method triggered by authentication managers who implement *IAuthenticationManager* interface.

void setUserContext(OntStaticAbsContext userContext)

Defines the user context holding the credentials to the claimed identity.

The user context for the classic model is defined by *UsernamePasswordContext* class and is valid only when the held credentials match the credentials stored at the knowledge base for the claimed identity. Also, worthy of note is that the credentials, as well as the user, are just extensions of *AbsResource*.
Another important class is the PiBOAuthClient, responsible for the client side communication of the OAuth protocol enhanced by the PiB framework.

PiBOAuthClient()

Empty constructor.

boolean registUser(String uri, String data)

Registers an user with parameters/credentials defined at *data* to the *uri* authorization service. The user identification should be in data, that is to be JSON formated string with at least the following keys: *dataType* and *data*. *dataType* contains either a *user* or *service* string and distinguishes a registration request from a user of a registration request from a service (client applications or resource endpoints). Returns true on successful registration

void makeRequest(
String authorizationEndpointURL,
String tokenEndpointURL,
String userID
)

Builds OAuth request to the *authorizationEndpointURL* with redirect to the *tokenEndpointURL* claiming the user identification *userID*.

OAuthJSONAccessTokenResponse getAccessToken()

Returns the currently held access token. This method returns null if no request has been sent.

In summary the presented prototype implementation is able to authenticate the claimed user through an extended, wide spread usage, protocol of authorization

applying multiple authentications.
The development of a dedicated client shows a proof of concept, that establishes a communication channel through which the authentication is negotiated and achieved by issuing tokens with expiration dates. Provides the occultation of users credentials from third parties while the need for managing multiple accounts fades away and possibly even not having to remember their credentials.

Complete source code can be found at link.

Chapter 5

Validation and Experimental Evaluation

In this section we present the experimental validation conducted to evaluate the PiB framework prototype. We conducted this evaluation in a controlled environment, where some assessment metrics were collected and analyzed for validation purposes. As initially stated (in section 1.4.1) our experiments and observations addressed the following issues, organized as follows:

Section 5.1 Evaluation of performance indicators of the PiB prototype namely comparing it with other authentication factors and methods in the context of well-known authentication protocols and services. This evaluation included two validation concerns:

- the comparison of the performance indicators when we use or combine PiB factors complementarily to conventional factors in concrete authentication services, namely OAuth v2.0
- the evaluation on how the PiB approach can be smoothly integrated in current applications and other authentication services and protocols, namely applications integrated with SSO (Single Sign On) authentication services and solutions.

Section 5.2 Evaluation and validity of the approach to address the complementarity of multi-factor authentication methods, performing combined validations of multiple authentication factors in hierarchical sequences.

Section 5.3 Evaluation of the impact in the adoption of the PiB framework, in the development process of updating existent applications, as well as, in a perspective of a methodology to adopt PiB in the development of new applications.

In sections 5.1 and 5.2 we analyse the impact of the proposed solution in terms of specific analysis criteria and metrics, focusing on the following:

Time observing the latency cost of authentication processes to fully execute correctly and completely the authentication procedures;

Memory observing the memory consumption in system operation;

Energy observing the impact in energy consumption, particularly relevant when we use mobile and resource constrained devices;

In sections 5.3 we are more particularly focused in presenting quantitative and qualitative metrics in a software development process.

5.1 PiB Operation and Performance Indicators

There are some restrictions on the implemented prototype that cause an accumulation of overheads, such as, the HTTP communication or the authentication processing of the ever more complex data (biometrics, fusion, ...). Some models are simpler such as the classic user name password combination, others require continuous communication rendering them almost unusable over the internet.
To test and evaluate the impact of PiB, it was chosen a base client application requiring users to login, with its source code available that makes use of the OAuth protocol. LabCoat [1] is developed by gitlab, and allows the users to access their git accounts.
The base case scenario is provided by measurements taken from the unaltered LabCoat application. These measurements are collected with Android Studio, that provides a detailed report which due to the memory management has no encapsulation of the application memory, but instead provides a context were is found the proportional set size of the heap, i.e., the kernel metric for memory that considers the amount that is shared with other applications.
Table 5.1 shows the average measurement taken with LabCoat for memory and time used to perform authentication. But without access to service source code or sample implementation makes impossible to report about server processing

[1] LabCoat current source code can be found at link.

Metric	LabCoat
Time (Seconds)	0.33
Memory (Kilo bytes)	87.715

Table 5.1: LabCoat Performance Table

metrics.
LabCoat is compared to PiB framework by modifying LabCoat to generate the proper credentials and send them to a mock git service where the PiB framework is used as well to process the credentials received. The implemented OAuth PiB web service generates logs containing the metrics, which are then processed by Hadoop framework to calculate maximum, minimal and average times and used memory of each relevant step of the OAuth process where PiB framework plays a role.

5.1.1 Single Factor Authentication

The one-shot authentication mode is the simplest case scenario, and is the base for the evaluation of the PiB framework performance. The managers operating in this mode gather the credentials (user name and password) from the principal and execute models performing the authentication only once, which if successful terminates.
The base case implemented extends an abstract *HttpServlet* class that answers standard OAuth2.0 requests matched against credentials stored at the same knowledge base as the PiB wrapper.
Table 5.2 shows an increase of time as well memory, although time is expected to increase the significant observation is caused by the HTTPS with self signed certificates that need to be accepted by hand, increasing dramatically the response time from the client. The memory however doesn't increase so much when compared to the original client application. To better understand the influence of the

Metric	Single Factor
Time (Seconds)	1.681
Memory (Kilo Bytes)	122.356

Table 5.2: PiB Client Single-Factor Performance Table

proposed framework on the values observed at table 5.2 the values from table 5.3 show the performance values on the server side. The authentication process starts on the client which initially holds no session nor any token, and therefore it will get redirected to the Login Endpoint. After a successful authentication it

PiB Service	**Metric**	**Average**	**Minimum**	**Maximum**
1-Factor Authentication Requests	Time (Miliseconds)	23	<1	126
	Memory (Kilo Bytes)	14.538	-	-
1-Factor Login Requests	Time (Miliseconds)	471	<1	685
	Memory (Kilo Bytes)	15.538	-	-
PiB Login Request Processing Times	Pre-Process	72	<1	685
	PiB execution	374	<1	972
PiB Token Request	Time (Miliseconds)	9	<1	972

Table 5.3: PiB Service Single Factor Performance Table

is redirected holding a session to the Authorization Endpoint, at which moment it will be answered with an authorization token. After getting redirected by the Authorization Endpoint a login request is built at the client side for the Login Endpoint, which will initially pre-process the request identifying the models to be used, and the provided credentials. If the claimed user provides the correct credentials the authentication process takes place successfully as a one-shot solution that redirects the client back to the Authorization Endpoint with a session containing the user identity. Once the authorization is provided the client is redirected to the Token Endpoint with an authorization token, that will be exchanged by an access token, as the regular OAuth protocol.

5.1.2 Two-Factor Authentication

Two-Factor authentication over OAuth contains a big overhead of transmitted data, when compared to the previous authentication method, caused by the associated credential containing a matrix of the captured frame from client camera sensor, i.e., a processed photo of the user.
The collected 321 requests have an average of 0.053 seconds to get redirected and less than 0.001 seconds if holding a session.
The login processing takes an average of 4.276 seconds among 239 collected requests. The PiB 2-factor verification of the credentials averages 1.655 seconds, and the pre-processing of the request (reading request, validating, building contexts) just 1.494 seconds.
From the initial 321 authorisation requests only 52 got to the Token Endpoint, holding an average answer time of 5.192 seconds. All these values are compiled in table 5.4.
Despite the extra credential information, the average time measured for a full authentication cycle is in the order of a few seconds, which considering the effort and complexity necessary when compared to simpler solutions is an expected

tradoff towards higher security.
From an user perspective including a second factor of authentication (in particular one that may be imperceptible) can cause some disruption of the expected App behaviour, which for the implemented prototype translated in an increment on the number of login attempts when compared to single factor authentication, caused by the non-deterministic nature of the recognition algorithm used (from JavaCV framework). This also may lead the distracted user to a possible miss interpretation of the incorrect credential.

PiB Service	**Metric**	**Average**	**Minimum**	**Maximum**
2-Factor Authentication Requests	Time (seconds)	0.053	<0.001	17
	Memory (Kilo Bytes)	12.5	-	-
2-Factor Login Requests	Processing Time (Seconds)	4.276	<0.001	160
	Pre-Process	1.494	<0.001	160
	PiB execution	1.655	<0.001	96
	Memory (Kilo Bytes)	16.5	-	-
PiB Token Request	Time (Seconds)	5.192	<0.001	258

Table 5.4: PiB Service 2-Factor Performance Table

5.2 Evaluation of Validity

The collected measurements show the feasibility of the integration and development of the used application with the PiB framework which does not significantly affect the overall performance of the base application used. The observed client performance decrease is of about 5 times slower and a memory footprint of 40% higher, that for the initial values is yet a very usable system specially if taking into account the gains on authentication security both for user and provider.
The inclusion of the PiB framework on the OAuth protocol handling the initial recognition of the claimed principal identity allowed for an enhancement of the overall protocol security, making, attacks such as over the shoulder peeking of the username/password combination harder to achieve due to the second factor. In particular due to the imperceptible implementation of the second factor which may be configured like such by the user.
The integration with a third party client application shows its adaptability and with higher completeness of models, contexts and resources may provide a standard off the shelf solution for rapid implementation of enhanced authentication systems.
The service was hosted on the development platform which is connected to the internet through an ADSL internet service provider with a bandwidth of 24 Mbps

download and 1Mbps. The small channel is compensated by the low traffic (only single requests where handled at any time), although the platform had multiple loads and requests over multiple schedules and on multiple devices.
In therms of energy consumption the observations over the debugging period are that it is highly dependable on how many resources and how often they are used. A two-factor authentication scheme constantly taking photos and sending them through internet, even if occasionally, will consume the battery about 10x faster than a plain simple username/password, which was within the average device usage. Being exactly what is expected to be observed throughout any android device with intense camera use and internet access, and therefore the proposed framework has a small energy footprint if used without low energy consuming resources.

5.3 Impact on Application Development

Developing the prototype required for many of the concepts introduced to be extended to pack extra information references in order to improve performance and to make the framework modular. This coupled with the aggregating nature of the PiB framework gives it the potential to reduce application development time by a significant amount by just instantiating a Manager class.

- Models will be implemented, such as, facial recognition or two step authentication, prepared for out of the box usage.
- Information storage is interchangeable with others, just by implementing an abstract class that can communicate and operate on such storage.
- Integration of the framework may be accomplished by implementing an authentication manager interface that defines where are the preferences, which models and when are they executed.
- Some prototype authentication managers available already. Makes development to be accomplished by a few lines of code.

The PiB library package, so far, contains 3 authentication managers, 4 models, 6 contexts and 6 resource objects, as well as an OAuth2.0 extension client dedicated to the PiB prototype service. They are supported by a set of 37 classes, written with 2646 lines of code. Where the PiB prototype (OAuth service wrapper) has 3 servlet classes written with 600 lines of code, most of them dedicated to parsing HTTP input. The client prototype has about 400 extra lines on a single class when

compared to the implemented standard OAuth service test application. Most of those lines (about 75%) are due to the Android Camera framework, that works asynchronously and providing access only for a moment, releasing the capture before ending.

Chapter 6

Conclusions

In this chapter we conclude the dissertation report. First, we present (in section 6.1) a summary of the more important concluding remarks related to the dissertation objectives and contributions, as well as, to the achieved results. Then we discuss (in section 6.2) some open issues, regarding the state of our work and the current version of the PiB framework implementation. Finally (in section 6.3), we conclude by addressing future work directions, not only in terms of possible refinements or extensions of our work, but also in terms of future evaluation criteria and a more extensive experimental evaluation that could be considered.

6.1 Concluding Remarks

The implemented framework is used within an authentication protocol over a web service providing an interface for claiming an identification provided the appropriate correct credentials. This provides a proof of concept for the usage of the PiB framework as well as a base case used for validating when compared to a standard authentication protocol usage without the PiB framework included.
It is also presented an analysis of an extension case demonstrating that the overheads are well within a reasonably usable system. Showing that the development of similar platforms may become complementary and faster if the PiB framework is used.

6.2 Relevant Open Issues

Some problems have risen while developing the prototype, and caused some objectives to be dropped, such as:

- Continuous authentication renders the OAuth protocol for the authentication useless, maybe even violating it. Making sense only for dedicated systems, or maybe as an extension of a scope triggering verification of the token along with the credential.
- Transportation of the library into the android environment as well as others, e.g., IOS, implied that some of the depending libraries couldn't be used, even across different versions of the Android O.S, OpenCv for instance can't do recognition for the tested environment. Another example is, Jena which doesn't currently provides a working version for Android.
- Reasoning about contexts and resources lacks an improved integration with the Jena framework, it currently only works almost as a data base for the user information and credentials.

6.3 Future Work Directions

As for any sort of research there is always something that can be improved. In PiB's case, here follows some future work issues:

- Integrate OAuth scopes with PiB models, making some models good for some degrees of security but not for other. Let's say one should present biometric credentials in order to write, but for reading a combination of username and password would suffice. And even implementing a semi-continuous authentication on each request sent.
- New implementation of the *AbsInformationStorage* interface to handle relational databases as for other information persistence methods. In particular to support Android knowledge management.
- Further development of the library and integration into different platforms such as IOS.
- More models, more contexts, more resources and more Managers to extend even further the framework until it is able to perceive and reason about the environment and their agents.

Bibliography

[1] R Anderson. *Security Engineering*. 2008. ISBN: 9780470068526.

[2] L.-a. Authentication and A. Concepts. "IMIS-2012 KEYNOTE Talk". In: (2012), p. 2012.

[3] N. Benletaief, a. Benazza-Benyahia, and S. Derrode. "Pupil localization and tracking for video-based iris biometrics". In: *10th International Conference on Information Sciences, Signal Processing and their Applications, ISSPA 2010* (2010), pp. 650–653.

[4] M. Bishop. *Computer Security : Art and Science By Matt Bishop Publisher : Pub Date : ISBN : Pages :* Addison Wesley, 2002. ISBN: 0201440997.

[5] S. Cantor and I. Kemp. "Assertions and protocols for the oasis security assertion markup language". In: *OASIS Standard (March ...* March (2005), pp. 1–86. URL: https://svn.softwareborsen.dk/sosi-gw/tags/release-1.1.4/vendor/doc/saml-core-2.0-os.pdf.

[6] D. Chud and P. Kr. "Usage of computer mouse characteristics for identification in web browsing". In: (2014), pp. 218–225.

[7] G. D. Clark. "Engineering Gesture- Based Authentication Systems". In: (2015), pp. 18–25.

[8] E. D. Cristofaro. "A Comparative Usability Study of Two-Factor Authentication". In: (). arXiv: arXiv:1309.5344v2.

[9] R. Dhamija and A. Perrig. "Deja vu: A User Study Using Images for Authentication". In: *In Proceedings of the 9th USENIX Security Symposium, Denver, CO: Usenix, 2000.* 102590 (2000), pp. 45–58.

[10] T. Feng, Z. Liu, K. A. Kwon, W. Shi, B. Carbunar, Y. Jiang, and N. Nguyen. "Continuous mobile authentication using touchscreen gestures". In: *2012 IEEE International Conference on Technologies for Homeland Security, HST 2012* (2012), pp. 451–456.

[11] A. Forget, S. Chiasson, and R. Biddle. "Choose Your Own Authentication". In: ().

[12] A. Fridman, A. Stolerman, S. Acharya, P. Brennan, and P. Juola. "Multi-Modal Decision Fusion for Continuous Authentication". In: ().

[13] A. A. Galib and R. Safavi-naini. "User Authentication Using Human Cognitive Abilities". In: ().

[14] a. Giffin, J. D. Skufca, and P. a. Lao. "Using Bayes factors for multi-factor, biometric authentication". In: 611 (2015), pp. 611–615.

[15] B. Y. M. I. Gofman, S. Mitra, M. I. L. Lions, and O. F. M. O. Bi. "Multimodal Biometrics for Enhanced Mobile Device Security". In: ().

[16] D Hardt. "The OAuth 2.0 Authorization Framework". In: (2012), pp. 1–76.

[17] E. Hayashi, S. Das, S. Amini, J. Hong, and I. Oakley. "CASA: context-aware scalable authentication". In: *Proceedings of the Ninth Symposium on Usable Privacy and Security - SOUPS '13* (2013), p. 1.

[18] M. Hofmann, J. Geiger, S. Bachmann, B. Schuller, and G. Rigoll. "The TUM Gait from Audio, Image and Depth (GAID) database: Multimodal recognition of subjects and traits". In: *Journal of Visual Communication and Image Representation* 25.1 (2014), pp. 195–206. ISSN: 10473203.

[19] R. Housley. *BinaryTime: An Alternate Format for Representing Date and Time in ASN.1*. April 2005.

[20] W. Jansen, V. Korolev, and B. A. Hamilton. "A location-based mechanism for mobile device security". In: *2009 WRI World Congress on Computer Science and Information Engineering, CSIE 2009* 1 (2009), pp. 99–104.

[21] M. Karnan, M. Akila, and N. Krishnaraj. "Biometric personal authentication using keystroke dynamics: A review". In: *Applied Soft Computing Journal* 11.2 (2011), pp. 1565–1573. ISSN: 15684946.

[22] E. Khoury, L. El Shafey, C. McCool, M. Günther, and S. Marcel. "Bi-modal biometric authentication on mobile phones in challenging conditions". In: *Image and Vision Computing* 32.12 (2013), pp. 1147–1160. ISSN: 02628856. DOI: 10.1016/j.imavis.2013.10.001. URL: http://dx.doi.org/10.1016/j.imavis.2013.10.001.

[23] S. Kurkovsky and E. Syta. "Approaches and issues in location-aware continuous authentication". In: *Proceedings - 2010 13th IEEE International Conference on Computational Science and Engineering, CSE 2010* (2010), pp. 279–283.

[24] M. Lee, G. Kim, S. Park, S. Jun, J. Nah, and O. Song. "Efficient 3G/WLAN Interworking Techniques for Seamless Roaming Services with Location-Aware Authentication". In: *NETWORKING 2005. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems* 3462 (2005), p. 441. ISSN: 03029743.

[25] T. K. M. Lee, M. Belkhatir, and S. Sanei. "A comprehensive review of past and present vision-based techniques for gait recognition". In: *Multimedia Tools and Applications* (2013), pp. 1–37. ISSN: 13807501.

[26] S. Mukherjee. "Differential Evolution Based Score Level Fusion For Multimodal Biometric Systems". In: (2014).

[27] M Nageshkumar, P. Mahesh, and M. Shanmukha Swamy. "An Efficient Secure Multimodal Biometric Fusion Using Palmprint and Face Image". In: 2 (2009), pp. 49–53. arXiv: 0909.2373. URL: http://cogprints.org/6696/.

[28] K. Niinuma and A. Jain. "Continuous user authentication using temporal information". In: *Defense, Security, and Sensing* (2010), pp. 76670L–76670L–11. ISSN: 0277786X.

[29] W. Paper. "Invisible challenges". In: (2014).

[30] U. Peisert, Sean (UC Davis and Berkeley Lab California, U. Talbot, Ed (UC Davis California, and U. Kroeger, Tom (Sandia National Laboratories California. "Principles of Authentication". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8112 LNCS.PART 2 (2013), pp. 390–399. ISSN: 03029743. DOI: 10.1007/978-3-642-53862-9-50.

[31] M. Poveda-villalón, M. C. Suárez-figueroa, and R. García-castro. "A Context Ontology for Mobile Environments". In: ().

[32] "Profiles for the OASIS Security Assertion Markup Language (SAML)". In: *Language* 16.3 (2005), p. 66.

[33] Q. Pu, S. Gupta, S. Gollakota, and S. Patel. "Whole-home gesture recognition using wireless signals". In: *Acm Mobicom* (2013), pp. 485–486.

[34] R. A. A. D. M. J. W. Raymond. "LOCATION-SENSITIVE SECURITY LEVELS AND SETTING PROFILES BASED ON DETECTED LOCATION". In: (2014).

[35] K. Revett. "Cognitive biometrics: a novel approach to person authentication". In: *International Journal of Cognitive Biometrics* 1.1 (2012), p. 1. ISSN: 2042-6461.

[36] G. G. S. Rick Lehtinen, Deborah Russell. *This is the must-have book for a must-know field. Today, general security knowledge is mandatory, and, if you who need to understand the fundamentals*, June. 2006. ISBN: 9780596006693.

[37] A. Ross and A. K. Jain. "MULTIMODAL BIOMETRICS : AN OVERVIEW". In: September (2004), pp. 1221–1224.

[38] N. Sae-Bae, K. Ahmed, K. Isbister, and N. Memon. "Biometric-rich gestures". In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12* (2012), p. 977.

[39] J. Seto, Y. E. Wang, X. Lin, and S. Member. "User-Habit-Oriented Authentication Model : Toward Secure , User-Friendly Authentication for Mobile Devices". In: 3.1 (2015).

[40] E. Shi, Y. Niu, M. Jakobsson, and R. Chow. "Implicit authentication through learning user behavior". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 6531 LNCS (2011), pp. 99–113. ISSN: 03029743.

[41] R. Shirey. *Internet Security Glossary*. May 2000.

[42] T. G. S. M. Standard. "InfoSec Reading Room tu , Aholl rights". In: May (2001).

[43] P. Standards. "Standards for Security Categorization of Federal Information and Information Systems". In: *Security Management* FIPS PUB 1.February (2004).

[44] D. Todorov. *Mechanics of User Identification and Authentication: Fundamentals of Identity Management*. Boca Raton : Auerbach Publications, 2007, 2007, p. 728. ISBN: 9781420052190.

[45] M. Turk. "Multimodal interaction: A review". In: *Pattern Recognition Letters* 36.1 (2014), pp. 189–195. ISSN: 01678655.

[46] J. a. Unar, W. C. Seng, and A. Abbasi. "A review of biometric technology along with trends and prospects". In: *Pattern Recognition* 47.8 (2014), pp. 2673–2688. ISSN: 00313203.

[47] I. T. Union. "Data Communication Networks: Open System Interconnection (OSI); Security, Structure and Application". In: *The International Telegraph and Telephone Consultative Comittee, Recommendation X.800* (1991).

[48] P. Verlinde, G. Chollet, and M. Acheroy. "Multi-modal identity verification using expert fusion". In: *Information Fusion* 1.1 (2000), pp. 17–33. ISSN: 15662535. DOI: 10.1016/s1566-2535(00)00002-6. URL: http://dx.doi.org/10.1016/s1566-2535(00)00002-6.

[49] L. B. W. Stallings. *Computer Security - Principles and Practice Publisher : Edition : Pub Date :* 3rd ed. Pearson - Prentice Hall, 2015.

[50] A. B. Whitepaper. "Reducing SMS Authentication by a Factor of 5 A BioCatch WhitePaper". In: July (2014).