Università di Firenze, Università di Perugia, INdAM consorziate nel CIAFM

DOTTORATO DI RICERCA
IN MATEMATICA, INFORMATICA, STATISTICA

CURRICULUM IN INFORMATICA
CICLO XXXI

Sede amministrativa Università degli Studi di Firenze
Coordinatore Prof. Graziano Gentili

Design and Evaluation of Multi-Biometric Approaches for Continuous Authentication and Non-Repudiation in Critical Services

Settore Scientifico Disciplinare INF/01

Tutor e Co-Tutor
Prof. Andrea Bondavalli
Dr. Andrea Ceccarelli
Prof. Ariadne Carvalho

Dottorando:
Enrico Schiavone

Coordinatori
Prof. Graziano Gentili
Prof. Cristina Pinotti

Anni 2015/2018

Enrico Schiavone: *Design and Evaluation of Multi-Biometric Approaches for Continuous Authentication and Non-Repudiation in Critical Services* , Dottorato in Matematica, Informatica, Statistica, © Anno Accademico 2017-2018

ABSTRACT

In our society, nowadays pervaded by ICT, security services as *authentication* and *non-repudiation* play a very important role. In some critical systems and applications, in fact, it is mandatory that only authorized users are allowed to have services and functionalities at their disposal, especially in working environments where operators are in charge of analyzing sensitive data. Moreover, it is also essential that once users or entities make use of critical services, taking decisions for which they are directly responsible and which may have serious implications on company's assets or even citizen's safety, cannot subsequently deny their involvement. Nevertheless, in order to avoid possible intrusions, it is fundamental that these two services are provided *continuously*, that is for the whole session duration. In addition, they have to preserve usability and to avoid disturbing the user activity, otherwise they may result ineffective or even counter-productive.

This Thesis aims to provide a solution to the problem of *continuous authentication* and describes the architecture design, protocol definition, prototyping and implementation of a multi-biometric system meant for this purpose. A risk assessment supports its design process from a security point of view, while an extensive and sound testing campaign, conducted with the involvement of real users, validates it from the usability perspective.

This work also coins the term *continuous non-repudiation*, and proposes three alternative approaches for the seamless provision of this service. All of them share a common multi-biometric identity verification core, while they differ in terms of the way they technically address non-repudiation, as well as in the architectures and protocols. In particular, each solution introduces an improvement constituted by digital signature, biometric signature, and blockchain technology respectively. The latter, called Block-CNR, has been designed and implemented after having evaluated the usefulness of distributed ledgers for our purposes and takes into account also risks and common issues of adopting this technology.

ACKNOWLEDGMENTS

I would like to sincerely thank my supervisor, Prof. Andrea Bondavalli, for supporting and motivating me during my research, for his always honest and constructive advice that really helped me finding solutions to problems and drawing lessons useful for a lifetime.

I express my deepest gratitude to Andrea Ceccarelli, for his tireless contribution with comments and ideas on many topics of this Thesis. Sincere thanks also to the other present and former members of the Resilient Computing Lab (RCL) research group: Paolo Lollini, Leonardo Montecchi, Tommaso Zoppi, Mohamad Gharib, Marco Mori, Hanna Kavalionak, and Simonetta Tegliai for the stimulating discussions and collaborations, as well as for the good time spent together.

My great appreciation goes also to all the people who participated in the National and International research projects in which I have been involved, especially the EU FP7 "DEVASSES" project, which has been a really enriching experience. In particular, I would like to thank Prof. Ariadne Carvalho, for her precious support during my secondments in Brazil, and her valuable contribution on the dissertation topics. I would also like to thank all the students, researchers, professors, and friends from the University of Campinas, without whose participation and help, the test campaign described in this Thesis would not have been possible.

I would like to thank the reviewers for their precious suggestions.

I am also thankful to all the people who I have been working with in these years in the PhD room, at Resiltech s.r.l, and everywhere else. Each of them really contributed to a better personal and professional experience.

Last but not the least, thanks to my parents, my brothers and my closest friends for supporting me throughout my studies and my life in general.

LIST OF PUBLICATIONS

The following is a list of relevant publications related to the research activity described in this thesis.

ICA3PP 2015: E. Schiavone, A. Ceccarelli, and A. Bondavalli. *"**Continuous User Identity Verification for Trusted Operators in Control Rooms**"*, In 15th International Conference on Algorithms and Architectures for Parallel Processing, (co-located with PRDC 2015, the 21st IEEE Pacific Rim International Symposium on Dependable Computing). Springer International Publishing, LNCS Volume 9532, 2015, Pages 187-200.

LADC 2016: E. Schiavone, A. Ceccarelli, A. Bondavalli, and A. Carvalho. *"**Usability Assessment in a Multi-Biometric Continuous Authentication System**"*, IEEE Proceedings of LADC 2016, 7th Latin-American Symposium on Dependable Computing, Pages 43-50.

ITASEC 2017: E. Schiavone, A. Ceccarelli, and A. Bondavalli. *"**Risk Assessment of a Biometric Continuous Authentication Protocol for Internet Services**"*, ITASEC17, 1st Italian Conference on Cybersecurity. CEUR Workshop Proceedings, Vol. 1816, 2017, Pages 53-65.

ARES 2017: E. Schiavone, A. Ceccarelli, and A. Bondavalli. *"**Continuous Biometric Verification for Non-Repudiation of Remote Services**"*, ACM International Conference Proceedings of the 12th International Conference on Availability, Reliability and Security, ARES 2017.

SRDS 2018:, M. Staderini, E. Schiavone, A. Bondavalli. *"**A Requirements-Driven Methodology for the Proper Selection and Configuration of Blockchains**"*, In 2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS), Pages 201-206.

LADC 2018:, G. Morganti, E. Schiavone, and A. Bondavalli. *"**Risk Assessment of Blockchain Technology**"*, to appear in IEEE Proceedings of 8th Latin-American Symposium on Dependable Computing, LADC, 2018.

The researches conducted so far originated also the following papers, which have been submitted and are currently under revision.

IJCCS 2018, E. Schiavone, A. Ceccarelli, A. Bondavalli, and A. Carvalho. ***"Design, Implementation, and Assessment of a Multi-biometric Continuous Authentication System"***, International Journal of Critical Computer-based Systems, 2018.

Finally, other articles which appeared in International Student Forums and Symposia are reported here.

DSN 2016: E. Schiavone, ***"Providing Continuous Authentication and Non-Repudiation Security Services"***, Student Forum of the 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN.

SRDS 2016: E. Schiavone, A. Ceccarelli, and A. Bondavalli ***"Continuous Authentication and Non-repudiation for the Security of Critical Systems"***, PhD Forum. In Proceedings of the IEEE 35th Symposium on Reliable Distributed Systems, SRDS, 2016. Pages 207-208.

MINISY 2017: E. Schiavone. ***"Securing Critical Systems through Continuous User Authentication and Non-repudiation"***, Proceedings of the 24th PhD Mini-Symposium of the Department of Measurement and Information Systems, Budapest University of Technology and Economics. 2017.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Part I

INTRODUCTION, BASICS AND STATE OF THE ART

1 INTRODUCTION

1.1 MOTIVATION

Information and Communication Technology (ICT) pervades modern society to the extent that we massively rely on it from private life to business. Today, users and operators can share confidential data, perform financial transactions, or remotely execute critical operations in real-time.

However, the need for security has gone hand in hand with the technological progress, and our reliance on ICT strictly depends on it.

In many critical systems and applications, it is fundamental that only authorized users are allowed to interact with a machine. In some working environments, in fact, users are in charge of analyzing potentially sensitive data, taking decisions they are directly responsible for, and which may have serious implications on company's assets or even on citizen's safety. Their workstations should be properly protected in order to prevent undesired consequences.

Authentication, which is intended to provide assurance in the identity claimed by an entity, is the security service designed for this purpose. Traditional authentication approaches are knowledge-based and take advantage of passwords or PINs; alternative solutions have been proposed, either possession-based (e.g., security token), or making use of biometric traits.

In all cases, when the identity verification is performed as a single occurrence during the login phase, and no identity checks are performed during sessions, unauthorized people may take physical control of the device. For instance, in an office environment if a worker leaves the computer unattended without logging out e.g., for a short break or to reach the printer, insiders may intervene accessing, modifying or deleting sensitive information, or even introducing vulnerabilities.

In order to address this issue, it is desirable to have user identity continuously verified, for the whole duration of a session. However, repeatedly asking for passwords and secrets over time, requires user active participation: it would disturb operations and reduce system's usability. It is well-known, in fact, that usability and security are often seen as competing goals. Improving usability is sometimes considered as improving vulnerabilities, to the extent that it has also been perceived as helping the attacker.

Solutions based on biometric authentication have been proposed in the literature, and some have the potential to continuously verify the identity of the user. Thanks to this approach, known as biometric *continuous authentication*, user identity verification is no longer a single occurrence, but a continuous process. Furthermore, it is commonly agreed that the use of multiple biometric traits, properly combined, can improve the performance of the identity verification process. In addition, appropriate sensors, together with specific design choices, permit to acquire biometric traits transparently, i.e., without the active involvement of the user.

However, there is a real lack of studies with emphasis and focus on end-users. In fact, the evaluation of a proposed system or framework is often conducted as a simulation, rarely with human involvement, and almost never through a proper usability assessment. Nevertheless, in our opinion, when introducing a new approach, it is fundamental to address not only technical issues, but to consider also the effects of this innovation on humans, on their thoughts, and perceptions.

Therefore, a proper usability testing campaign may combine measurements on effectiveness and efficiency of the system with an analysis of user satisfaction, e.g., through interviews or questionnaires.

Authentication is not the only security service which plays a very important role in critical ICT services. The ability to demonstrate that users or entities requested specific functionalities, or performed certain actions, among other abilities, is also very useful. In fact, when a dispute arises or an error occurs, people may attempt to deny their involvement, and to repudiate their behavior. For instance, customers may disclaim a withdrawal from their account, or a payment with their own credit card, when they actually did it.

In addition to users denying their usage of a service, there have been cases of malicious operators or disreputable service providers. These two are both cases of repudiation, which can be defined as the denial of having participated in – all or part of – an action, by one of the entities involved.

Non-repudiation, consequently, is the ability to protect against such denial. A non-repudiation mechanism has to provide evidences in order to clarify responsibilities, and to guarantee the establishment of the facts, if necessary, even in front of a court of law. Therefore, a non-repudiation service can be useful both as a mean to obtain accountability, and as a deterrent for deliberate misbehaviors. In the two situations previously reported, a non-repudiation mechanism would prevent that the fraudulent customer succeeds in denying a transaction, or would help an innocent client to protect his investments.

In the literature, the problem of repudiation has been mainly tackled in the electronic transactions context. A transaction is a one-shot information exchange, which can be defined as transferring of a message from A to B. Transactions' security is useful in digital contract signing, e-commerce, or electronic voting, in which communicating parties may try to cheat each other. The issue of transactions' non-repudiation is usually addressed exploiting digital signatures. In order to perform a digital signature, users need to keep secret of their private key. It can be stored in a computer, or more frequently in a smart card or USB device, and protected by a secret (password, or PIN). Items and secrets must not be stolen, lost or forgotten. But what happens if the information that needs to be signed, is not a single transaction, but a continuous information flow? For instance, the access to the private area on a web service for online banking, or VoIP calls in case of verbal contracts signing, and so on, are different scenarios which may benefit from non-repudiation. Asking for the password or secret for each single transaction would drastically reduce the usability of the service, and may end up frustrating the user. Conversely, requesting the secret only at the first login does not guarantee authenticity of the user for the whole session. For applications in which the communication is confidential, or involves information regarding assets which need to be protected with high assurance, authentication has to be repeated frequently. Otherwise, insiders may be able to interfere, taking advantage of the parties, or causing errors and consequent disputes hard to be solved.

In those cases, new solutions are necessary in order to provide a non-repudiation service for the entire duration of the session. Therefore, there is a need for a *continuous non-repudiation* service. Many research questions arise here. Can continuous authentication mechanisms be complemented with non-repudiation? Does biometric authentication provide sufficient and undeniable evidence of user's participation in an action? And then, thanks to the recent technological advancements, – e.g., the introduction of distributed ledgers – , is it possible to reinforce the traditional non-repudiation solutions?

1.2 CONTRIBUTION AND THESIS ORGANIZATION

Part I In the first part of the Thesis, we introduce basic concepts regarding dependability and security, as well as fundamentals of authentication, and non-repudiation. Then, targeting a continuous authentication system that is both usable and incurs in little system overhead, we address the discipline of biometrics, and review the state of the art, comparing the most relevant multi-biometric solutions from academia and in commerce.

Part II In the second part, we perform a risk assessment of an interesting state-of-the-art approach. The assessment is part of a design and requirements definition process, and is performed in order to guide the development of our own authentication and non-repudiation solutions. We propose a multi-biometric continuous authentication mechanism, which integrates face, fingerprint and keystroke recognitions, and removes the necessity of conscious human-computer interactions. Data is transparently acquired by the workstation and transmitted to an authentication server, which performs the identity verification. In case of successful verification, the authentication server permits the establishment of a user session. Then it calculates and updates a trust level that decreases as time passes; the session expires when such level becomes lower than a predefined threshold. We evaluate the system through a usability testing campaign, involving a population of 60 users selected amongst academia. We devise experiments where participants are asked to perform different office processing tasks. We collect users' feedbacks regarding system usability and their satisfaction.

During the tests, we also measured the acceptance and rejection rates of the face, fingerprint and keystroke subsystems, and of the integrated system, to evaluate the effectiveness of the authentication solution. Furthermore, we considered system efficiency measuring the time interval during which a legitimate user remains authenticated, and the window of time needed by the system to reject an impostor.

Finally, the trade-off between usability and security is quantified. Results show that even taking into account usability as a primary goal, security of users' workstations is increased. We publish a repository of the log files recorded by the system, which, as far as we know, is the first public dataset on logs of a continuous authentication service available to researchers. The supplementary data consists also in detailed questionnaire results regarding user satisfaction.

Part III In the last part of the Thesis, we address the problem of *continuous non-repudiation* through multi-biometric continuous verification of identities. The idea is to introduce, in the architecture of our continuous authentication solutions, the modifications needed to provide also continuous non-repudiation.

We propose three alternative solutions: the first two are respectively called *DS-CNR (Digital Signature based Continuous Non-Repudiation)*, and *BS-CNR (Biometric Signature based Continuous Non-Repudiation)*. They mainly differ in the generation and handling of the cryptographic keys and in the underlying architecture. We assume that the communications between the entities are encrypted, and the involved third parties are trustable. Under the stated assumptions, both solutions offer a non-repudiation service for a continuous information flow scenario between a client and a remote internet service, protecting both parties for the whole duration of the session. We also discuss the technological readiness of biometrics to offer non-repudiability. We show that algorithms that possess high verification accuracy already exist in the literature. In our opinion, if those algorithms are properly integrated in a multi-biometric system, and possibly coupled with other security mechanism, biometrics is ready to offer non-repudiation.

Finally, after the birth of blockchain, which has recently spread to the extent that resembles a real fever, we investigate if it can provide advantages in offering a non-repudiation service (e.g., reducing the number of third parties to trust), and we show which of its multiple configurations would better fit. For this purpose, we propose a methodology for the requirements-driven selection and configuration of blockchains. We also perform a risk assessment in order to be aware of its threats and to identify existing countermeasures.

After these steps, we introduce the third version of the continuous non-repudiation protocol, named *Block-CNR (Blockchain based Continuous Non-Repudiation)*, which benefits from the properties provided by the distributed ledger technology, as immutability, transparency and intrinsic non-repudiability. We also show, with an implementation, that creating a blockchain, and configuring it for the storage of non-repudiation evidences, is actually feasible.

2 DEPENDABLE AND SECURE SYSTEMS

This Chapter describes the context in which the work concerning this Thesis has been developed. Section 2.1 explains what a dependable and secure system is; Sections 2.2, and 2.3 respectively provide the basic terminology and concepts regarding threats and means to attain dependability and security. Finally, Sections 2.4 and 2.5 respectively introduce authentication and non-repudiation, two security services particularly relevant for this Thesis, and the traditional approaches to provide them.

2.1 DEPENDABILITY, SECURITY AND THEIR ATTRIBUTES

According to the taxonomy of [1], a **system** is an entity that interacts with other entities, i.e., other systems, including hardware, software, humans, and the physical world. The *service* delivered by a system is its behavior as it is perceived by its user(s); a user is another system that receives service from the provider. A system may sequentially or simultaneously be a provider and a user with respect to another system, i.e., deliver service to and receive service from that other system.

2.1.1 *Basic Definitions and Attributes*

The original definition of **dependability** [1] is:

Definition 1. *The ability to deliver service that can justifiably be trusted.*

This definition stresses the need for justification of trust. An alternate definition proposed in [1], that provides the criterion for deciding if the service is dependable, is the the following.

Definition 2. *Dependability of a system is the ability to avoid service failures that are more frequent and more severe than is acceptable.*

As reported in [1], other different definitions of dependability exist in the standards. The the ISO definition [2], clearly centred upon availability, is:

Definition 3. *The collective term used to describe the availability performance and its influencing factors: reliability performance, maintainability performance and maintenance support performance.*

Instead, the IEC defines deependability as [3]:

Definition 4. *The extent to which the system can be relied upon to perform exclusively and correctly the system task(s) under defined operational and environmental conditions over a defined period of time, or at a given instant of time.*

As developed over the past decades, dependability is an integrating concept that encompasses the following attributes:

- *Availability*: readiness for correct service;
- *Reliability*: continuity of correct service;
- *Safety*: absence of catastrophic consequences on the user(s) and the environment;
- *Integrity*: absence of improper system alterations;
- *Maintainability*: ability to undergo modifications and repairs.

When addressing **security**, an additional attribute which has great prominence is:

- *Confidentiality*: the absence of unauthorized disclosure of information.

Security has not been characterized as a single attribute of dependability. In fact, it is a composite of the attributes of confidentiality, integrity, and availability, requiring the concurrent existence of availability for authorized actions only, confidentiality, and integrity with *improper* meaning *unauthorized* [1]. Figure 1 summarizes the relationship between dependability and security in terms of their principal attributes. This is in agreement with the usual definitions of security [4, 5], that propose it as a composite notion, namely:

Definition 5. *The combination of confidentiality, the prevention of the unauthorized disclosure of information, integrity, the prevention of the unauthorized amendment or deletion of information, and availability, the prevention of the unauthorized withholding of information.*

The unified definition for security in [1] is:

Definition 6. *The absence of unauthorized access to, or handling of, system state.*

The attributes of dependability and security that have been defined in this Section may be of varying importance depending on the application intended for the given computing system: availability, integrity, and maintainability are generally required, although to a varying degree depending on the application, whereas reliability, safety, and confidentiality may or may not be required according to the application. The extent to which a system possesses the attributes of dependability and security should be considered in a relative, probabilistic, sense, and not in an absolute, deterministic sense: due to the unavoidable presence or occurrence of faults, systems are never totally available, reliable, safe, or secure [1].

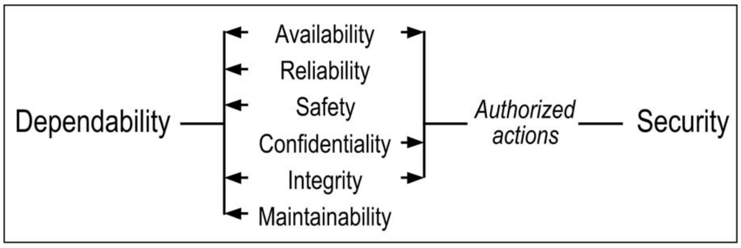


Figure 1: Dependability and security attributes and their relationship. [1]

Secondary Attributes

Besides the attributes defined and discussed above in this Section, other, secondary, attributes can be defined, which refine or specialize the primary attributes. The notion of secondary attributes is especially relevant for security, and is based on distinguishing among various types of information. Examples of such secondary attributes, which are particularly relevant for the purpose of this Thesis, are [1]

- *Accountability*: availability and integrity of the identity of the person who performed an operation;
- *Authenticity*: integrity of a message content and origin, and possibly of some other information, such as the time of emission;
- *Non-repudiability*: availability and integrity of the identity of the sender of a message (non-repudiation of the origin), or of the receiver (non-repudiation of reception[1]).

2.2 THREATS TO DEPENDABILITY AND SECURITY: FAULTS, ERRORS, FAILURES

Correct service is delivered when the service implements the system function. On the contrary, a **failure**, is an event that occurs when the delivered service deviates from correct service, as shown in Figure 2. A service fails either because it does not comply with the functional specification, or because this specification did not adequately describe the system function.

A service failure is a transition from correct service to incorrect service, i.e., to a state that does not implement the system function. The period of delivery of incorrect service is a service outage. The transition from incorrect service to correct service is a service restoration. The deviation from correct service may assume different forms that are called service failure modes and are ranked according to failure severities [1].

[1] sometimes referred as non-repudiation of destination

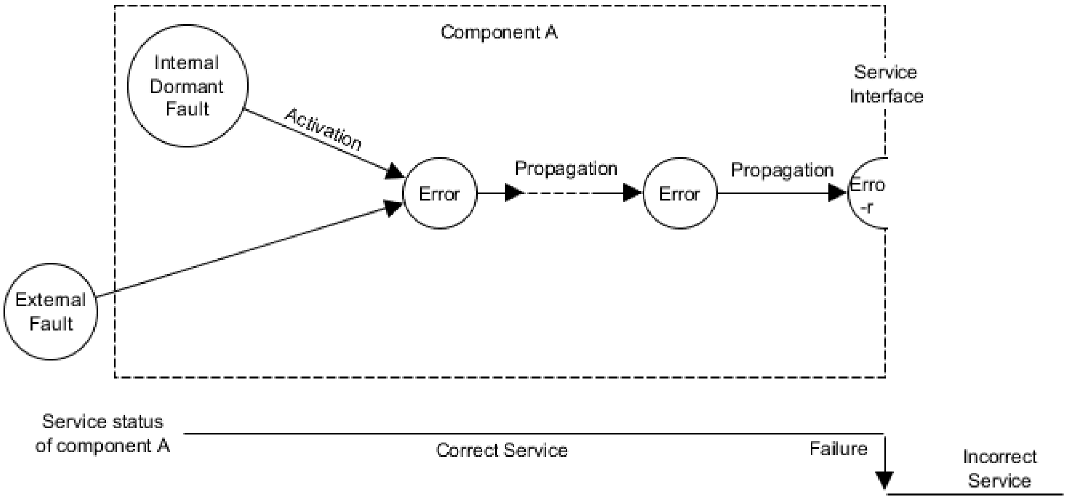


Figure 2: Error Propagation scheme.

Since a service is a sequence of the system's external states, a service failure means that at least one (or more) external state of the system deviates from the correct service state. The deviation is called an *error*. The adjudged or hypothesized cause of an error is called a **fault**. Faults can be internal or external of a system. The prior presence of a vulnerability, i.e., an internal fault that enables an external fault to harm the system, is necessary for an external fault to cause an error and possibly subsequent failure(s). In most cases, a fault first causes an error in the service state of a component that is a part of the internal state of the system and the external state is not immediately affected.

For this reason, the definition of an **error** is the part of the total state of the system that may lead to its subsequent service failure. It is important to note that many errors do not reach the system's external state and cause a failure. A fault is *active* when it causes an error, otherwise it is *dormant*.

When the functional specification of a system includes a set of several functions, the failure of one or more of the services implementing the functions may leave the system in a degraded mode that still offers a subset of needed services to the user. The specification may identify several such modes, e.g., slow service, limited service, emergency service, etc. A such system is said having suffered a partial failure of its functionality or performance.

2.2.1 *Faults and Errors Classification*

All faults that may affect a system during its life are classified according to eight basic viewpoints, leading to the elementary fault classes, as shown in Figure 3. If all combinations of the eight elementary fault classes were possible, there would be 256 different combined fault classes. However, not all criteria are applicable

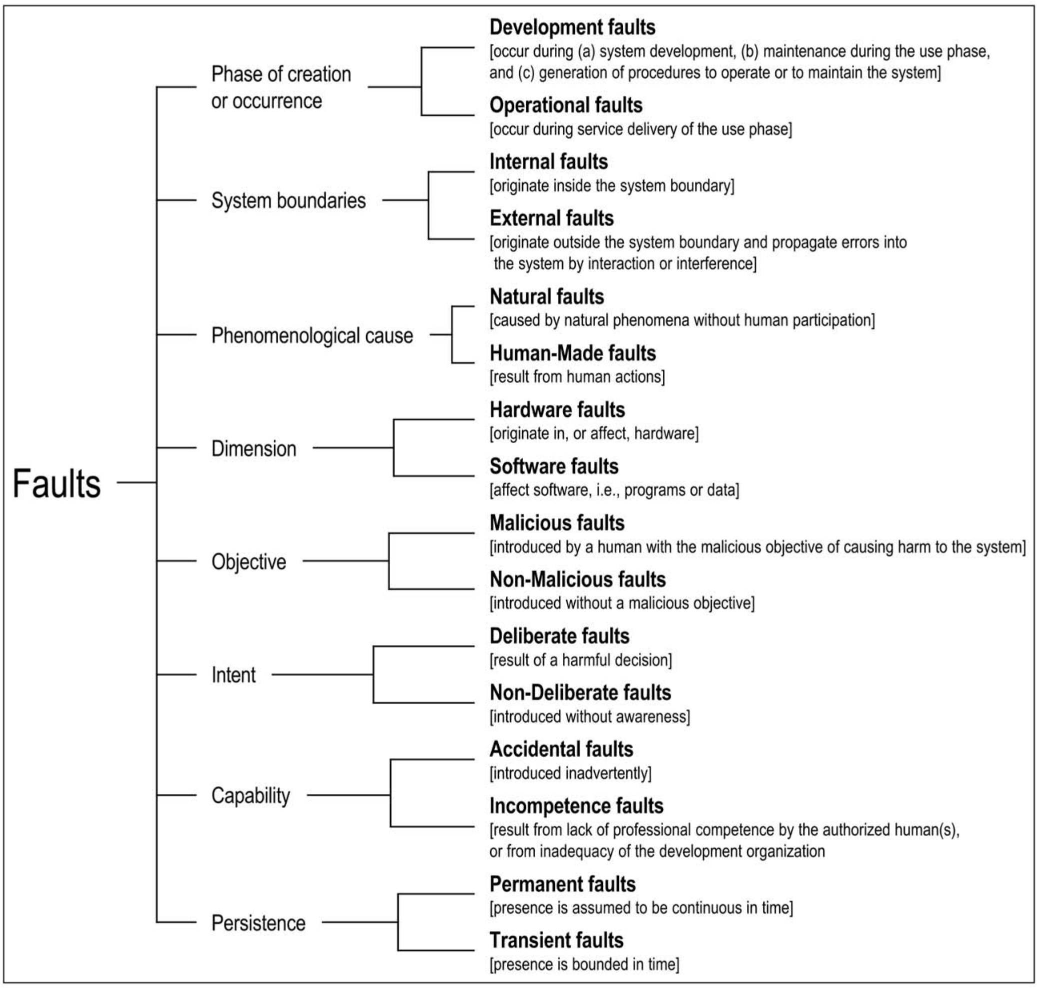

Faults
Phase of creation or occurrence
Development faults
[occur during (a) system development, (b) maintenance during the use phase, and (c) generation of procedures to operate or to maintain the system]
Operational faults
[occur during service delivery of the use phase]
System boundaries
Internal faults
[originate inside the system boundary]
External faults
[originate outside the system boundary and propagate errors into the system by interaction or interference]
Phenomenological cause
Natural faults
[caused by natural phenomena without human participation]
Human-Made faults
[result from human actions]
Dimension
Hardware faults
[originate in, or affect, hardware]
Software faults
[affect software, i.e., programs or data]
Objective
Malicious faults
[introduced by a human with the malicious objective of causing harm to the system]
Non-Malicious faults
[introduced without a malicious objective]
Intent
Deliberate faults
[result of a harmful decision]
Non-Deliberate faults
[introduced without awareness]
Capability
Accidental faults
[introduced inadvertently]
Incompetence faults
[result from lack of professional competence by the authorized human(s), or from inadequacy of the development organization
Persistence
Permanent faults
[presence is assumed to be continuous in time]
Transient faults
[presence is bounded in time]

Figure 3: The main fault classes. [1]

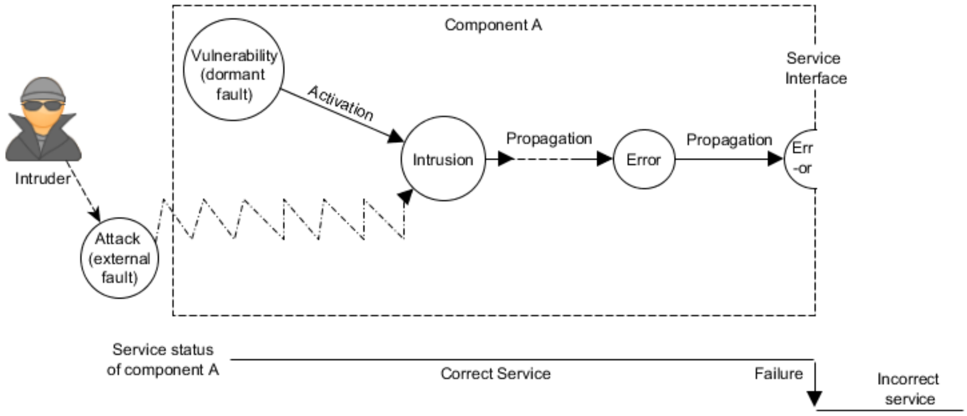


Figure 4: The AVI model. Image reworked from [103]

to all fault classes; for example, natural faults cannot be classified by objective, intent, and capability. Possible combined fault classes are shown to belong to three major partially overlapping groupings:

- *development faults* that include all fault classes occurring during development;
- *physical faults* that include all fault classes that affect hardware;
- *interaction faults* that include all external faults.

Knowledge of all possible fault classes allows the user to decide which classes should be included in a dependability and security specification.

2.2.2 *Security Terminology and the Attacks, Vulnerabilities and Intrusions (AVI) Model*

As discussed so far, the potential causes of unexpected behaviors that may make a system undependable are: faults, errors and failures. In particular, considering *security*, it is possible to devise a specialization of the *fault-error-failure* chain, the so-called *Attack-Vulnerability-Intrusion (AVI)* fault model, shown in Figure 4. In this Section, we describe the AVI model and introduce some terminology that will be useful throughout the Thesis. Some concepts will be further addressed in specific Chapters.

Such model considers *vulnerabilities* as internal and dormant faults, introduced either at design stage or during system operation. According to [7], a vulnerability is a flaw or weakness in a system's design, implementation, or operation and management that could be exploited to violate the system's security policy.

Attacks are malicious faults, typically external and generated with the aim of exploiting one or more vulnerabilities of the system. In [7], an attack is defined

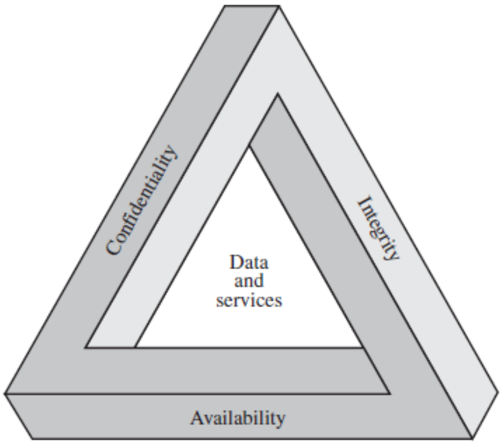


Figure 5: The CIA triad: dependability attributes involving security. [7]

as an assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system.

A successful attack provokes an *intrusion*, which generally is characterized by an erroneous state of the system (e.g., the attacker obtains an unjustified permission to access some system configuration file). Without *countermeasures* in place, capable of coping with the intrusion, a failure in the system is likely to occur.

The dependability attributes involving security – often referred as CIA triad, as shown in Figure 5– can be threaten by different kinds of attack, attributable to the following four main categories [7, 8]:

- *disclosure*: unauthorized access to information (e.g., exposure, interception, inference, intrusion);
- *deception*: acceptance of false data (e.g., masquerade, falsification, repudiation);
- *disruption*: interruption or prevention of correct operation (e.g., incapacitation, corruption, obstruction);
- *usurpation*: unauthorized control of some part of a system (e.g., misappropriation, misuse).

However, it is necessary to specify that some attacks can have consequences belonging to multiple categories contemporaneously.

Sometimes, the consequences we just mentioned could be not the result of a deliberate attempt from an attacker, but come from environmental characteristics.

Under different circumstances, the threat comes from the legitimate user: this takes the name of *insider threat* [104, 105].

After this brief overview on the threats to dependability and security, it is useful to discuss some countermeasures which can reduce the vulnerability in a system. In the following, we refer to both general strategies and specific mechanisms particularly relevant for the Thesis.

2.3 THE MEANS TO ATTAIN DEPENDABILITY AND SECURITY

Over the course of the past decades, many means have been developed to attain the various attributes of dependability and security. Those means can be grouped into four major categories:

- *Fault prevention* means to prevent the occurrence or introduction of faults.
- *Fault tolerance* means to avoid service failures in the presence of faults.
- *Fault removal* means to reduce the number and severity of faults.
- *Fault forecasting* means to estimate the present number, the future incidence, and the likely consequences of faults.

Fault prevention and fault tolerance aim to provide the ability to deliver a service that can be trusted, while fault removal and fault forecasting aim to reach confidence in that ability by justifying that the functional and the dependability and security specifications are adequate and that the system is likely to meet them.

Security Policies

A security policy can be considered a mean to obtain dependability and security. According to [7], it is a set of rules and practices that specify or regulate how a system or organization provides security services to protect sensitive and critical system resources. In [1] a security policy is a set of security-motivated constraints, that organizations or systems (e.g., computer systems) have to adhere to.

The enforcement of such rules, practices and constraints may be via technical, management, and/or operational controls, and the policy may lay down how these controls are to be enforced. In practice, there may be a hierarchy of security policies, related to a hierarchy of systems, for example. If presence of different security issues, there may be separate but related policies, e.g., a policy regarding the controlled public disclosure of company information, one on physical and networked access to the company's computers. Some computer security policies include constraints on how information may flow within a system as well as constraints on system states.

Dependability and security classes are generally defined via the analysis of failure frequencies and severities, and of outage durations, for the attributes that are of concern for a given application. This analysis may be conducted directly or indirectly via *risk assessment* (this concept, particularly relevant for the Thesis, will be further addressed in Section 5.1).

The variations in the emphasis placed on the different attributes directly influence the balance of the techniques (fault prevention, tolerance, removal, and forecasting) to be employed in order to make the resulting system dependable and secure. This problem is all the more difficult as some of the attributes are conflicting (e.g., availability and safety, security and usability), necessitating that trade-offs be made.

Security Services and Mechanisms

In [7], a *security mechanism* is described as a mechanism that is designed to detect, prevent, or recover from an attack. Thus it can be seen as a mean specifically directed to external faults.

A *security service*, instead, enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

According to the standards ITU-T, Rec. X.800 [106] and IETF RFC 2828 [107], the data security is pursued by assuring, among others, the following services: *authentication*, and *non-repudiation*.

2.4 AUTHENTICATION

2.4.1 *Definitions and Approaches*

Authentication can be defined as the process that provides assurance in the claimed identity of an entity [38]. Authenticating a user or an application thus means verifying their *authenticity* property. This is possible thanks to a piece of information typically called authentication *factor* [39]. In literature, many types of authentication factors have been proposed, and they are usually divided in the following three categories:

- *Knowledge* factors: something the user knows (e.g., passwords, PINs).

- *Possession* factors: something the user has (e.g., passports, private keys, security tokens).

- *Inherence* factors: something the user is or does (e.g., physiological or behavioral biometrics).

This process is composed of two consecutive steps: *registration* (or *enrollment*) and *verification*. Registration consists in storing an authentication factor associated with the user identity, and which will be verified in the subsequent step (on verification).

Traditionally, most of the access controls in ICT – as login operations for accessing Internet services or local devices – are performed through a knowledge factor. This is probably because of simplicity, as it can be easily implemented through software, without the need for specific hardware or sensors. In those cases, however, the chosen password impacts on the dependability of the authentication; for some scenarios, a normal password that the user is able to remember, can be enough. In fact, many users choose memorable words or alphanumeric identifiers, e.g., proper names or birth dates of their relatives, of celebrities, or nouns that can be found on a dictionary. However, those choices are probably the worst from a security point of view: guessing the noun, or trying with a brute force strategy (as in the *dictionary attack*) may lay the foundations for a successful *spoofing*. Moreover, many users choose the same password for accessing several different services, and, consequently, a successful attack could mean several intrusions. In order to solve this problem, a long and complex password may be sufficient; however, it becomes hard to remember and the user is often forced to write or copy it somewhere (typically, on a post-it attached on the screen). In this situation, it is necessary to protect the location where the credentials are stored, which means simply displace the problem elsewhere. Another limitation of knowledge factors appears when the secret is intercepted or deduced: the system cannot verify if the user is actually the legitimate one or not, and repudiating actions performed by and intruder becomes very difficult.

The so-called object-based authentication constitutes a viable alternative and exploits the possession of physical objects from the user. It is mainly adopted in governmental or financial applications, which for instance make use of badges, tokens, credit cards, licence, passports, and so on. The main issue of this approach is represented by the possibility that those objects can be lost or, even worse, stolen, thus preventing the system from granting authenticity and non-repudiation properties.

Combining multiple traditional authentication factors, gives rise to the so-called *strong authentication*, which is quite common, e.g., for cash withdrawals at the automated teller machine (ATM): only customers in possession of the card and having knowledge of the corresponding PIN are able of performing the financial transactions.

The third family of factors that can be used for authentication are biometric data, typically diffused in application domains as forensic science, medical, or criminal investigation. In the last decade, biometric sensors have also been integrated in common notebooks and smart phones, and consequently the number of application scenarios is growing. The main advantages of biometric traits in comparison with other authentication factors are the following: they are hard to be copied or shared, cannot be lost or forgotten and normally they need the user to be physically present at the moment when, and in the place where, the authentication is being performed.

Authentication Protocols: Static, Robust and Continuous

Together with the type of authentication factor used, taxonomy of authentication includes *static*, *robust*, and *continuous* authentication, depending on the sustaining protocol [40]. An *authentication protocol* is [39]: a defined sequence of messages between an entity and a verifier that enables the verifier to corroborate the entity's identity.

Authentication is said to be *static* when a specific factor (as a password) is not modified from one login session to another, and consequently can be used more than once; this protects the system from attacks in which the intruder does not know, or does not possess the information needed to complete the verification process. Traditional mechanisms based on passwords, exploits protocols of this family, which security directly depends on how hard is to guess the password or how effectively it is protected.

An additional category consists in the so-called *robust* authentication protocols, based on dynamic factors, which vary from one session to another, and, therefore, are valid just once. A classical example is a dynamic key as OTP (*One-Time Password*), which prevents an attacker from accessing the system, even obtaining the credentials: they will not be usable anymore after the first login has been performed. Users cannot memorize and reuse OTPs – they are typically generated by specific devices, as tokens, or trusted mobile applications; consequently, robust authentication protocols are mainly *object-based*. However, once the user is authenticated, either statically or dynamically, he/she obtains full access to the system for the whole duration of a session, which terminates only with an explicit logout or after an idle activity period. A problem that can arise, and that continuous authentication intends to address, is thus the situation in which a user leaves the device unattended, even for a short break, without logging out: an intruder may intervene and take over confidential resources, undermining the system, its integrity and availability. In some cases, the user identity has to be verified so frequently as the access to confidential resources is granted [41], [19].

In a *continuous* authentication protocol, the system constantly and repeatedly requests the authentication factor(s) necessary to confirm the identity of the entity which is being authenticated.

Authentication factors, which for their own nature are most suitable for continuous authentication are biometric traits, because by mean of specific sensors, they can be acquired in an implicit way. For example, let us consider a system which every thirty seconds asks the user to enter a password or a code written on an badge: this procedure hinders the user from carrying out the activities and forces him/her to actively and repeatedly participate in the authentication procedure.

2.5 NON-REPUDIATION

In this Section, we present some basic concepts dealing with non-repudiation, its goal and the traditional mechanism used to provide it.

Together with authentication, also the ability to demonstrate that users or entities requested specific services, or performed certain actions, is useful. In fact, when a dispute arises or an error occurs people may attempt to deny their involvement and to repudiate their behavior. For instance, customers may disclaim a withdrawal from their account, or a payment with their own credit card that they actually did. According to The New York Times, 0.05 percent of MasterCard transactions worldwide are subjects of disputes, which probably means around 15 million questionable charges per year [51]. In addition to users denying their usage of a service, there have been cases of malicious operators or disreputable service providers. One example is a fraudulent or inaccurate web service for stock trading: a broker is instructed by a customer to buy or sell stocks, or to follow a particular investment strategy, but retards the process in order to help other investors have their trades executed quickly [52], causing a consistent loss of money for the client. Then, the trader denies the involvement.

2.5.1 *Definitions and Typical Approach*

The two above are both examples of *repudiation*, which can be defined as the denial of having participated in all or part of an action by one of the entities involved [53]. Consequently, *non-repudiation* is the ability to protect against such denial. A non-repudiation mechanism has to provide evidences in order to clarify responsibilities and guarantee the establishment of the facts even in front of a court of law. Therefore, a non-repudiation service can be useful both as a mean to obtain accountability as well as a deterrent for deliberate misbehaviours. In the two situations reported above, a such mechanism would prevent that the fraudulent customer succeeds in denying a transaction, or would help an innocent client to protect his investments.

Non-repudiation provides the capability to determine whether a given individual or entity took a particular action [54]. Referring to secondary attributes of dependability and security, it can be seen as the ability to provide *non-repudiability* property as defined in [1] and already introduced in Section 2.1.1. This property has been presented in the context of message exchange, but it – and consequently non-repudiation services– can be useful also in different contexts. Typically, non-repudiation protects individuals against:

i) authors, repudiating having authored particular documents;

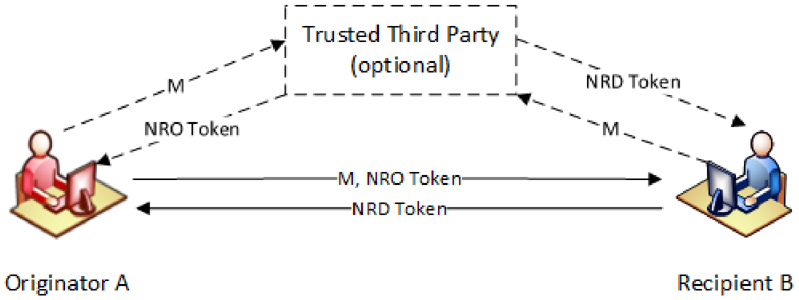


Figure 6: Traditional non-repudiation approaches for the transaction scenario. Image adapted from ([56]).

ii) messages senders, denying having transmitted messages;

iii) messages receivers, denying messages reception;

iv) signatories, repudiating their signature on documents.

Thus, non-repudiation services can be used to determine if information originated from a particular individual, or if an individual took specific actions (e.g., send an email, sign a contract, approve a procurement request), or received specific information [54].

The goal of the non-repudiation service is to generate, collect, maintain, make available and validate *evidence* concerning a claimed event or action in order to resolve disputes about the occurrence or no occurrence of the event or action. Such evidence is an essential object, and may be produced either directly by an end entity, or involving a Trusted Third Party (TTP) [56]. In order to obtain non-repudiation services, organizations employ various techniques or mechanisms; the most common is digital signature [57].

Regarding transactions, assuming that an entity A wishes to send a message M to entity B, typical disputes that may arise are 6:

- *Repudiation of Origin*. B claims that it received M from A, while A denies sending it;

- *Repudiation of Delivery*. A claims that it has sent M to B, while B denies having received it.

Then, we also mention two additional disputes that may exist in case a delivery authority is involved: Repudiation of Submission and Repudiation of Transmission [56]. According to the existing ISO/IEC standard, two different mechanisms for non-repudiation are distinguished [56]. A NRO (Non-Repudiation of Origin) token is used to provide protection against A's false denial of having originated the message. It is generated by A (or by the TTP), sent by A to B, and stored by B after verification of its validity. An NRD (Non-Repudiation of Delivery) token, instead, is used to provide protection against the B's false denial of having received and recognized the content of the message. It is generated by B (or by the TTP), sent to A, and stored by A after verification of its validity. If TTP is involved (optional), it must keep all NRO tokens generated and record whether or not each NRO token is used to generate a NRD token. For electronic transactions, the evidences (non-repudiation tokens) are created using digital signatures. A knows its own public key certificate and the associated private key, B knows its own public key certificate and associated private key, and the corresponding public key certificates are available to all the entities concerned [56].

3 INTRODUCTION TO BIOMETRICS

This Chapter provides basic concepts about biometrics, starting from its definition, in Section 3.1. Then, Section 3.2 briefly introduces the most common physical and behavioural biometric modalities and their attributes. To conclude the Chapter, Section 3.3 presents biometric systems: how they are employed for user authentication, the accuracy and error metrics, as well as the typical sources of error that have to be faced.

3.1 ORIGINS AND DEFINITION

The term *biometrics* derives from the ancient Greek words (*métron*), and (*bíos*), and literally means measure of life. Although this discipline became known in the 19th century for its usage in the judicial and forensic fields [10, 12, 16], today it is widely used for people recognition in a relatively large number of applications, also because dedicated sensors have been recently integrated into many commercial devices as notebooks and smartphones.

Biometrics is defined by the ISO[1] as: "automatic recognition of individuals based on their biological and behavioral charachteristics" [17].

3.2 BIOMETRIC TRAITS AND THEIR ATTRIBUTES

In this Section, we first present attributes of biometric traits and systems. Then, there is a brief description of some the traits which are currently the most used for recognition of individuals. A detailed analysis of all the possible biometric modalities is out of the scope of this Thesis: the interested reader is referred to [11].

It is impossible to determine whether exists and which is the very best biometric trait for the aim of recognition. However, the following is a list of requirements that each trait should possess and which may help a designer of a recognition system in choosing the most appropriate traits for its purpose [10].

- *Universality*: ideally, everyone should have the trait;
- *Distinctiveness* or *Uniqueness*: the trait is sufficiently different from one person to another;

[1] International Organization for Standardization

- *Permanence*: the trait is sufficiently invariant, with respect to the matching criterion, over a period of time, otherwise becomes unusable for recognition algorithms;
- *Collectability* or *Measurability*: the trait can be acquired and measured quantitatively, without the need of sophisticated sensors;

There are also requirements and issues, often depending on the traits employed, that practical biometric recognition systems should address, including [10]:

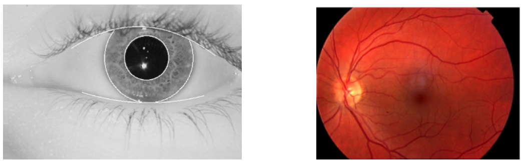
- *Performance*: which refers to the recognition accuracy, and occasionally also to speed, and the resources required to achieve the desired recognition accuracy;
- *Acceptability*: indicates the extent to which people are willing to accept to be subject to recognition of a specific biometric trait, or to use it in their lives;
- *Circumvention*: which refers to the easiness in fooling the system by fraudulent methods like physical biometric traits reconstruction - e.g. fake fingers - or by imitation of user behaviour.

The list of biometric traits usable for recognition is very long; some of them have been studied since the nineteenth century, others in the last thirty years, and others have arisen more recently. In the following, we present a selection of some of the most relevant traits, exposing their main characteristics, highlighting their main strengths and weaknesses [10, 12, 13, 15].

Face is a biometric characteristic that we all use in our everyday life for people identification and is probably the most used in biometric recognition systems. The trait is characterized by a set of macro elements (as mouth, nose, eyes, lips, etc.), information regarding their dimensions, and relations between them. Face recognition is a non-intrusive method, which can be static, e.g., using videos or pictures, or dynamic, by means of a real-time scanning.

Fingerprint has been employed in recognition since the past century thanks to this trait's high accuracy. A fingerprint, as for the palm print, is a set of ridges and valleys present in the surface of the fingers, whose information is determined during the first seven months of fetal development. In addition, fingerprints of identical twins are different as well as the prints on each finger of the same person.

A modern fingerprint recognition system, if based on a relatively economical sensor, is capable of recognizing accurately few hundred users. In order to identify people on a larger scale (as millions of users), a smart strategy employs more than a single fingerprint for each individual.



(a) (b)

Figure 7: (a) An eye with the iris highlighted. [42] (b) Picture of a retina. [43]

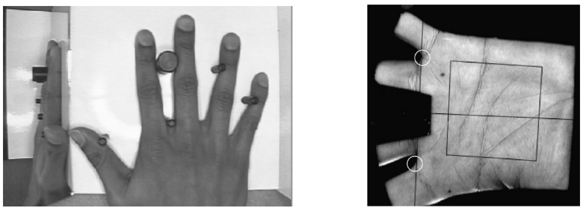
Keystroke, sometimes also known as keystroke dynamics, has been shown to be a behavioral biometric trait employable for user recogntion in the last decades (e.g., by [44] in 1990). In fact, the rhythm with which a user presses and releases the keys, even if it has not a high uniqueness, permits to create a *template* for user identity verification. This trait can be monitored in a non-intrusive way, while the person is using the keyboard, and it favours the continuous verification. However, keystroke recognition systems are not considered the most accurate.

Voice is a biometric trait which combines physiological and behavioral traits and is based on the shape and size of the appendages (e.g., vocal tracts, mouth, nasal cavities, and lips) that are used in the synthesis of the sound. These physiological characteristics of human speech are invariant for an individual, but the behavioral part of the speech of a person changes over time due to age, medical conditions (such as a common cold), emotional state, etc. There are *voice/speech recognition* systems capable of recognizing the words, and others whose goal is the *speaker identification*.

Iris is the 11mm annular coloured region of the eye, bounded by the pupil and the sclera (the white part of the eye). The visual texture of the iris is formed during fetal development and stabilizes during the first two years of life. The complex iris texture carries very distinctive information useful for personal recognition. Each iris is distinctive and, like fingerprints, even the irises of identical twins are different.

Retina is a vasculature with a very rich structure, possesses a high distinctiveness and is hard to change or be replicated. Experiments showed that also identical twins present different vascular patterns.

Measurements regarding the human *hand geometry*, which normally are employed for recognition, are: shape, size of palm, lengths and widths of the fingers. Although those measurements do not significantly vary from an individual to another, commercial hand geometry-based verification systems have been installed in hundreds of locations around the world. As for the other traits, it has its own advantages and weaknesses. It is a simple technique, not intrusive, easy and cheap to use. In addition, its accuracy in verification is not influenced by environmental factors, as the weather, or factors like dryness of the skin. Nevertheless, one of the main drawbacks of this modality is that it is hard to be embedded in other devices, because of the physical size.



(a) (b)

Figure 8: (a) Analysis of hand geometry. [45] (b) Palmprint. [46]

Palmprint of the human hands, similar to fingerprint, contains pattern of ridges and valleys, some of which are determined during the first weeks of pregnancy, and others appear at the birth. Some of the characteristics are genetic, some are not, and are different even between identical twins. The palmprint area is larger than the area of a finger, therefore, this means that palmprints have even higher distinctiveness than fingerprints. However, on the other side, palmprint scanners need to capture a larger area, so they are not easy do be embedded and usually more expensive than other sensors.

DeoxyriboNucleic Acid (DNA) is well known for being used in the context of forensic applications for person recognition. It can be considered one of the most accurate traits, taking into account that the probability of two individuals having the same DNA sequence is lower than 1 on 100 millions (except for the fact that identical twins have identical DNA patterns). However, it is not often used in applications different from forensics, considering that:

- it is relatively easy to obtain the DNA of a subject without his/her permission, and this can facilitate the masquerading.
- it is difficult to imagine an application in the field of real-time and automatic verification of individuals, because it usually needs quite complex chemical methods, as well as advanced tools and knowledge in order to be analyzed and compared;
- for ethical and privacy reasons, because it can bring information as susceptibilities of a person to certain diseases, and it could be somehow abused or result in discrimination.

Vital Parameters have been studied in literature as biometric traits which can be applied for human recognition. An example is monitoring heart rate from the ECG. Recognizing and classifying the heartbeat are extremely useful for medical purposes, but can be a viable solution also for recognition, because the geometric and physiological differences of human heart of different subjects also impact, with a certain uniqueness, the heart signals.

Gait is a quite complex behavioural trait and consists in the way an individual walks, runs, goes up and down the stairs, or moves in descent. Gait acquisition

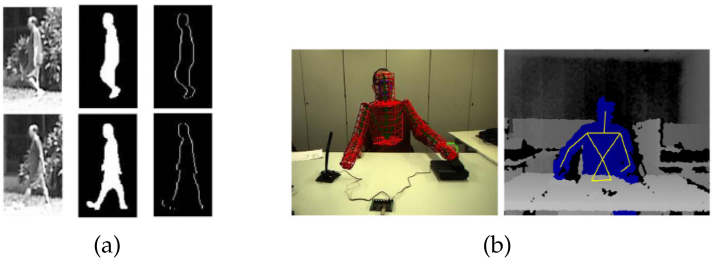


Figure 9: (a) Gait analysis. [47] (b) Skeleton for movement template generation. [48]

is usually done with the video-sequence footage of a walking person or with accelerometers placed on the body to register the activity of the subject. Generalizing, gait analysis can be considered one of recognition methods of human activities belonging to the so-called *activity-related biometrics*, which is based on complex body movements as moving, picking up an object, which are performed by everybody with their own style and personal behavior. It is possible to track, with the help of specific sensors, trajectories of body movements (e.g., head and hands), and generate a template of a subject performing a specific action, (as shown in Figure 9 (b)), for the purpose of authentication.

The handwritten *signature* is a well known, and highly adopted, behavioural trait which consists in the way a person signs his or her name. The weaknesses of this trait are multiple: it requires a contact and a conscious effort by the user, it changes over a period of time and is influenced by physical and emotional conditions of the individual. In addition, signatures of some people vary substantially. Further, professional forgers may be able to reproduce signatures that fool the system. However, it has been accepted in government, legal, and commercial transactions as a method suitable for verification.

The way a user interacts with the *touchscreen* of a smartphone can be considered a behavioral biometric modality. In fact, acquiring the basic movements – such as scrolling in different directions – has shown useful for the system to recognize the users. The accuracy of recognition does not seem very high to let this trait employable for authentication; however, it could be integrated with other characteristics in a multi-biometric system.

3.3 BIOMETRIC SYSTEMS

A biometric system is nothing more than a *pattern recognition* system that compares the biometric trait of a user with a previously acquired and stored version of the same trait.

The first step consists in a registration, also known as *enrollment*, of the user; he/she presents a biometric trait to the sensor capable of extracting the main features and storing their digital representation in a database. The scanning of

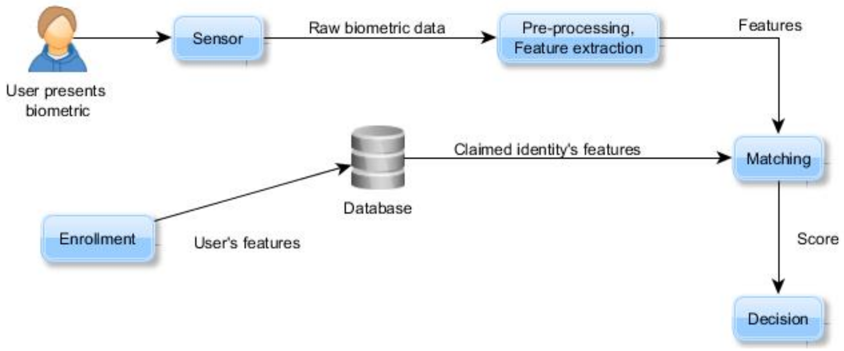


Figure 10: A typical biometric verification process.

raw data with sensors may be repeated until the quality of information is not adequate. Subsequent steps can be slightly different depending on the application context: they may belong to the *verification* or *identification* processes.

The verification process, shown in Figure 10, is often adopted for the user authentication scenario and consists in a *one-to-one matching*.

- First, the individual who desires being authenticated, claims the identity (e.g., by mean of a user name or another identifier);
- a sensor acquires in real-time the raw biometric data of the user;
- the system, during a pre-processing step, extracts the main features from the trait;
- then, it matches the acquired sample with the corresponding template, extracted from a database;
- matching determines a score, based on which the system assesses the user authenticity and decides whether to accept or reject it.

Typically, verification is used by *positive recognition* systems that aim to prevent multiple users to access them with a unique identity, and, consequently, its ideal application is access control and intrusion prevention.

Identification, usually, is adopted by *negative recognition* systems, with the aim of preventing a unique user to have multiple identities. It consists in a *one-to-many* matching: the system possesses a biometric information and, looking for a corresponding template in the database, wishes to establish the identity of the owner of that trait. Consequently, unlike verification, identification is a process that may even fail, and it actually happens if the user to be identified has not been previously enrolled. This procedure is similar to the way the human brain performs identification when, for instance, we meet a person: it matches the information obtained by the eyes with the image already present in our memory. Occasionally, identification is adopted also for positive recognition, for instance when the user is not required to claim his/her identity; however, this could be very expensive from a computational point of view, especially in databases of large size.

3.3.1 *Accuracy and Errors*

Biometric traits are quite complex, and often their digital representation can vary considerably. Therefore, it is not reasonable to require an exact matching between two samples. For example, two acquisitions of face images belonging to the same person are never completely identical. For this reason, matching algorithms, which are integrated in biometric systems, generate a matching score, typically between 0 and 1, which quantifies the similarity between the trait received as an input and the sample stored in the database. A score is close to 1 when the similarity to the template is higher; the algorithm uses a threshold, based on which the system can either decide if the representation belongs to the authentic user or not. But this inference is error-prone. If a received biometric trait is erroneously considered as legitimate, we have a *false accept*, meaning that an impostor is accepted as genuine user. Conversely, if the matching of two samples of biometric traits actually belonging to the genuine user produces a score lower than the threshold, the user is not authenticated, and a *false reject* happens.

Consequently, the error can belong to the aforementioned two categories, and the corresponding error metrics are [6, 10]:

- *False Acceptance Rate (FAR)*: the proportion of verification attempts with wrongful claims of identity that are incorrectly confirmed;
- *False Rejection Rate (FRR)*: the proportion of verification attempts with truthful claims of identity that are incorrectly denied.

Often, they are also referred as False Matching Rate (FMR), and False Non Matching Rate (FNMR) respectively. The matching scores, s, obtained by a genuine user who tries many times to authenticate, have a probability density function (pdf), as shown in Figure 11; analogously, we show the distribution of the scores, indicated as *Imposter profile*, obtained by non-legitimate individual claiming to be the actual user.

As it can be seen in the Figure 11, the two error rates vary based on the threshold t, as in a trade-off between security and usability. Indeed, by increasing the threshold, the number of false accepts decreases, and this is reflected in an enhancement of security level for the system protected by such a security mechanism. However, the adverse effect would be a worsening in usability.

Conversely, by decreasing the threshold we typically obtain a more usable system, but at the expense of security.

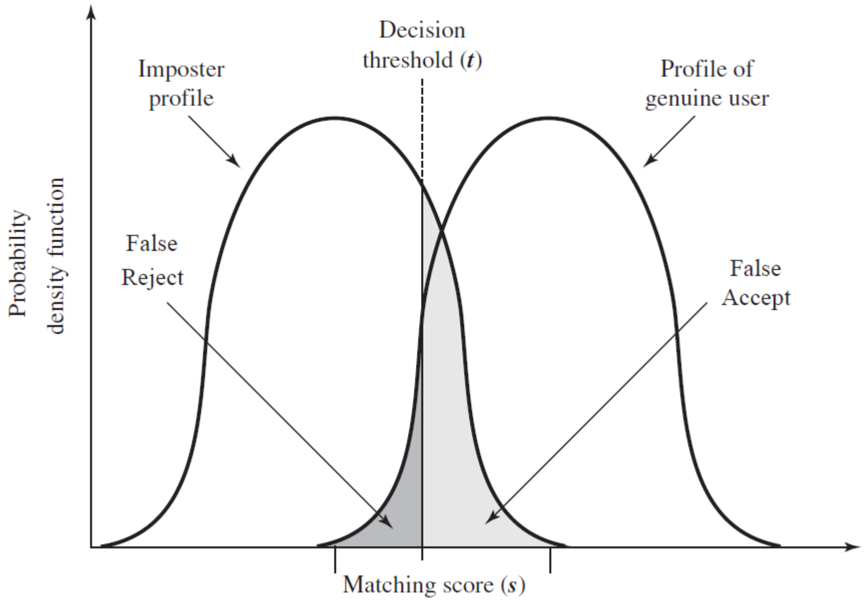


Figure 11: Ad-hoc example of error rates in a biometric system; FAR and FRR, and their relation with the threshold. Image adapted from [7].

The adequate accuracy and the choice to be taken, normally depend on the application scenario and on the requirements.
It has to be noticed, however, that the score distribution of Figure 11, is just an example built with the purpose of explaining error rates, while a more realistic score distribution can be see in Figure 12 [152].

Together with errors related to the matching process, a biometric system can be subject to the following [14]:

- *Failure to Capture* (FTC): indicates an unsuccessful tentative of acquiring a biometric trait. It typically happens when a sensor is not able to detect a signal of sufficient quality associated with a biometric characteristic.
- *Failure to Enroll* (FTE): indicates an unsuccessful tentative of registering a user. It may happen when, during the enrollment a system rejects a template because of low quality.

The enrollment is thus very important: requiring that a database contains only high quality templates, increases the matching accuracy and, consequently, reduces both FAR and FRR. Failure and error rates in biometric systems are therefore dependent upon one another: each of these metrics constitutes an important specification for the accuracy of a biometric system.

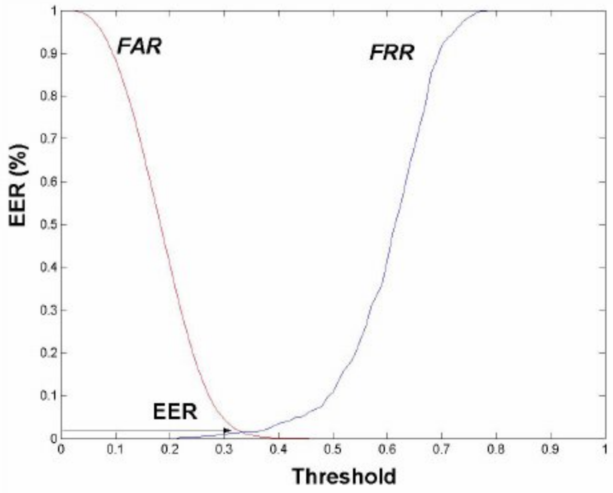


Figure 12: Realistic example of error rates in a biometric system [152].

3.3.2 *Sources of Errors*

Biometric signals of an individual and their representations may vary considerably based on multiple factors, including: the acquisition method, the surrounding environment, the human-machine (sensor) interactions, as well as physiological phenomena or medical conditions. The most common reasons are the following [10]: the presentation of the trait is inconsistent, is not-replicable, or there is an imperfect acquisition of the signal.

Inconsistent presentation. The signal captured by a sensor depends either on the biometric trait itself, or on the way the trait is presented. Consecutive acquisitions are never completely identical. Indeed, it may happen that a biometric signal is acquired as a non-deterministic composition of physical biometric traits of the user behaviour, and of the interaction between the human and the acquisition interface. As an example, let us consider the fingerprints: we should notice that the finger is not a hard object and possesses a three-dimensional shape, but its surface is typically projected on a two-dimensional surface, that is, the surface of the sensor. This process of projection is never controlled precisely. For this reason, different acquisitions of the same fingerprint are always slightly different.

Regarding the face images, instead, is possible that the variation consists in a different pose of the user in front of the camera, as well as the facial expression, as in the example of Figure 13[2].

Non-replicable presentation. Measurements of biometric traits may be prone to laceration and consumption, malfunction, or physical development. For example,

[2] where the subject is the author of this Thesis.

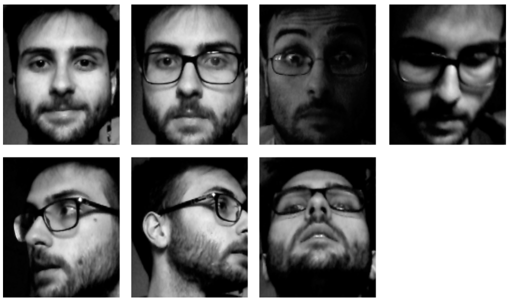


Figure 13: Example of inconsistent presentation: changing the pose w.r.t the camera or the facial expressions. Non-replicable presentation: caused by wearing or not wearing accessories as glasses.

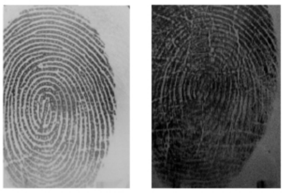


Figure 14: Non-replicable presentation: temporary changes in fingerprint caused by laceration and consumption in the ridges. [14]

as shown in Figure 14, the structure of the ridges constituting fingerprints may vary because of accidents or manual labour, sometimes permanently.

Face recognition may suffer from many factors, including: beard growth or cutting, accidents (as nasal septum fractures), wearing accessories (e.g., glasses, hats, jewelry), make-up, haircuts. All these reasons may make the presentation non-reproducible.

Imperfect acquisition of the signal. Conditions regarding signals, in practice, are rarely perfect and cause additional variations in the acquisition of a biometric trait. Typically, this problem is due to noise, and noisy data can give rise to errors in the matching process (false accept or false reject). As an example, face images are affected by differences in the lighting, which cause significant variations in the face appearance. In particular, back-light significantly complicates face recognition.

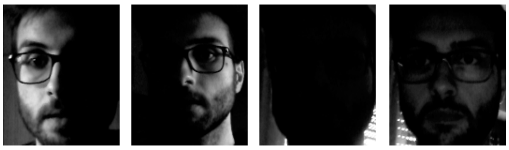


Figure 15: Imperfect acquisition of face images, caused by a strong back-light. In the first face on the right, it can be noticed how a strong frontal light - the light emitted by a screen- partially solves the problem.

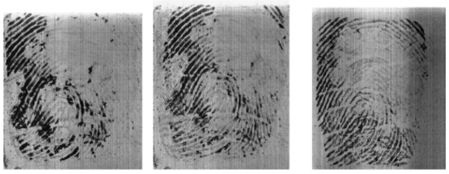


Figure 16: Imperfect acquisition of fingerprints: the figures show low quality cresta, probably because of skin dryness ([14]).

Regarding fingerprints, sometimes the noise is due to accumulation of dirt on a fingerprint sensor, which provokes an incomplete contact of the crests and a consequent low quality acquisition. A similar situation can be influenced by a skin dryness or injuries, old age, or genetic factors which cause a crests consumption, as well as sweat, or humidity in the air.

Because of the several aforementioned causes of variations in the biometric signal, determining if two representations belong to a unique individual, requires complex pattern recognition mechanisms. In addition, it is expected that signal variations constitute actual input which, if not properly handled, may raise errors in the biometric system (e.g., false accept and false reject). Errors not properly corrected, may lead the system to a failure.

Together with sources of errors, there are also other limitations for biometric systems, directly related to the attributes of the traits – presented in Section 3.2 – as low distinctiveness, and non-universality.

3.3.3 *Non-zero Effort Attacks*

Another way to classify sources of errors and attacks is between *zero effort*, and *non-zero effort*. For the errors discussed so far, the cause is ascribable to an unintentional event, or to environmental condition. However, we have to acknowledge that in biometric recognition systems there may be also issues related to the latter category, where the attacker is a person who actively masquerades as someone else by falsifying the biometric of the claimed identity, e.g., using artificial materials [153]. Among these attacks, *spoofing* is one of the most known and relevant: it has been shown in literature that it can be performed using commonly available materials and furthermore, it does not require specific knowledge of the internal functionality of the system. For instance, a person can fool a fingerprint system by using artificial or gummyfingers of another person in order to gain unauthorized access [154].

Non-zero effort attacks and related countermeasures will be further addressed in the context of Section 5.1.

Many of these errors, issues and attacks are difficult to overcome, especially by *unimodal* biometric systems, which are based on a single trait. However, some

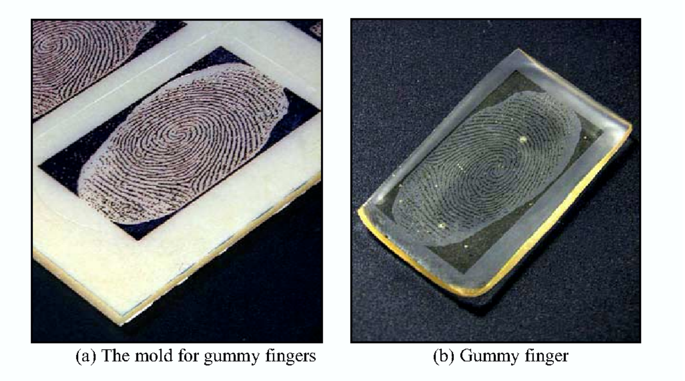


Figure 17: A non-zero effort attack may be performed e.g., with a gummy finger (b), obtained with a specific mold (a) and produced from a residual fingerprint ([154]).

of the limitations can be reduced thanks to the usage of multiple biometric characteristics properly combined in a single system.

4 CONTINUOUS AUTHENTICATION

This Chapter focuses on continuous authentication. Section 4.1.1 introduces multi-biometric systems, and the reasons why they are considered a viable solution to provide the desired security service. Section 4.1.2, presents a literature review and comparison of the most interesting continuous authentication solutions, with a detailed description of CASHMA [19] in 4.1.2. The review of the state of the art is enriched with an overview and a gap analysis about commercial solutions for continuous authentication.

4.1 MULTI-BIOMETRIC CONTINUOUS AUTHENTICATION SYSTEMS

We now introduce multi-biometric systems especially focusing on the characteristics that make them applicable and effective for the continuous authentication scenarios.

4.1.1 *Multi-Biometric Systems*

A biometric system can be defined as *multimodal* if it uses more than a single biometric trait -either physical or behavioral- in order to perform enrollment, verification or identification. For example an authentication system may request the user to present fingerprints of multiple fingers, or to provide both the face image and the iris scan.
It has been shown in the literature (e.g., [58]) that this kind of systems possess various advantages:

- higher accuracy in the recognition;
- more reliability, thanks to redundancy provided by the multiple traits;
- they permit to overcome limitations of unimodal systems (e.g., non-universality: with multi-modality, the population covered is relatively wide);
- they are robust against *spoofing* attacks given that it becomes harder for an attacker to provide simultaneously multiple traits, so that they can actually resemble the traits of the legitimate user (especially if the combination of biometric traits is randomly assigned).

In terms of security, the improvement of employing multimodality, instead of unimodal biometric authentication, is evident. However, if the system has not been properly designed, other aspects as computational cost, time needed to complete the verification process, or the resources usage, may significantly increase.

Integrations Schemes and Fusion levels

According to the application context and to the requirements, it is fundamental to choose the traits, sensors and algorithms in order to make them perfectly integrated. This integration can happen according to different schemes [10, 59]:

- *Multiple sensors*: using more sensors to acquire the same type of biometric trait; this introduces redundancy and consequently increases system reliability and fault tolerance. The system can still provide an accurate decision about biometric recognition even if one of the sensors is not able to complete the verification.

- *Multiple traits*: employing multiple sensors to acquire different types of biometric traits; this not only introduces redundancy, but also permits to tackle the potentially low universality of some traits.

- *Multiple acquisitions*: the sensor acquires exactly the same trait for multiple times (or, it acquires multiple samples of the same trait, as fingerprints of different fingers). This can increase the accuracy in the recognition without having a significant impact on computational costs;

- *Multiple algorithms*: to process the same trait through different formats, features vectors, or matching strategies.

Once decided how each unimodal subsystem has to operate, they have to be integrated to create a unique multimodal system. Biometric recognition systems are composed of different modules, so the integration process becomes a fusion of the outputs of those modules. The fusion can happen at various levels:

1. *Sensor*: raw data acquired by different sensors can be processed and integrated in order to generate new data from which subsequently extract the main features.

2. *Feature*: features sets generated from different traits can be fused to create a new set of features representing the same individual. For example, characteristics regarding hand geometry can be added to coefficients extracted from face image in order to obtain a new set of higher dimension.

3. *Score*: matching scores obtained from single modules can be fused to generate a final score. For instance, scores obtained through fingerprint recognition and face image can be combined (e.g., by a simple sum) to obtain a new score which supports the final decision.

4. *Ranking*: this fusion is especially useful for biometric systems working in identification mode. Each classifier associates a matching order to the identities stored in the database (higher positions correspond to higher matching scores). The fusion of rankings creates a new matching order to be used for the final decision.

5. *Decision*: when the matching process terminates, each unimodal subsystem generates a label corresponding to the decision (e.g., accepted or rejected if the biometric system operates in verification mode, or the identity of an individual in case of identification mode). The final decision is derived in many ways, for example by majority voting.

4.1.2 *The State of the Art*

Several studies describe frameworks, systems and novel characteristics for biometric authentication of humans. The following literature review includes the most relevant and recent papers available so far on this topic and is especially concentrated on approaches based on continuous authentication which conducted usability tests involving real users.

Academic Literature Review

We describe our findings with the support of Table 1.
The surveyed works are ordered by the year of publication starting from the most recent. Then, the table reports on various aspects:

- *Multi-bio*: the solution integrates multiple traits, thus is multi-biometric;
- *Cont. Authe.*: the authentication is continuous (instead of static or robust);
- *Usability Testing*: the usability of the solution has been measured;
- *Security & Perf. Evaluation*: the solution has been tested from a security and performance point of view;
- *Tests with user base*: the assessment involved real users and is not performed only through simulations.
- *Trust score*: the solution is based on a trust level or similarity score;
- *Use case*: explains if the solution has been designed for a specific system (and which is it), or for a specific use case.

Sitova et al. [60] introduced a set of behavioral biometric features for continuous authentication of smartphone users: hand movement, orientation, and grasp (HMOG). They evaluated them from three perspectives: continuous authentication, biometric key generation performance, and energy consumption.

Table 1: Comparison of related works on multi-biometric continuous authentication

	Year	Multi-	Cont.	Usability	Security &	Tests with	Trust	Use case &
		bio.	Auth.	Testing	Perf. Eval.	user base	score	Target system
[60]	2016	✓	✓		✓	✓		Mobile
[19]	2015	✓	✓		✓		✓	Internet Services
[61]	2015	✓	✓		✓			Mobile
[62]	2015	✓	✓		✓	✓		Mobile
[63]	2014	✓	✓	✓	✓	✓		Mobile
[64]	2014	✓	✓		✓	✓		Desktop/Generic
[65]	2014	✓	✓		✓	✓		Desktop/Generic
[66]	2014		✓		✓	✓		Desktop/Generic
[67]	2014	✓	✓		✓			Desktop/Generic
[68]	2013	✓	✓		✓	✓	✓	Mobile
[50]	2013		✓		✓	✓		Mobile
[69]	2013	✓	✓	✓		✓	✓	Mobile
[70]	2013	✓	✓	✓	✓			Mobile
[71]	2013		✓		✓		✓	Desktop/Generic
[72]	2013	✓	✓		✓	✓	✓	Desktop/Generic
[73]	2012		✓		✓	✓		Mobile
[74]	2011	✓	✓		✓	✓		Mobile
[75]	2011	✓	✓		✓			MANET
[76]	2010	✓			✓		✓	Desktop/Generic
[77]	2010	✓	✓		✓	✓		Desktop/Generic
[78]	2010	✓			✓			Desktop/Generic
[30]	2009	✓	✓	✓	✓	✓		Desktop/Generic
[59]	2008	✓	✓		✓		✓	Desktop/Generic
[31]	2007	✓	✓	✓	✓	✓		Desktop/Generic
[79]	2006			✓	✓	✓		Distributed platf.
[80]	2003	✓	✓		✓			Desktop/Generic

The evaluation was performed on multi-session data collected from 100 subjects under two motion conditions (i.e., sitting and walking). The results show that HMOG is well suited for continuous authentication of smartphone users.

In [19], a sequential multi-modal biometric authentication system is composed of an authentication service, web services and clients. Clients are users' devices (e.g., laptop and desktop PCs, smartphones, tablets) that acquire the biometric data, and transmit those data to an authentication server for a single-sign on procedure. This solution inspired the work of this Thesis and will be described in more detail in a dedicated Section 4.1.2.

Saevanee et al. [61] propose a novel text-based multimodal biometric approach built on linguistic analysis, keystroke dynamics and behavioral profiling. They present a framework that is able to provide robust, continuous and transparent authentication. Due to the lack of public datasets, the effectiveness of the proposed framework for providing security and user convenience was evaluated via a simulation approach (using the MATLAB environment). The simulation process involved implementing a virtual user. The result showed that it is able to provide a 91% reduction in the number of intrusive authentication requests required for high security applications.

Crouse at al [62] present a work on a face-based continuous authentication system that operates in an unobtrusive manner. The authors propose a methodology for fusing mobile device (unconstrained) face capture with gyroscope, accelerometer, and magnetometer data to correct camera orientation and, by extension, the orientation of the face image. Experiments demonstrate (i) improvement on face recognition accuracy from face orientation correction, and (ii) efficacy of the prototype continuous authentication system.

De Marsico et al. [63] propose a biometric application based on a multimodal recognition of face and iris, which is designed to be embedded in mobile devices. The system is inspiring for our purpose even if specific for a different target system. A framework complementing face recognition and clothing colors for continuous authentication is proposed in [77] et al. Similarly, the work by Tsai et al. [64] builds a passive continuous authentication system based on both hard and soft biometric features (e.g., clothes color). These approaches could be integrated to other multi-biometric continuous authentication solutions to further improving the detection capability; however, at present stage a proper usability study is missing and the improvements in security, especially tolerance to counterfeit, are not detailed.

Bailey et al. [65] present a behavioral biometric system that fuses user data from keyboard, mouse, and Graphical User Interface (GUI) interactions. As a multimodal system, authentication decision is based on a broader view of the user's computer activity while requiring less user interaction to train the system than previous work. The system performs authentication every two minutes. Testing over 31 users shows that fusion techniques significantly improve behavioral biometric authentication accuracy over single modalities on their own.

Roth et al. [66] propose a novel biometric modality named *Typing Behavior* (TB) for continuous user authentication. Given a webcam pointing toward a keyboard, they develop real-time computer vision algorithms to automatically extract hand movement patterns from the video stream. Unlike the typical continuous biometrics, such as keystroke dynamics (KD), TB provides a reliable authentication with a short delay, while avoiding explicit key-logging. They collected in a database videos of 63 unique subjects, with type static text and free text for multiple sessions. The experimental results show a superior performance of TB when compared with KD.

Prakash et al. [67] introduce a new continuous user authentication scheme, which is designed to authenticate the user irrespective of their posture in front of the system. The system continuously monitors the user by using soft biometrics (color of user's clothing and facial skin) along with hard biometrics. It automatically registers soft biometric traits every time the user logs in and fuses soft biometric matching with the conventional face biometric authentication.

In their paper, Draffin et al. [68] propose a novel passive authentication method for mobile devices users. The authors show that every user possesses unique physical and behavioral characteristics reflected by how rapidly they type with the device soft keyboard, and influenced by a variety of soft-keyboard specific micro-behavior features. Using this data, plus a variety of statistical tools, they generate a certainty score of whether the user's phone is in a stranger's hands. Without any contextual information, they can passively identify that a mobile device is being used by a non-authorized user.

In their work, Frank et al. [50] investigate whether touchscreen gestures are a viable biometric trait for continuous authentication of smartphone users. Experiments to assess security and performance involving users are presented. Based on the results, the authors identify this method as suitable for multimodal biometric authentication system. However, the solution is specific for smartphones, and the authors acknowledge that it cannot securely serve as an exclusive authentication mechanism of a device.

Zhu et al. [69] investigate the usage of passive sensory data collected from accelerometers, gyroscopes and magnetometers to ensure the security of applications and data on mobile devices. They build the gesture model of how a user uses the device and propose a framework which calculates the sureness that the mobile device is being used by its owner. Based on the sureness score, mobile devices can dynamically request the user to provide active authentication, or disable certain features of the mobile devices to protect user's privacy and information security.

Crawford et al. [70] address mobile device authentication as provided by a password or sketch. They propose an extensible Transparent Authentication Framework that integrates multiple behavioral biometrics with conventional authentication to implement a continuous authentication mechanism. The security and usability evaluation of the proposed framework showed that a legitimate device owner can perform tasks while being asked to authenticate explicitly 67% less often than without a transparent authentication method. Furthermore, the evaluation showed that attackers are soon denied access to on-device tasks as their behavioral biometrics is collected.

Both the works by Mondal et al. [71] and by Deutschmann et al. [72] are based on a trust model and influenced by Bours et al. [22]. Authors of [71] propose to perform continuous authentication using Mouse Dynamics as the behavioral biometric modality. They used a publicly available mouse dynamics with data of 49 users and evaluated the system performance with 6 machine learning algorithms. Their continuous authentication scheme is based on a trust model which uses both global thresholds and person specific thresholds. Deutschmann et al. [72] investigate the possibility of authenticating users continuously on desktop computers. They tested a continuous *behaviometric* authentication system on 99 users over 10 weeks, focusing on keystroke dynamics, mouse movements, application usage, and the system footprint. They continuously monitored users' activity during an entire working session. Such a continuous-authentication system uses the set of behavioral traits to calculate a similarity ratio (score) between the user's current and expected behavior.

Meng at al. [73] propose a novel user authentication scheme based on touch dynamics that uses a set of behavioral features related to touch dynamics for accurate user authentication. They select 21 features, collect and analyze touch gesture data of 20 Android phone users, comparing several known machine learning classifiers.

Shi et al. [74] describe *SenGuard*, a user identification framework that enables continuous and implicit user identification service for smartphone. It leverages availability of multiple sensors on smartphones and passively uses sensor inputs as sources for user authentication. SenGuard invokes active user authentication when there is mounting evidence that the phone user has changed. A prototype of SenGuard was created using voice, location, multitouch, and locomotion. Preliminary empirical studies with a set of users indicate that those four modalities are suited as data sources for implicit mobile user identification.

In their paper, Bu et al. [75] focus on user-to-device authentication in high security *Mobile Ad hoc NETworks (MANETs)*. The paper studies distributed combined authentication and intrusion detection with data fusion in such MANETs. Multimodal biometrics is deployed to work with intrusion detection systems (IDSs). The system decides whether user authentication (or IDS input) is required and which biosensors (or IDSs) should be chosen, depending on the security posture.

Xu et al. [76] propose a feature level fusion method called *matrix-based complex PCA (MCPCA)*, for bimodal biometrics that uses a complex matrix to denote two biometric traits from one subject. The method respectively takes the two images from two biometric traits of a subject as the real part and imaginary part of a complex matrix. In order to obtain features with a small number of data items, they have devised a two-step feature extraction scheme and shown through experiments that it can achieve higher classification accuracy than other techniques, as 2DPCA and PCA. The authors used different existing unimodal databases (of ear, palm print and face images) to simulate bimodal databases and create virtual subjects for the experiments.

Kumar et al. [78] present a new approach for adaptive combination of multiple biometrics, employed to determine the optimal fusion strategy and the corresponding fusion parameters. The score-level fusion rules are adapted to ensure the desired system performance. The experimental results presented in the paper illustrate that the proposed score-level approach can achieve significantly better and stable performance over the decision-level approach. Their experiments leverage on publicly available biometric databases, which were combined one another to obtain multimodality.

Kwang et al. [30] performed a usability study for a bi-modality continuous biometrics authentication system that combines fingerprint and facial biometrics to authenticate users. The system suffers from a computational overhead of up to 42% to the computer system.

The goal of the paper by Azzini et al. [59] is to investigate if a multimodal biometric system can be used as input of a fuzzy controller for preventing user substitution. The chosen modalities are face and fingerprint. The fuzzy controller requests the fingerprint data only if the face recognition matching produces a trust level that is below a threshold. Experiments have not been performed with specific biometric systems, but simulated in different conditions. In our opinion, the explicit request for fingerprint does not seem to be a proper transparent acquisition of biometric traits, which we think is a fundamental requirement to meet usability in continuous authentication.

Sim et al. [31] present a multimodal biometric verification system that continuously verifies the presence of a logged-in user based on face and fingerprint modalities. The imposter attacks were detected within 3 s, but at the cost of an overhead of 25% of time completion for CPU-intensive tasks.

Toledano et al. [79] promote user-centered design and usability and security evaluation of biometric technologies, including fingerprint, voice and signature verification. However, the biometric modalities are studied in isolation, so they are not combined for a single authentication decision and there is no tailoring to a specific algorithm or context.

Altinok et al. [80] proposed a multi-modal biometric continuous authentication system that integrates information temporally, as well as between modalities. Simulations show that temporal integration improves authentication accuracy.

Therefore, the literature review highlights that, whereas all studies focus on security or performance evaluation, there is a real lack of usability testing in the field of continuous authentication. Furthermore, if we do not consider works specifically tailored for mobile devices, the list becomes even shorter. Our goal is to present the first -at least to our knowledge- multi-biometric continuous authentication system designed, implemented and evaluated from a user-centered perspective.

Commercial Continuous Authentication Solutions

This Section integrates the analysis of the state of the art conducted so far with a list of the main continuous authentication solutions currently available in commerce. The description is entirely based on public information accessible on companies' websites, and the mechanisms have not been tested.

ThisData [109] provides a real-time API called *Verify* which can be used to identify a user based on a behavioral profile, refined during the usage of the client application. In order to build up a behavioral profile, this solution uses risk attributes as: unknown devices, high risk IP addresses, suspicious login locations, Tor usage, white-list and blacklist countries, velocity checking, and geo-location anomalies. If the risk for a user is moderate, the administrator might ask to re-confirm their identity through a Two Factor Authentication code. If the risk is extremely high, it is possible to log them out.

NoPassword [110] offers behavioral and continuous authentication especially targeted to enterprise workforce; after successful authentication, if user suspicious or unusual behavior is detected, NoPassword can limit the user access to non-sensitive applications or ask the user to perform additional biometric authentication. The layers composing the mechanism are: online activities, screen touch pattern, smartphone handling, typing pattern, and mouse movement pattern.

BioCatch [111] proposes continuous authentication of online users selecting and verifying 20 unique features from a set of 2000 behavioral patterns. The behavioral profile is based on: (i) cognitive factors such as eye-hand coordination, applicative behavior patterns, usage preferences, device interaction patterns and responses to *Invisible Challenges*[1]; (ii) physiological factors such as left/right handedness, press-size, hand tremors, arm size and muscle usage, and (iii) contextual factors such as transaction, navigation, device and network patterns.

BehavioSec [112] analyzes the activity between login and logout, considering behavioral traits as keystroke dynamics, touchpad or mouse movements, and comparing them with the previous interaction belonging to the same user. It creates a ticket for the session and a score, which is provided to a risk engine, so that is possible to automatically intensifying security, where needed. The algorithm is updated with the behaviors and skills of the users, so if they change or improve, the same happens with the security layer.

[1] subtle tests that users subconsciously respond to

Table 2: Comparison of commercial continuous authentication solutions

Solution	Physiological traits	Behavioral traits and context info	Characteristics and use case	Issues (gap analysis)
ThisData [109]	n.a.	IP address, login location, Tor, known devices	API (Ruby, PHP, .NET)	Not biometric solution
NoPassword [110]	unimodal (unspecified)	Touchscreen, keystroke, mouse mov. apps usage, online activity	smartphone, browser desktop	Phys. biometrics only at first auth.
BioCatch [111]	arm dimension	navigation pref., pressure preferred hand	Online users	Doubts regarding recognition accuracy
BehavioSec [112]	n.a.	Keystroke, mouse mov., touchpad accelerometer, pressure,...	SDK for web and mobile	Only behavioral biometrics
Plurilock [113]	n.a.	Keystroke, mouse movement	Desktop, laptop	Only behavioral biometrics
Kryptowire [114]	n.a.	Touchscreen, pressure, movement	mobile devices	Only behavioral biom., limited use case
SensifyID [115]	n.a.	Typing, touchscreen, preferred hand, context	transactions on mobile devices	Only behavioral biom, limited use case

Plurilock [113] provides monitoring and continuous authentication tools based on behavioral biometric traits as keystroke and mouse usage. It is meant for desktop or laptop users, and is capable of learning the typing style in 30 minutes of regular keyboard usage.

Kryptowire [114] proposes a continuous authentication technology that identifies users of mobile devices through behavioral patterns. It is based on multiple on-board sensors of mobile devices including touch, pressure, movement, and power to recognize users based on the way they interact with the device and within the context of each mobile app. The proposed in-app instrumentation can detect imposters attempting to gain unauthorized access to data and services available on the device.

SensifyID [115] is a solution designed for providing continuous authentication acquiring information from sensors, and the environment. The main features are: task-based authentication (users are asked to perform a specific action as an authenticator to determine whether the user or a bot is trying to use the device, such as holding the phone and swiping across the screen), and real-time intelligence (a model is built from behaviors as typing, swiping and tapping, or the hand preferred for holding the device), proof of presence (e.g., network intelligence), and contextual information.

The commercial solutions review, summarized with the help of Table 2, highlights that almost none of them uses physiological biometrics, and in the only two cases in which this happens, either the trait is employed only for the first authentication, or its accuracy is questionable (e.g., for traits as arm dimension). In our opinion, instead, a multi-biometric continuous authentication mechanism should benefit from the advantages of physiological biometrics (possibly in multiple samples) for the whole verification procedure. Moreover, in these solutions the selected biometric traits make the recognition possible only for a limited use case as the mobile device users authentication. Finally, it is unknown whether the approaches in Table 2 have been tested or not with a proper usability assessment involving real users.

CASHMA Protocol

The continuous authentication protocol proposed in [19] improves security and usability of user session. The protocol computes adaptive timeouts on the basis of the trust posed on the user activity and in the quality and kind of biometric data transparently acquired during their activity. The overall system, shown in Figure 18, is composed of the CASHMA authentication service, the clients and the web services, connected through secure communication channels.

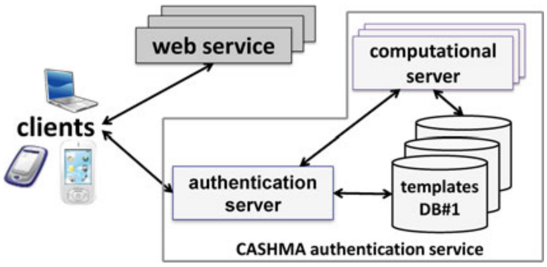


Figure 18: Overall view of the CASHMA architecture [19]

The CASHMA authentication service includes:

i) an authentication server, which interacts with the clients,

ii) a set of high-performing computational servers that perform comparisons of biometric data for verification of the enrolled users,

iii) databases of templates that contain the biometric templates of the enrolled users (these are required for user authentication/verification).

The web services are the various services that use the CASHMA authentication service and demand the authentication of enrolled users to the CASHMA authentication server. These services are potentially any kind of Internet service or application with requirements on user authenticity. Finally, clients are the users' devices that acquire the biometric raw data corresponding to the various biometric traits from the users, and transmit those data to the CASHMA authentication server as part of the authentication procedure towards the target web service. The CASHMA authentication server is in charge to transmit a certificate to the client. The certificate is composed by the following information:

i) Timestamp and sequence number useful to univocally identify each certificate, and to protect from replay attacks;

ii) ID is the user ID, e.g., a number;

iii) Decision, which represents the outcome of the verification procedure carried out on the server side;

iv) the expiration time of the session – the absolute instant of time at which the session should expire– , dynamically assigned by the CASHMA authentication server.

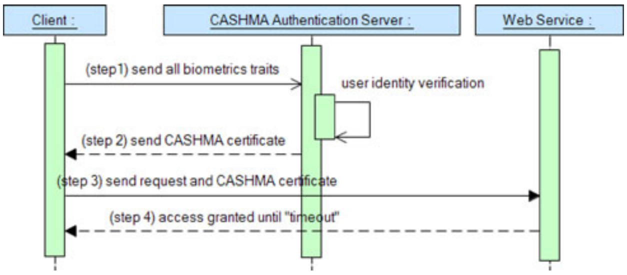

Client :
CASHMA Authentication Server :
Web Service :
(step1) send all biometrics traits
user identity verification
(step 2) send CASHMA certificate
(step 3) send request and CASHMA certificate
(step 4) access granted until "timeout"

Figure 19: Initial phase of CASHMA protcol [19]

The execution of the protocol is composed of two consecutive phases: the initial phase (Figure 19), and the maintenance phase.

Initial phase. This phase is structured as follows:

- Step 0 - The user (the client) contacts the web service for a service request; the web service replies that a valid certificate from the CASHMA authentication service is required for authentication.

- Step 1 - Using the CASHMA application, the client contacts the CASHMA authentication server. During the first step, at time t_0, the client device acquires and sends the different biometric traits, specifically selected to perform a strong authentication procedure. The application explicitly indicates to the user the biometric traits to be provided and possible retries.

- Step 2 - The CASHMA authentication server analyzes the biometric data received and performs an authentication procedure. Two different possibilities arise here. If the user identity is not verified (the global trust level is below the trust threshold g_{min}), new or additional biometric data are requested (back to step 1) until the minimum trust threshold g_{min} is reached. Instead if the user identity is successfully verified, the CASHMA authentication server authenticates the user, computes an initial timeout of length T_0 for the user session, set the expiration time at $T_0 + t_0$, creates the CASHMA certificate and sends it to the client.

- Step 3 - The client forwards the CASHMA certificate to the web service coupling it with its request.
- Step 4 - The web service reads the certificate and authorizes the client to use the requested service until expiration time.

The maintenance phase is composed of three steps, analogous to Step 1, Step 2 and Step 3, repeated iteratively [19].

Finally, we report the following set of security solutions in place for CASHMA and some assumptions which will be important for the following of the Thesis. First, all communications between the components of CASHMA architecture are made through encrypted communication channels, using Secure Sockets Layer (SSL). With SSL, the channel is protected against replay attacks using the MAC (Message Authentication Code) computed from the MAC secret, the sequence number, the message length, the message contents, and two fixed character strings [85]. Further, biometric data is transmitted in raw format from client to the CASHMA authentication service, and this has been a design decision applied to reduce the dimension, intrusiveness and complexity of the application installed on the client device [19]. However, biometric data in not stored on the client side: the templates are stored on the CASHMA authentication service side.

Part II

OUR APPROACH TO CONTINUOUS AUTHENTICATION

5 DESIGN AND EVALUATION OF A CONTINUOUS AUTHENTICATION MECHANISM

The present chapter describes our approach to continuous authentication. The solutions proposed here and in the following chapters have some common features with CASHMA [19]. For this reason, in Section 5.1 we first provide a risk assessment of [19] that supports decisions related to its modifications. Then, in Section 5.2, we present the proposed mechanism, its design, prototyping and implementation. Finally, in Section 5.3, we explain how we assessed and evaluated our solution, especially focusing on usability, security and their trade-off.

5.1 RISK ASSESSMENT OF CASHMA CONTINUOUS AUTHENTICATION PROTOCOL

In this Section we perform a qualitative risk assessment of the CASHMA protocol [19], already presented in the previous chapter. The assessment is based on the methodology of NIST SP-800-30 [55], and its purpose is to establish if some of the steps of the protocol and of the entities involved may be exposed to relevant security risks. This activity supports decisions related to protocol modifications and improvements, where needed, and will be an important step for designing the continuous authentication and non-repudiation solutions presented in this Thesis.

It has to be noticed, however, that the result of the assessment may always be considered subjective by the reader, even if it has been conducted in compiance with a well known methodology and with the help of a standard scoring framework [84].

5.1.1 *Definitions*

In order to describe the details of the risk assessment, we first introduce some useful definitions from NIST SP-800-30 [55].

A **threat** is any circumstance or event with the potential to adversely impact a system via unauthorized access, destruction, disclosure, or modification of information, and/or Denial of Service (DoS). Threat events are caused by threat sources, i.e., hostile cyber or physical attacks; human errors of omission or commission; structural failures of organization-controlled resources (e.g., hardware,

Table 3: Attackers and their characteristics

	Hacker (Hk)	*Insider (Is)*
Access	External	Internal
Resources	Moderate	Organization
Capabilities	High	Minimal

software, environmental controls); natural and man-made disasters, accidents, and failures beyond the control of the organization.

We already introduced in Section 2.2 the term **vulnerability**. In the specific context of this assessment, we also report to the NIST definition: a weakness in the system security procedures, internal controls, or implementation that can be exploited by a threat source.

The **likelihood** of occurrence is a weighted risk factor based on an analysis of the probability that a given threat is capable of exploiting a given vulnerability (or set of vulnerabilities). The likelihood risk factor combines an estimate of the likelihood that the threat event will be initiated with an estimate of the likelihood of impact (i.e., the likelihood that the threat event results in adverse impacts). For adversarial threats, an assessment of likelihood of occurrence is typically based on adversary:

(i) *intent;*

(ii) *capability;*

(iii) *targeting.*

The level of **impact** from a threat event is the magnitude of harm that can be expected to result from the consequences of unauthorized behavior.

Risk is a function of the likelihood of a threat event's occurrence and potential adverse impact should the event occur.

5.1.2 *The Assessment Methodology*

We first determine which types of threat sources have to be considered during the risk assessment. We consider adversarial threat sources only, as opposed to non-adversarial like human errors, structural failures or natural disasters. The characteristics of the identified attackers are summarized in Table 3 and are adapted from the quantitative security evaluation of [19].

A *Hacker (Hk)* represents an external individual having high technological capabilities but moderate resources. An *Insider (Is)* is an internal attacker, having minimal capabilities and organization-level resources.

The threat events that we consider can be divided into two categories [6, 81, 82, 83]:

- *Transport level threats*: spoofing, forgery, unauthorized access, eavesdropping, message corruption;
- *System level threats*: brute force attack, reuse of residuals, insertion of imposter data, component replacement, database compromise.

In the following, we rate the likelihood of occurrence as a combination of

- the likelihood that a threat is initiated, possibly related to the attack gain, and
- the likelihood that a threat event, once initiated, will result in adverse impact.

The likelihood of occurrence determination is based on authors' experience and obtained with the help of CVSS framework [84]. The overall likelihood is expressed in a qualitative scale: *Very Low (VL)*, if the threat event is highly unlikely to occur and have adverse impact, *Low (L)* if it is unlikely, *Moderate (M)* if it is somewhat likely, *High (H)* if it is highly likely, and *Very High (VH)* if it is almost certain.

As discussed in [55], impacts from threat events are determined considering

- the characteristics of the threat sources that can initiate the events;
- the vulnerabilities/predisposing conditions identified;
- the susceptibility reflecting the safeguards/countermeasures planned or implemented to impede such events.

The assessment scale is: *Very Low (VL)*, meaning that the adverse effect is negligible; *Low (L)*, if the impact is expected to be limited; *Moderate (M)*, if the threat event is expected to have a serious adverse effect; *High (H)*, in case of severe or catastrophic impact; and *Very High (VH)*, if the threat event is expected to have multiple severe or catastrophic adverse effect.

As summarized in Table 4, risk is a combination of:

- the likelihood of the events occurring.
- the impact resulting from the events;

So, as an example, based on the methodology in [55], an event which we assess having Moderate likelihood and a Very High impact will result in a High risk.

In the following, we call: Hk and Is, the attackers (as in table 3), being a Hacker or an Insider, respectively; C, the legitimate client; AS, the Authentication Server;

Table 4: Level of risk determination. [55]

Likelihood	Level of Impact				
	Very Low	Low	Moderate	High	Very High
Very High	Very Low	Low	Moderate	High	Very High
High	Very Low	Low	Moderate	High	Very High
Moderate	Very Low	Low	Moderate	Moderate	High
Low	Very Low	Low	Low	Low	Moderate
Very Low	Very Low	Very Low	Very Low	Low	Low

and WS, the Web Service. In order to not being too redundant, the assessment reported here only regards the initial phase of the protocol, but it can be easily extended to the maintenance phase too, which is analogous.

For each threat event, and for both the attackers, we explain the reasons and our perceptions guiding the likelihood and impact determination. Then, we propose countermeasures -where possible-, and repeat the assessment assuming their inclusion in a new hypothetical version of CASHMA. The process is summarized with the help of Tables 5- 9.

5.1.3 *Transport Level Threats Analysis*

Spoofing

Table 5: Spoofing Threats in CASHMA

Threat	Description	Attacker	Risk Assessment			Countermeasures	New Risk Assessment		
			Likelihood	Impact	Risk		Likelihood	Impact	Risk
Spoofing S1	X pretends towards AS to be C (step 1)	Hk	H	H	H	1) Digital signature of the message. 2) Additional encryption of the message with receiver's public key. 3) Do not transmit raw biometrics over the wire: encrypt biometric data on client-side	M	H	M
		Is	H	H	H		M	H	M
Spoofing S2	X pretends towards C to be AS (step 1 and 2)	Hk	H	H	H		M	H	M
		Is	M	H	M		L	H	L
Spoofing S3	X pretends towards C to be WS (step 3)	Hk	H	M	M		M	M	M
		Is	M	M	M		L	M	L
Spoofing S4	X pretends towards WS to be c (step 3)	Hk	M	H	M		L	H	L
		Is	M	H	M		L	H	L

Table 6: Forgery Threats in CASHMA

Threat	Description	Attacker	Risk Assessment			Countermeasures	New Risk Assessment		
			Likelihood	Impact	Risk		Likelihood	Impact	Risk
Forgery F1	The attacker fabricates one or more biometrics to spoof C's identity	Hk	L	H	L	Do not transmit raw biometrics over the wire encrypt data on client-side	L	H	L
		Is	M	H	M		L	H	L
Forgery F2	X fabricates a fake certificate, claiming that was received from AS	Hk	M	H	M	Digital signature of the message	L	H	L
		Is	L	H	L		L	H	L

The spoofing threat (often also referred as *masquerade*), is a communication level threat that happens when an entity pretend to be recognized as a different entity, possibly laying the foundation for other threats like forgery or unauthorized access [81]. For instance, it can be accomplished using stolen user credentials. Traditional countermeasures are: the usage of strong authentication, avoiding the storage of secretes (e.g. passwords) in plaintext, avoiding transfer of credentials in plaintext over the wire, and protecting communication with SSL [82]. Based on the protocol description and assumptions, CASHMA is integrating most of the countermeasures to this threat. However, we identified four different points of vulnerability, and the related threats are referred as S1, S2, S3 and S4, shown in Table 5.

The threat Spoofing S1 specifically refers to step 1 of the CASHMA protocol (see Section 4.1.2): the adversary injects believable biometric raw data into a message, claiming to be C, and obtains a valid certificate from the AS. Noteworthy, the attacker needs previous possession of the biometric raw data of the legitimate user. An Insider may be somehow capable of acquiring the biometrics needed, helped by the proximity and knowledge of C habits. In order to perform step 1 of the protocol and act as being C, it should send the data to the AS. This may be harder for Is because we consider that high skills are needed to circumvent the SSL protocol. We consider as High the overall likelihood. On the contrary, if the threat agent is a Hk, it may be hard to obtain access to the set of biometric raw data remotely. If data are obtained, the Hk may possess enough capabilities to circumvent the SSL protocol. The overall likelihood is High also for Hk. If the attacker is able to accomplish the spoofing attack, it may lay foundations for subsequent threat events: being recognized as C, and obtained a valid certificate, X may send a valid request to the Web Service. For this reason, we determine as High the Impact of this threat. As shown in Table 5, the resulting Risk is High for both the attackers.

The threat Spoofing S2 refers both to steps 1 and 2 of the protocol. On step 1, the attacker spoofs the AS, receives the biometric traits of C, with detrimental effects on C's privacy. In addition it can be used in step 1 of a subsequent iteration of the protocol to act as being C towards AS (Spoofing S1). As a result, we consider High the Impact of this threat. On step 2, instead, if X can corrupt a certificate and claim to be the AS, the corruption probably provides a fake and useless certificate to C. We consider Moderate the likelihood of spoofing the AS for the Insider and High the likelihood of occurrence of S2 for the Hacker.

The threat Spoofing S3 applies to step 3 of the protocol. The attacker spoofs the WS, receiving the request and a valid certificate from C. These data can be useful in step 1 to spoof C (see Spoofing S1). However, a CASHMA certificate integrates information like timestamp and user ID that protect from replay attacks (see also Message corruption M1). For this reason the impact can be considered Moderate: the gain is not so relevant if the obtained data is not useful for the attacker.

The threat Spoofing S4 applies to step 3 of the protocol. The attacker sends a request with a valid certificate, obtaining the access to the service provided by WS. The impact is High, but we consider the successful reuse of a valid certificate moderately likely, because it is univocally identified by Timestamp and sequence number.

Possible countermeasures for threats S1-S4 range from sending the message coupled with its digital signature (measure useful for auditing and as a deterrent), adding a further level of encryption using AS's public key or avoid passing the raw biometric data over the wire.

Forgery

The forgery happens when an entity fabricates information and claims that such information was received from another entity or sent to another entity [81]. For the CASHMA protocol, we discuss two main forgery threats, F1 and F2, shown in Table 6.

For threat Forgery F1, the attacker fabricates one or more biometric traits in order to spoof C's identity. The occurrence likelihood depends on the kind of biometric traits necessary for the authentication, on their FRR (False Rejection Rate) and on the FRR of the system. However, as discussed for threat S1, it is probably not highly likely to forge a set of traits and provide them continuously for the remote Hk, and moderately likely for the Is. For this reason, even if the threat has a High impact (especially if the application protected is critical), the risk can be considered low for Hk. Instead, we consider a Moderate risk for the Is. An additional countermeasure useful for reducing this risk is to not transmit raw biometric over the wire. This threat is related to Sensor Spoofing (See Table 10).

The threat Forgery F2 refers to the forgery of a CASHMA certificate. We can consider Unlikely for the Insider attacker to forge a certificate and Moderate the likelihood of occurrence if the threat source is Hk. The impact is High. As a countermeasure, the message containing the certificate sent on step 3 to the WS, should be digitally signed by the sender.

Unauthorized Access

When an entity accesses data in violation to the security policy in force [81], we have the so called unauthorized access threat event. We now comment the two threats U1 and U2, shown in Table 7.

With the threat Unauthorized access U1, we refer to the event of getting (physically or remotely) possession of C's device or workstation, but not the critical functions only provided through continuous authentication. We consider the likelihood for a remote attacker (Hk) as Moderate, and High for the Is.

Table 7: Unauthorized Access Threats in CASHMA

Threat	Description	Attacker	Risk Assessment		
			Likelihood	Impact	Risk
Unauthorized Access U1	The attacker gets possession of C's device or workstation	Hk	M	L	L
		Is	H	L	L
Unauthorized Access U2	X gets access to data protected by the continuous authentication	Hk	L	H	L
		Is	L	H	L

However, the resulting impact and risk are Low because, as reported in 4.1.2, no biometric data is stored on the device.

The threat Unauthorized access U2 refers instead to the access to the critical functions/services protected with the continuous authentication. Proper configuration of security permission is able to mitigate this threat. No modifications to the protocol are required in this case. We rate the likelihood and the related risk as Low. No additional countermeasures are proposed for these two threats.

Eavesdropping

Eavesdropping (or Sniffing) is a breach of confidentiality by unauthorized monitoring of communications [81]. We detect the threats E1, E2, and E3 for CASHMA, shown in Table 8. A sniffing happens when an attacker captures packets from the network and reads the data content in search of sensitive information like biometric data, secret keys, certificates or any kind of confidential information. Traditionally, the full encryption of communication, including credentials, prevents sniffed packets from being usable to an attacker. SSL is an example of encryption solution [82].

The threat Eavesdropping E1 refers to the sniffing of the message on step 1, containing the biometric raw data of C. Referring to Section 4.1.2, the message on step 1 is transmitted through an SSL channel, thus it is encrypted with a session key that only C and AS know. We consider Moderate the likelihood of occurrence for E1 if the attacker possesses high capabilities, as the Hk, and Low if it is the Is. The impact is High, because obtaining the biometrics may cause identity theft. Possible countermeasures are a subset of what is proposed for the Spoofing threats: adding a further level of encryption using AS's public key or avoid passing the raw biometric data over the wire.

The threat Eavesdropping E2 refers to the sniffing of the message on step 2 or step 3, containing the CAHSMA certificate. Again, we assess the likelihood as Moderate for Hk and Low for Is. The impact can be considered Moderate, as discussed for Spoofing S3 threat, being that the CASHMA certificate contains ID and timestamp that protect from replay attacks.

Table 8: Eavesdropping Threats in CASHMA

Threat	Description	Attacker	Risk Assessment			Countermeasures	New Risk Assessment		
			Likelihood	Impact	Risk		Likelihood	Impact	Risk
Eavesdropping E1	X sniffs the message on step 1 with the biometrics of C	Hk	M	H	M	1) Encrypt the msg with receiver's public key 2)Not send raw traits: encrypt on client	L	H	L
		Is	L	H	L		L	H	L
Eavesdropping E2	X sniffs the message on step 2 or step 3, containing the certificate	Hk	M	M	M	Additional encryption of the message with receiver's public key	L	M	L
		Is	L	M	L		L	M	L

Table 9: Message Corruption Threats in CASHMA

Threat	Description	Attacker	Risk Assessment			Countermeasures	New Risk Assessment		
			Likelihood	Impact	Risk		Likelihood	Impact	Risk
Msg corruption M1	X copies a message (step 1 to 4) and replays it to the receiver	Hk	M	L	L	Do not transmit raw biometrics over the wire encrypt biometrics on client side	L	L	L
		Is	L	L	L		L	L	L
Msg corruption M2	X deletes a message (steps 1 to 4)	Hk	M	L	L	n/a	n/a	n/a	n/a
		Is	L	M	L		L	M	L
Msg corruption M3	X inserts a msg into the flow from C to A or from C to WS (and vice versa)	Hk	M	L	L	1) Encryption with receiver's public key 2) Do not transmit raw biometrics	L	M	L
		Is	L	M	L		L	M	L
Msg corruption M4	X changes the sequence of messages for AS	Hk	M	L	L	n/a	n/a	n/a	n/a
		Is	L	L	L		n/a	n/a	n/a
Msg corruption M5	X alters a msg in plausible way: C and/or WS do not detect the changes	Hk	M	M	M	1) Digital signature of the msg 2) Encryption with receiver's public key	L	M	L
		Is	L	M	L		L	M	L

Message Corruption

The message corruption threats refer to the situation in which the integrity of transferred data is compromised by unauthorized copying, deletion, insertion, modification, reordering, replay or delay [81].
For the CASHMA protocol, we detect the set of message corruption threats shown in Table 9. Referring to Section 4.1.2, the SSL encryption and the MAC address protect the protocol from this threat.

The threat Message corruption M1 happens when the attacker copies and replays one of the messages, for instance the message sent on step 1 containing the biometrics of C. We consider as Low the impact of the simple replaying of a message (without modification) thanks to the SSL encryption, already present in CASHMA, as discussed above.

The threat Message corruption M2 applies to the deletion of a message sent on steps from 1 to 4.
If the message on step 1 with biometric traits is deleted, no verification is done. However, it does not have any relevant consequence in terms of security.
If the message on step 2 containing the certificate is deleted, the client has to restart from the first step of the protocol.
If the message sent on step 3 with the certificate is deleted, the client has to send the same message again or, if the session is expired, restart from step 1.
Finally, if the message *"access granted until timeout"* (step 4) is deleted, C waits for the message until the session expiration.
We may consider all these threats together having a Low impact and a resulting Low risk.

The threat Message corruption M3 refers to the insertion of a message in between the information flow from sender and receiver (from C to AS, from AS to C, from C to WS or from WS to C). The impact is Moderate, being that the insertion of a message can be used for instance to conduct a spoofing threat. However, in CASHMA, messages from one party cannot be inserted into the other's output, since they use independent MAC secrets [86]. Consequently, the likelihood for this threat is set to Low. The resulting risk is set to Moderate if the attacker is Hk, and to Low for the Is.

The threat Message corruption M4 represents a resequencing attack conducted against AS. Again, it is not likely due to the encryption obtained with SSL. Moreover, the impact is limited. For instance, inverting messages in step 2 and 4 would make C thinking to be able to access the WS, when actually it is not. So it just results in unsatisfied user due to unavailable or delayed service.

For threat Message corruption M5, we can distinguish four events, one for each step from 1 to 4.

Step 1 : The attacker corrupts the message with the biometric traits of C, but cannot act as C. The corruption makes the verification procedure to reject C, causing a DoS.

Step 2 : The attacker corrupts the message containing the certificate, alters the timestamp, the sequence number, the expiration time, the decision, with remarkable consequences for the subsequent steps.

Step 3 : The attacker alters a valid certificate, generated for C when it is being sent to the WS. This impacts the next step.

Step 4 : if messages are corrupted by X, the WS denies the access for C or grant it for a time window shorter than normal, thus provoking a Denial of Service.

All considered, we rate the impact of this threat as Moderate, and the likelihood as Moderate for the Hk and Low for the Is. The resulting risk and proposed countermeasures are shown in Table 9.

5.1.4 *System Level Threats Analysis*

In this Section we analyze a set of system level threats, shown in Table 10, which are typical of a biometric system [83].

Brute Force

Brute force attacks in general rely on computational power to crack secrets secured with hashing and encryption.

The Brute Force B1 threat refers to the submission of a huge set of biometric traits to the sensors embedded or connected to the client workstation with the objective of finding a trait (or a set of traits) that let the attacker to be authenticated. However, the CASHMA protocol, as it is, should maintain Low the impact of this threat, because the number of retries is limited, as discussed in 4.1.2. As a result, in our assessment the risk is also Low for both the attackers.

Sensor Spoofing

An attacker provides a set of fake physical or digital biometric traits designed to circumvent the biometric system. The CASHMA system integrates a multimodal biometric system: the user has to provide a set of traits in order to be authenticated. The multi-modality itself can be considered a first defensive measure to this threat [83]. The impact of this threat is High. We assess a Moderate risk for the Is attacker, which may have direct access to the device and the biometric sensors. An additional countermeasure that can reduce the risk is the liveness detection to ensure the biometric sample presented to the reader is from a live person.

Reuse of Residuals

An attacker gains somehow access to valid biometric data and reuses it. As discussed in the assumptions, no biometric data is stored on client's device. However, it is important to defend from this threat, because the attacker may obtain data in other ways (i.e. through Eavesdropping). The risk is Moderate and the Impact is considered High for the Hk attacker. An effective defensive measure is prohibiting identical samples being used consecutively.

Database Compromise

An attacker obtains access to template repository and is able to read, modify or substitute templates. We consider Low the likelihood of this threat: we assume that the CASHMA authentication server implements a very efficient and complete set of access control mechanisms. However, the impact of this threat is very high for template security and users' privacy, and, as a consequence, the risk for Hk is Moderate.

To reduce the risk, it may be necessary to introduce additional defensive measures. A viable solution may be cancelable biometrics, thanks to which data are stored after a transformation, and if these data are compromised, a new transform can be applied, thus replacing the original template [83, 87].

With the integration of this countermeasure, or an alternative template protection approach [88], we imagine a reduction of the Impact due to database compromise from Very High to High, which together with a Low likelihood determines a Low risk.

Table 10: System Level Threats in CASHMA

Threat	Description	Attacker	Risk Assessment			Countermeasures	New Risk Assessment		
			Likelihood	Impact	Risk		Likelihood	Impact	Risk
Brute Force	X submits many traits to find a matching	Hk	L	L	L	n/a	n/a	n/a	n/a
		Is	M	L	L		n/a	n/a	n/a
Sensor Spoofing	X uses fake traits to circumvent the system	Hk	L	H	L	Liveness detection	L	H	L
		Is	M	H	M		L	H	L
Reuse of residuals	X gets access to biometrics and reuses them	Hk	M	H	M	Forbid consecutive use of identical samples	L	H	L
		Is	L	H	L		L	H	L
Database Compromise	X gets access to templates and can corrupt them	Hk	L	VH	M	Cancelable biometrics or or other template protection	L	H	L
		Is	VL	VH	L		VL	H	L

The assessment of CASHMA highlighted the threats that, in our opinion, have to be addressed in order to reduce security risks connected to its usage. The CASHMA approach can be considered a starting point for the continuous authentication and non-repudiation solutions presented in this Thesis. Its assessment is part of a design and requirements definition process, and is performed in order to guide the development of our own authentication and non-repudiation solutions. In particular, some of the countermeasures highlighted here are included in protocols described in Chapters 6 and 7, which are an improvement of CASHMA targeted to the inclusion of non-repudiation service.
The solution presented in the current Chapter, -from Section 5.2 on-, instead, is a different approach which still possesses some common features with [19], thus the assessment is useful for its design process.

5.2 THE PROPOSED MECHANISM

The solution proposed in this Section aims at providing continuous authentication, especially focusing on maintaining a high usability, and being contemporaneously general enough to be adapted to different desktop scenarios.

5.2.1 *Application Scenario*

In fact, the mechanism has been thought for human operators of desktop workstation but we think that it is applicable to desktop users in general. This solution has been integrated in the prototype of the the Secure! [156] crisis management system. In that scenario, operators are in charge of analyzing and interpreting situations that describe the current status of an emergence. Using the information available, the operator from his workstation (mainly via text messages, using a keyboard) is able to command intervention teams on field, and to dispatch instructions to civilians in the target area.
It is required to protect the workstation from unauthorized people (intruders) and insiders that may want to acquire privacy-sensitive data, disrupt the crisis management operations, disseminate false information, or simply commit errors, which will be ascribed to the operator in charge of the workstation.
In order to protect the workstations, we need to guarantee authenticity of the commands/functions executed, meaning that commands that are transmitted and expected from an operator, are actually generated from him or her.

5.2.2 *The BCAS System Architecture*

The system we designed is called Biometric Continuous Authentication System (from now BCAS) and it is composed of:

- A *desktop workstation*, including sensors for the biometric signals acquisition.
- An *authentication server*, responsible for receiving the biometric traits digitalized by the workstation;
- A *database of templates* in which the traits are stored.

The choice of the biometric traits may depend on the scenario and the sensors available. The solution described here, in our opinion, fits the workstation scenario, but the BCAS system can easily be adapted to different scenarios and choices of biometric traits.

Based on the characteristics of a generic workstation, the user interface typically consists at least of a screen, a keyboard, and a mouse. In our opinion, the best choice to achieve system acceptability is to avoid the introduction of any additional device with which the user actively interacts. In this way, there is no loss of time spent in learning how to use the device and, consequently, no loss of proficiency and efficiency in the working activity. The system only requires the usage of a particular kind of mouse that incorporates a fingerprint scanner where users normally place their thumb [9]. This measure may be unpleasant but is necessary; otherwise the biometrics acquisition would not be possible in a transparent way. Then, the other sensors are a keyboard and a camera which nowadays is very common to be integrated in a laptop or on top of the screen of a workstation, e.g., for usage in video conferences.

The BCAS is therefore based on three unimodal biometric subsystems, for fingerprint recognition, face recognition, and keystroke recognition. Each subsystem is composed of hw/sw elements necessary for the acquisition of the trait and for the verification process, including sensors and recognition algorithms, such that each one is able to decide independently if the user is genuine or not. The fusion is performed at decision level.

These three biometric traits have different levels of performance and measurability [10, 18], and we think that they well complement each other. In fact, high measurability of facial images will help covering temporal gaps that could exist between two fingerprint acquisitions, when the operator is not using the mouse. Keystroke supports the other two traits, despite its lower performance (as described in Section 3.2), and it can be useful especially when fingerprint acquisitions are missing.

5.2.3 *The Protocol*

The proposed continuous authentication protocol is shown in the sequence diagram of Figure 20. It is divided in two phases: the *initial phase* and the *maintenance phase*. Before the initial phase, we assume that the enrollment already took place as a preliminary step.

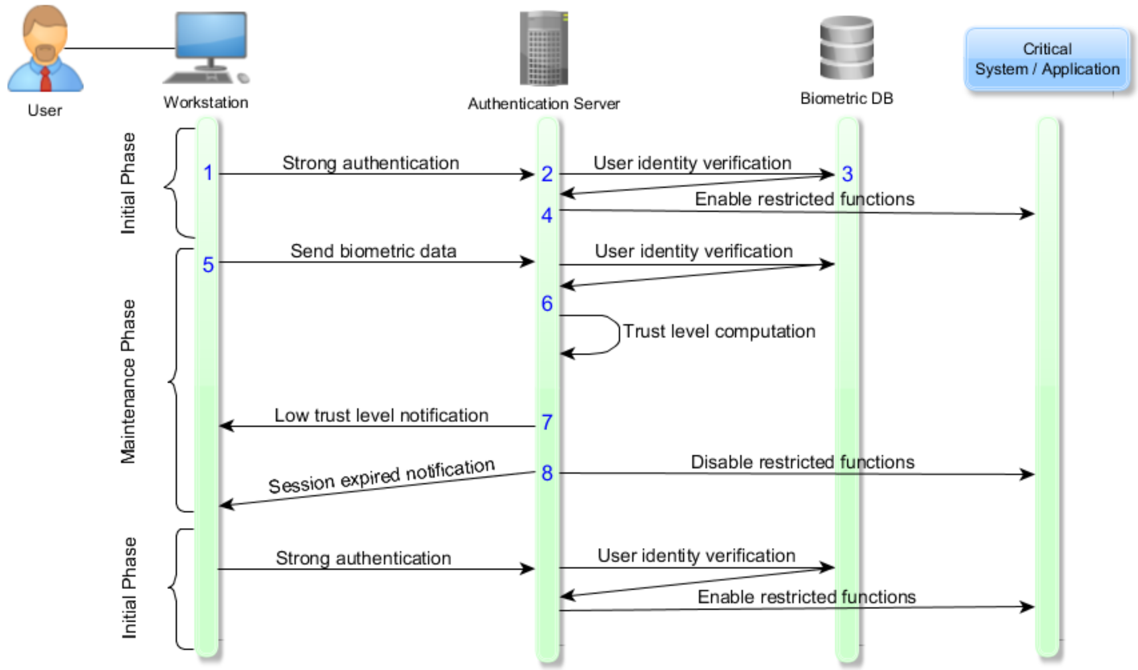

User
Workstation
Authentication Server
Biometric DB
Critical System / Application
Initial Phase
Maintenance Phase
Initial Phase
1
Strong authentication
2
User identity verification
3
4
Enable restricted functions
5
Send biometric data
User identity verification
6
Trust level computation
Low trust level notification
7
Session expired notification
8
Disable restricted functions
Strong authentication
User identity verification
Enable restricted functions

Figure 20: Sequence diagram of the protocol

Initial phase

It is composed of the following steps:

1 The user logs in and each subsystem performs the biometric verification in a short time interval. At this time instant, indicated with t_0, the trust is set to $\mathtt{trust}(t_0) = 1$.

2 Biometric data is acquired by the BCAS workstation and transmitted to the authentication server.

3 The authentication server matches the user's templates with the traits stored in the database and verifies his/her identity.

4 In case of a successful verification, the BCAS application establishes a session and allows access to restricted functions.

Maintenance phase

The biometric continuous authentication protocol works as follows:

5 The user's biometric data are periodically acquired by the biometric subsystems operating on the workstation and are transmitted to the authentication server.

6 The authentication server waits for fresh biometric data, from any of the three subsystems. When new biometric data is available, it verifies the identity claimed by the user and, depending on the comparison results of each subsystem, it computes and updates $\mathtt{trust}(t)$.

7 The session expires when $\mathtt{trust}(t_i)$ becomes lower than $\mathtt{trust}_{min}$.

8 When the trust level is below the threshold, $\mathtt{trust}(t) < \mathtt{trust}_{min}$, the session expires and the restricted functions are disabled. The user receives a notification of this event, and, if necessary, restarts again from the initial phase.

5.2.4 *The Trust Level Computation*

We describe the algorithm executed by the authentication server to compute the trust level. Our system integrates three unimodal biometric subsystems

- S_1 for fingerprint recognition,
- S_2 for face recognition,
- S_3 for keystroke recognition

such that each one is able to decide independently if the user is genuine or not. The algorithm which computes the trust level is executed periodically on the authentication server as follows. During the maintenance phase, the authentication server verifies the user identity thanks to all biometric data provided in a specific time interval. In our implementation, this interval is 20s, during which multiple attempts of fingerprint, and face acquisition are sequentially performed, while a keystroke listener runs in parallel for almost the whole interval. In general, let us consider the time interval $[t_{i-1}; t_i]$, where t_i is the current time instant and t_{i-1} is the time instant in which the previous iteration of the protocol has been concluded. Regarding the status of the system at time instant t_i, we have three following alternatives: *three recognitions*, *two recognitions*, *one or no recognitions*.

Three recognitions: for any time interval in which all the three biometric subsystems led to successful verifications, the authentication server sets $\mathtt{trust}(t_i) = 1$.

Two recognitions: two-out-of-three biometric subsystems led to successful verification. The trust level is updated to a static value, which can be set a priori based on the estimated accuracy of the subsystems that decided the user legitimacy. The trust level is computed following Equation 5.1

$$\mathtt{trust}(t_i) = m(S_{k1}) + (r \cdot m(S_{k2})) \tag{5.1}$$

where:

Table 11: Trust computation with two out of three successful recognitions

Pair of biometric subsystems	$\mathrm{trust}(t_i)$
Fingerprint, Face	$0.9 + (0.1 \cdot 0.8) = 0.98$
Fingerprint, Keystroke	$0.9 + (0.1 \cdot 0.7) = 0.97$
Face, Keystroke	$0.8 + (0.1 \cdot 0.7) = 0.87$

- S_{k1} and S_{k2} are the subsystems which correctly verified the identity of the user, and S_{k2} is the one with the lower performance;
- r is a parameter to weight $m(S_{k2})$ in order to have $\mathrm{trust}(t_0)$ between 0 and 1.

In our implementation, setting $r = 0.1$, $m(S_{k1}) = 0.9$, $m(S_{k2}) = 0.8$, $m(S_{k3}) = 0.7$, we have the combinations shown in Table 11 .
The selection of these values has been conducted comparing the biometric traits, analyzing how their performance is evaluated in literature (e.g., in [10, 18]), and it is related to the number of false accepts produced by each subsystem. We found that these values can properly represent the accuracy of each subsystem, but other different values can be easily adopted, if necessary, following a similar approach.

One or no recognitions: if instead, at most one biometric verification is successful at time instant t_i, $\mathrm{trust}(t_i)$ decreases nonlinearly through time. Given $\mathrm{trust}(t_{i-1})$, that is the trust level computed at the previous iteration of the algorithm, we have that $\mathrm{trust}(t_i)$ will be smaller than $\mathrm{trust}(t_{i-1})$. Its value is given by 5.2 (from [19])

$$\mathrm{trust}(t_i) = \frac{(-\arctan((\Delta t_i - s) \cdot k) + \frac{\pi}{2}) \cdot \mathrm{trust}(t_{i-1})}{-\arctan(-s \cdot k) + \frac{\pi}{2}} \tag{5.2}$$

where $\Delta t_i = t_i - t_{i-1}$ and k are introduced to tune the decreasing function: k affects the inclination towards the falling inflection point. In regard to [19], in 5.2 we set the value of s, the parameter which allows delaying the decay. Through an experimental evaluation, we found that 5 is the most appropriate value to manage the delay in our setup.
The selection of k, in particular, affects the speed of the decrease of the trust level. We adopt three different values of k according to which and how many verifications are successful. A *fast* decrease is set when no verifications are successful or no biometric data is transmitted. The decrease is said *average* if only one verification is successful, for any biometric subsystem. Finally, we have a *slow* decrease if face is correctly verified and keystrokes are detected but not

recognized, either because data is not sufficient to perform keystroke recognition or because keystroke recognition fails.
The latter is the situation in which:

- the user is actually busy in the usage of the keyboard,
- the user is not able to send any fingerprint data,
- the amount of keys pressed is too low or too sparse to permit keystroke recognition.

Thus, a small penalization is assigned to the trust in the user, smoothly decreasing the trust level. The triples of *k* values selected for the experiments are discussed in the following of the Thesis.
Algorithm 1 summarizes in pseudo-code what has been described so far about the trust level computation.

Algorithm 1 Trust Level Computation

```
function InitialPhase
    if user $u$ is strongly authenticated or accepted by 3 subsystems then
        $\mathrm{trust}(t_0) = 1$.
    else if user $u$ is accepted by 2 subsystems then
        $\mathrm{trust}(t_0) = m(S_{k1}) + (r \cdot m(S_{k2}))$.
    MaintainancePhase($\mathrm{trust}(t_0)$)
function MaintainancePhase($\mathrm{trust}(t_{i-1})$)
    Try to collect biometric data and perform verifications
    if user $u$ is accepted by 3 subsystems then
        $\mathrm{trust}(t_i) = 1$.
    else if user's face is recognized and keyboard usage is detected then
        $\mathrm{trust}(t_i)$ =TrustDecaying($\mathrm{trust}(t_{i-1})$, slow).
    else if user $u$ is accepted by 1 subsystem then
        $\mathrm{trust}(t_i)$ =TrustDecaying($\mathrm{trust}(t_{i-1})$, average).
    else if user $u$ is rejected by all subsystems then
        $\mathrm{trust}(t_i)$ =TrustDecaying($\mathrm{trust}(t_{i-1})$, fast).
    if $\mathrm{trust}(t_i) \geqslant \mathrm{trust}_{min}$ then
        MaintainancePhase($\mathrm{trust}(t_i)$).
function TrustDecaying($\mathrm{trust}(t_{i-1})$, k)
    return $\frac{(-\arctan((\Delta t_i - s) \cdot k) + \frac{\pi}{2}) \cdot \mathrm{trust}(t_{i-1})}{-\arctan(-s \cdot k) + \frac{\pi}{2}}$
```

It is important to say that, in our prototype, the trust computation is only influenced by the number of successful verifications, and not by unsuccessful verifications. In fact, as an implementation choice, we do not distinguish between

Figure 21: The SecuGen OptiMouse Plus

a missing biometric characteristic and a trait verified as not legitimate. This is to favour usability, considering also the high number of false rejects that may happen under different operating conditions, for instance, when the fingers are sweat, or the room is darker than usual. However, an alternative solution may address this difference: if a subsystem considers the trait belonging to an impostor, this may cause a faster trust level decreasing w.r.t. a missing acquisition of the same trait.

5.2.5 *The Prototype*

Together with the protocol design, we have developed a prototype of the BCAS system aiming at assessing the applicability of the proposed solution and set it up since the early stages of development. In this Section we describe the prototype from different perspectives, including hardware components, biometric sensors, implementation choices, and the algorithms adopted for biometric recognition in each of the subsystems.

Hardware Prototype

The hardware is entirely COTS (Commercial of-the Shelf). For fingerprint acquisition, our choice is the SecuGen OptiMouse Plus mouse [9], shown in Figure 21, which incorporates an optical fingerprint scanner where a user normally places the thumb. Such fingerprint scanner does not require active participation by the user and, therefore, it is not necessary to periodically perform biometric-related tasks that are not part of the normal user activities. Its approximate cost is 150 US Dollars.
For acquisition of the images for face recognition, we use the built-in camera of a laptop that can continuously capture images. Finally, we collected keyboard data using the standard PS/2 keyboard integrated in the laptop.

Software Design

In this Section, we describe our software implementation. All software we developed is implemented in Java. Client-server communication is based on RESTful web services (REpresentational State Transfer architectural, [23]), and developed using the Jersey framework [24].

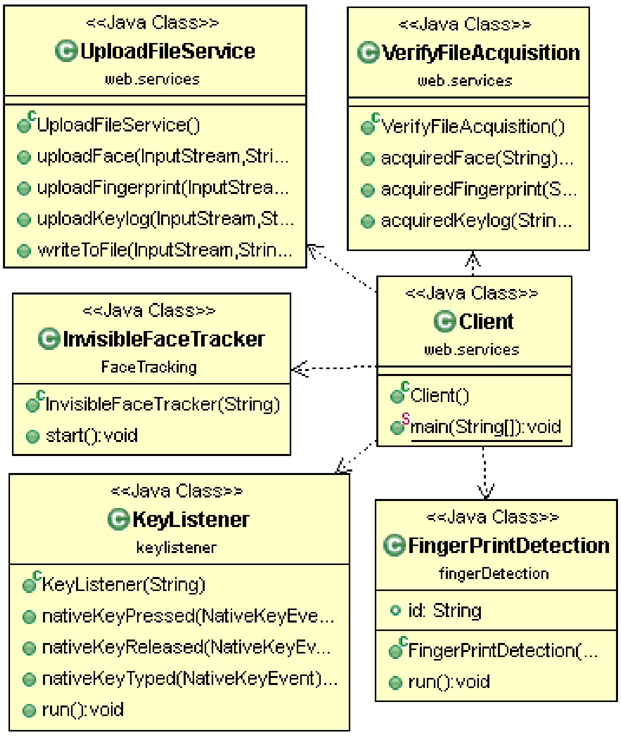


Figure 22: Class diagram of the most relevant workstation classes

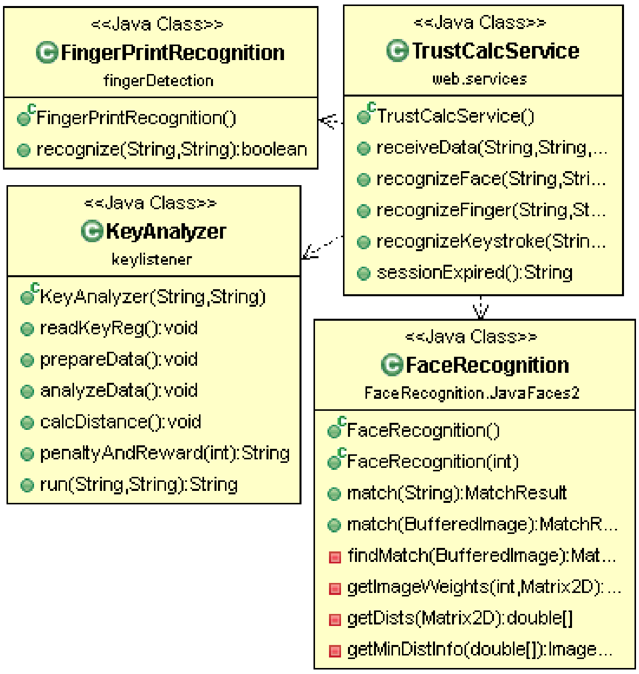
The workstation software, described in Figure 22, is started by a `Client` object, which activates the methods of: (i) `InvisibleFaceTracker`, (ii) `KeyListener` and (iii) `FingerPrintDetection`. Each class contains respectively:

(i) an algorithm that, exploiting the camera, saves an image if (at least one) face is detected,

(ii) a procedure that, exploiting the SecuGen OptiMouse Plus, cyclically detects and saves a fingerprint which, when available, is transmitted to Client,

(iii) an algorithm that detects the pressing of the keys. At fixed time intervals, a text file is saved containing, for each row, the press and release times for each pressed key. Such file is delivered to the authentication server.

The `Client` is in charge of invoking the `UploadFileService` REST client to transmit via HTTP post the saved image, fingerprint and text file to the authentication server, together with the client ID.
On the authentication server side, the `TrustCalcService`, shown in Figure 23, guides the biometric verification, the calculation of the trust level, and the communication of session expiration to the critical application or system which has to be protected, and to the client's workstation. The `TrustCalcService` class contains the RESTful web service that receives the transmitted biometric traits



<<Java Class>>
FingerPrintRecognition
fingerDetection
FingerPrintRecognition()
recognize(String,String):boolean
<<Java Class>>
TrustCalcService
web.services
TrustCalcService()
receiveData(String,String,...
recognizeFace(String,Stri...
recognizeFinger(String,St...
recognizeKeystroke(Strin...
sessionExpired():String
<<Java Class>>
KeyAnalyzer
keylistener
KeyAnalyzer(String,String)
readKeyReg():void
prepareData():void
analyzeData():void
calcDistance():void
penaltyAndReward(int):String
run(String,String):String
<<Java Class>>
FaceRecognition
FaceRecognition.JavaFaces2
FaceRecognition()
FaceRecognition(int)
match(String):MatchResult
match(BufferedImage):MatchR...
findMatch(BufferedImage):Mat...
getImageWeights(int,Matrix2D):...
getDists(Matrix2D):double[]
getMinDistInfo(double[]):Image...

Figure 23: Class diagram of the most relevant authentication server classes

from the client, and a REST client to communicate the session expiration to the workstation. Getting the information from the client, the web service decides which method should be called to start the verification process. Each subsystem produces a decision about the legitimacy of the user.
At fixed time intervals, the trust level is raised according to Table 11 or, if less than two traits are correctly verified, selecting the trust decaying function with the appropriate k value as in equation 5.2.
The objective of this implementation is a proof of concept, while we are aware that for being commercialized, the implementation should integrate recognition algorithms with much higher accuracy.

An Off The Shelf solution for fingerprint recognition

The SecuGen's FDx Software Developer Kit [9] provides low-level APIs for device initialization, fingerprint capture and matching functions. It provides support for Java and during the enrollment and the verification stages generates the templates in one format to be chosen between different alternatives available.

Customized algorithm for face recognition

We have customized the face recognition software available in [20], which is able to:

(i) analyze the frames captured via a camera,

(ii) locate a face in the frames,

(iii) verify user's identity.

When a face is present in front of the camera, the algorithm detects its presence; in most of the cases, this happens within approximately 40 ms. Otherwise, if a face is not present, the algorithm takes up to 200 ms to ultimately notify that no face is present. The implementation available in [20] requested the installation and configuration of OpenCV [25], an open source library that includes several hundreds of computer vision algorithms, and of JavaCV, the related Java interface.

The face recognition is based on the so-called *eigenface* technique [21]. Our customization of the software was necessary in order to:

(i) structure the implementation available in [20] in two client and a server sides, where the first is in charge of capturing images and deciding if a face is present, and the second of performing verification, and

(ii) make the acquisition of the biometric data transparent and automatic, removing the graphical interface and interactions of the user with the software.

An enrollment phase is necessary. During this phase, the user ID is associated to a set of face templates. The verification phase compares the selected face to the enrolled templates to produce a matching result.

The implementation of keystroke recognition

Keystroke data acquisition relies on the library JNativeHook [26] that provides keyboard listeners for Java. In particular, this library allows detecting keys press and release events and captures, in correspondence to those events, the time instant of the events. JNativeHook also permits to detect the keyboard usage (and the keys pressed), both if the user is typing in a specific text area or not: the cursor position is not relevant. This is consistent with our needs as we can capture keystroke data without being invasive for the activity of the control room operator.

Relying on such library, we realized the keystroke recognition in the `KeyAnalyzer` class, implementing the algorithm described in [22]. Such algorithm continuously collects the keystroke dynamics (the typed key and related pressing and release time) and applies a *penalty/reward* function on the dataset to measure the confidence that the user has not changed in the selected time interval. An enrollment session is required where the operator types several sentences, to create a biometric template based on the timing information for each typed key and key combination [22].

In our implementation of [22], when the keyboard is used with continuity, i.e., there is evidence that someone is currently typing, we collect keystroke

Table 12: An example of the first processing operation for keystroke data

current key	dwell time	flight time	next key
16	432	-154	65
65	133	101	76

dynamics for a defined time interval and then we transmit all values to the authentication server.

The selection of the time interval is critical because if the number of values collected is too low, verification will most likely fail: a short time interval would probably result ineffective for keystroke authentication following [22]. Moreover, a long time interval would imply a long wait before transmitting the values, thus risking that the session expires meanwhile.

We evaluated that listening for up to 10s of continuous typing was deemed sufficient to allow successful verification (we also remember that keystroke is the weakest of our biometric traits, and it is easily prone to false positives).

The KeyListener class executed on workstation side, records data in the following format:

```
KeyDown,16,Maiusc,1409386484764
KeyDown,65,A,1409386485042
KeyUp,65,A,1409386485175
KeyUp,16,Maiusc,1409386485196
KeyDown,76,L,1409386485276
KeyUp,76,L,1409386485340
```

where every row corresponds to a key pressure or release, and for every key we have the corresponding hashcode, name, and the timestamp (at microsecond) of the action.

The first processing operation of this biometric trait, performed by the authentication server and implemented in the KeyAnalyzer class, is a modification in the formatting of the obtained raw data, as in Table 12, where *dwell time* (or duration) indicates the time during which the key remains pressed, whereas *flight time* is the interval between a release and the pressure of the next key (this interval can also be negative, as in the example of Table 12, meaning that the next key has been pressed before the release of the current key).

Then, we analyze the obtained sequence to compute statistical data (average and standard deviation) of the detected measures and the number of occurrences of every single key. Starting from these data, it is possible to create a template, in case of the enrollment step (and, in order to do so, the keys with a number of occurrences lower than 50 and with a ratio between standard deviation and average of duration lower than 0.35 are excluded); otherwise, during the verification

step, comparing the values with the ones stored as a template, and producing the distance measures.
Distance measures let us compute the difference from the template, obtained with the *penalty/reward* function [22] and, once established the maximum threshold, it is possible to decide about the user identity.

Parameters Configuration

The proposed solution offers a wide set of parameters that can be tuned according to system requirements, in order to manage the trade-off between security and usability. The three subsystems possess their own parameters that can be managed. For example, if we consider the keystroke subsystem in our prototype, possible configurations are the penalty/reward function, and the time interval for the keystroke listening. More in general, an administrator responsible for the workstation security can act on the parameters that affect the trust level computation:

- The weight r and the trust in the individual subsystems $m(S_k)$ in equation 5.1.
- The decreasing function parameters k and s, in equation 5.2.
- The time interval between two consecutive verification attempts.
- The minimum threshold $\mathtt{trust}_{min}$ required by the system to maintain the user authenticated.

In our prototype, all these parameters are easily set via configuration file.

5.2.6 *Exemplary Run*

For clarity, we show in Figure 24 an exemplary run. As discussed in Section 5.2.3, the user initially performs a strong authentication, which sets $\mathtt{trust}(t_0) = 1$, indicated by square markers in the figure. The BCAS acquires biometric traits at time intervals of approximately $20s$.
In this run, we have three traits contemporaneously verified during the intervals ending at seconds 347, 1749, and 2002; this means that the trust level is raised to 1.0 three times.
Instead, when exactly two traits are recognized, the $\mathtt{trust}(t_i)$ is set to the corresponding value of Table 11. In Figure 24, round marker signals that the two recognized traits are face and fingerprint, and $\mathtt{trust}(t_i)$ is 0.98; instead, with a diamond marker we indicate that face and keystroke are recognized, and $\mathtt{trust}(t_i)$ is 0.87.
In the run of Figure 24, the situation of having fingerprint and keystroke recognized in the same interval has never arisen, thus $\mathtt{trust}(t_i)$ has never been set to 0.97.

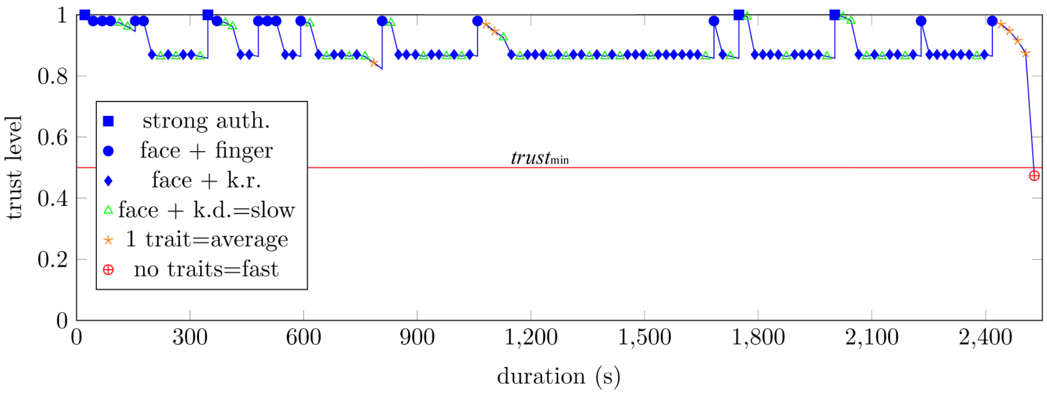


Figure 24: Example of continuous verification with the user remaining authenticated for about 42m until leaving.

When the face trait is recognized and keyboard usage is detected (but it is not possible to complete keystroke recognition), the $\mathtt{trust}(t_i)$ starts decaying slowly. This is indicated in Figure 24 by a triangle.

The decaying becomes a bit faster (average speed, star marker in Figure 24) if exactly one trait is recognized, and even faster if no traits are recognized (fast speed, target marker). In the run of Figure 24, we can see that the user remains authenticated for about 42 minutes, then at second 2441 he stops using both mouse and keyboard: only the face is recognized and the $\mathtt{trust}(t_i)$ decreases with an average speed for four intervals.

Finally, after second 2506, the user leaves the workstation and no data is recognized: as can be seen in Figure 24, the trust quickly decreases to 0.47, below the $\mathtt{trust}_{min}$ threshold of 0.5 selected for this run.

5.3 EXPERIMENTAL EVALUATION AND USABILITY-SECURITY ASSESSMENT

In order to evaluate the proposed mechanism, we assessed its usability and security trade-offs varying configuration parameters. This allows identifying the more suitable configuration for different requirements. This section describes the tests methodology and results, after the introduction of some fundamental concepts.

5.3.1 *Foundations on Usability Analysis*

The formal definition of usability by the ISO (International Organization for Standardization) is *the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use* [27].

There are important attributes characterizing usability. These are [28, 29]:

- *Effectiveness*, which answers the question: *"Can users complete tasks with the system?"* In other words, a system is effective if it behaves as expected and can be used easily. This is usually measured quantitatively with error rate, which in our context means FAR and FRR.
- *Efficiency*, which answers the question: *"How much effort is required from the users to do this?"* In other words, a system is efficient if users can accomplish goals quickly, accurately, completely and with limited resources consumption. It is usually a measure of time.
- *Satisfaction*, which answers the question: *"What do users think about the easiness of the products' use?"* It refers to the users' perceptions, feelings, and opinions about the product, their comfort and feedback about the system usage. Usually satisfaction is captured through interviews or questionnaires.

5.3.2 *Overview and Goals of the Assessment*

The best way to investigate the usability of our system is conducting a study involving real users, because it provides direct information about how they perceive the system and interact with it. Following the definition of usability by the ISO, we want to study our BCAS in terms of effectiveness, efficiency and satisfaction, taking into account also the trade-off with security. We will compare our results mainly with [30] and [31], which among the works in literature are the closest approaches. We will also compare with [10] the acceptability of biometric traits.

Effectiveness

Can users of our system complete their tasks while the continuous authentication is running? How often are they disturbed or even rejected as impostors? Are the impostors rejected if they try to intrude the system?
Effectiveness itself is in a certain way a measure of the trade-off between usability and security. We want to measure the effectiveness of our solution calculating the FAR and the FRR for the individual biometric subsystems (fingerprint, face, and keystroke biometric traits), and for the BCAS. These two metrics, in fact, are indicators of system effectiveness: lower are the error rates, more effective is the continuous authentication.

Efficiency and Resources Utilization

How long the legitimate user remains authenticated during a task execution? How fast an intrusion is detected? Are the system and the user activity slowed down by continuous authentication?

In our tests, the efficiency of user-system interaction is represented by the time that the legitimate user remains authenticated before session termination. Our goal is to assess the efficiency of the system measuring the time interval between the initial strong authentication and the unexpected session termination. We call this measure *Authentication Time (AT)*.
Similarly, we are interested in the *Time to Impostor Rejection (TIR)*, namely the time necessary for the authentication system to reject an impostor that gains possession of a workstation left unattended.

We also want to clarify if the overhead introduced by the continuous authentication system slows down the workstation and, therefore, increases the effort required to the users. For this purpose, we asked participants to complete four tasks on a workstation provided with our BCAS application running in background. One of the tasks is performed with a placebo application that resembles the real one but actually does not perform continuous authentication. Consequently, it has an insignificant overhead.
We want to measure the *Completion Time (CT)* for each task, and compare the CT difference between the same tasks completed with the real and with the placebo applications. The tasks resemble real-life work using Microsoft Office, and the participants are requested to reproduce documents using Word, Excel and PowerPoint. This choice will also permit a comparison with [30]. In addition, we want to calculate the overhead of the BCAS in terms of percentage of CPU usage.

Satisfaction

What do users think about working at a workstation with continuous authentication running in background?
The satisfaction of the participants has been measured with a Likert scaled post questionnaire [32, 33, 34], designed to gather users opinions and comments about their acceptance to provide the biometrics, and their interaction with the BCAS for both the enrollment and the continuous authentication phases.

Trade-off Security/Usability

How do changes in parameters configuration affect security and usability?
Another goal is to perform the specified measurements with different parameters configuration, such as varying the $\mathtt{trust}_{\mathtt{min}}$ threshold or the triple of k parameter values, which modifies the speed of the trust decaying function, as discussed in Section 5.2.3.
We tested two configurations of k, where its value is proportional to the decaying speed, combined with three configurations of $\mathtt{trust}_{\mathtt{min}}$. Their values are respectively shown in Table 13, and Table 14). As an example, with the first triple of k values in Table 13, the trust level decreases slower than with the second

Table 13: Configurations of k parameter

$\mathrm{Configuration}$	Type and Value
k 1)	$fast = 8 \times 10^{-3}$
	$average = 8 \times 10^{-4}$
	$slow = 5 \times 10^{-4}$
k 2)	$fast = 1 \times 10^{-2}$
	$average = 1 \times 10^{-3}$
	$slow = 8 \times 10^{-4}$

Table 14: Configurations of $trust_{min}$ parameter

$\mathrm{Configuration}$	Value	Target
$trust_{min}a)$	0.3	highly usable system
$trust_{min}b)$	0.5	trade-off security/usability
$trust_{min}c)$	0.7	highly secure system

triple.
We also want to compute the *Probability of Time to Impostor Rejection (PTIR)*, which is the probability that the *TIR* is lower than a *Window of vulnerability (W)*.

5.3.3 *Design of the Single Experiment*

During the briefing, the observer welcomes the test participant and gives a brief explanation of the purpose of the participation: testing a biometric continuous authentication system. Participants are asked to complete a set of four extremely simple tasks, designed to represent realistic tasks of an office worker.
For each participant, the entire session lasts from 1 to 2 hours, depending on the participant's speed to perform the required tasks. Figure 25 shows the work-flow of the experiment.
After a brief introduction, the enrollment phase is performed: exploiting the GUI of the BCAS, and with the help of the observer, the users register 10 facial images, their right thumb fingerprint and their keystrokes. This phase lasts approximately 17 minutes. Acquiring the trait and training the related algorithm requires approximately 1 minute for the face subsystem, and less than 1 minute for the fingerprint subsystem. The keystroke acquisition phase needs approximately 15 minutes of keyboard typing in order to acquire sufficient statistical data about key pressure and release. We decided to have an enrollment

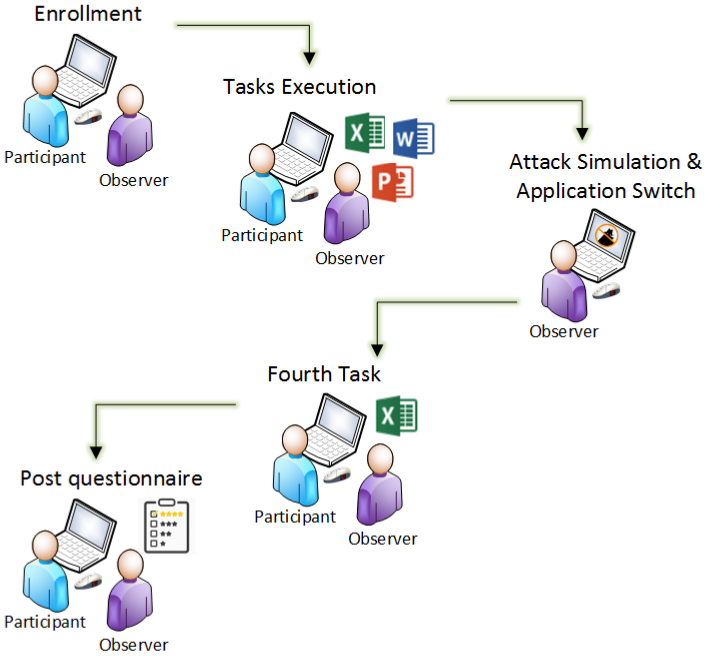


Figure 25: Workflow of the Experiment

stage of this duration because, even if it may appear long and boring to the user, having a long text to type usually increases the recognition accuracy [35]. Then, when the enrollment is completed, the observer starts the BCAS, which can either be the real or the placebo version. As in [30], the users are not informed of the presence of the placebo version of the BCAS, which does not perform any authentication and, consequently, does not introduce a significant overhead, but which has the same interface and appearance of the real BCAS. The identified tasks represent some of the ordinary operations, as realistic as possible, that users may perform in a working environment:

- *Task Word:* writing a given text document with Microsoft Word;
- *Task Excel*: producing a spreadsheet file with Microsoft Excel;
- *Task PowerPoint*: creating a presentation with Microsoft PowerPoint.

Tasks order is selected randomly before assignment. After the participants have completed the third task, they are asked to leave the workstation for a short break.

The participants who have been using the actual BCAS are asked to exit the room without logging out from the BCAS. In that time interval, and with the BCAS application running, the observer sits in front of the computer and is able to verify if the BCAS rejects him as an impostor – as it is supposed to do – and the *Time to Impostor Rejection*. The impostor looks at the screen and uses the mouse until being rejected by the BCAS. This attack scenario may look artificial, because an impostor would probably avoid contact with the fingerprint sensor if he/she needed to use the mouse. However, in our implementation – as explained in Section 5.2.3– in terms of trust level and TIR, the consequence of no fingerprint recognition is the same as presenting a non-legitimate fingerprint to the sensor. The scenario also allowed to test if the face and fingerprint recognition subsystems recognize the intrusion, or if and how many times they erroneously accept the intruder. After having performed the attack and just before the end of the short break, the observer switches the BCAS to the placebo version.

On the other hand, for the participants previously using the placebo version of the BCAS are simply asked to exit the room for a short break. In that time interval, the observer switches the placebo to the real BCAS. The attack scenario is then executed exactly in the same way as for the other group, but at the end of the whole experiment.

After the short break, all the participants are asked to complete a fourth task, which is the replication of the first task, and it is supposed to take approximately the same CT. We introduced changes to the documents in order to reduce the learning effects [30]. The changes are on the format of the documents and on their appearance, but not on their length.

5.3.4 *Participants and Experiments Plan*

Participants were students and researchers from the University of Campinas (UNICAMP), in Brazil, and the tests took place at the Institute of Computing of the same University. We spent some weeks looking for participants, sending them an invitation through mailing lists and contacting them in laboratories and classrooms. Among the 60 respondents, 65% were male (39) and 35% (21) female. The mean age of the sample was 27.72, ranging from 19 to 41 years, with a standard deviation of 4.54. Their educational level varied from undergraduate to postdoc, and their field of study was mainly computer science or engineering. Even if the participants are computer experts, the task completion did not require any particular skill except from being capable of writing documents using mouse and keyboard, and the basic knowledge of the Microsoft Office suite.

In preparation of the experiment, we divided the 60 participants in 6 groups, having 10 participants per group. Each group had assigned a combination of $\mathtt{trust}_{\mathtt{min}}$ and k parameters. The assignment of participants to groups followed the order of appearance: for example, group I had participants number 1, 7, 13, 19,..., 55. Participants were not aware of the existence of the groups, neither of the differences in parameters configuration.

We ordered the six groups (shown in Table 15) based on our expectations about system security: group I conducted the test with the most usable parameter

Table 15: Configurations of k and $trust_{min}$ for each group of participants

Table 14 Table 13	$trust_{min}a)$ 0.3	$trust_{min}b)$ 0.5	$trust_{min}c)$ 0.7
k 1)	group I	group III	group V
k 2)	group II	group IV	group VI

configuration, and group VI with the most secure one. Each user had the possibility to test both the real and the placebo version of the BCAS before or after the break. In detail, 80% of the participants (48 users, 8 per group) performed the three main tasks with the real application running, and the fourth task with the placebo version. Instead, the other 20% (12 users, 2 per group), had the placebo version running during the execution of the three tasks, and the real BCAS for the fourth repeated task.

5.3.5 *Data Collection Techniques*

The Completion Time (CT) of the repeated task was logged by the observer, using a chronometer. We used the BCAS application to track all the other data. An extract from the log file is the following:

```
2017/05/02 12:24:33,1,1,0,0,nodecay,acq,acq,not,0.98
```

Data contained in the log are, respectively:

- a timestamp of each continuous authentication iteration;
- four Boolean values representing respectively the result of face, fingerprint, and keystroke recognition, and keyboard usage detection (`1` for legitimate user, `0` when the trait is not acquired or the user is not legitimate);
- the decaying speed of trust (`fast`, `average`, `slow` or `nodecay`);
- data about the biometric traits acquisition by the server – in other words, if face, fingerprint and keystroke were acquired (`acq`) in that time window or `not`,
- the trust level $trust(t_i)$.

As an example, a 40 minutes session corresponds approximately to a log file with a length of 120 rows, where each row is generated by an iteration of BCAS: this means about one identity verification each 20s, as expected for our setup.

Table 16: Rejection and acceptance rates of the system

System FRR	System FAR	Face FRR	Fingerprint FRR	Keystroke FRR	Face FRR	Fingerprint FAR	Keystroke FAR
0.61%	3.33%	4.61%	25.20%	19.78%	3.43%	0.00%	-

We decided not to involve any additional monitoring tool, except from the BCAS itself, in order to avoid introducing any overhead that could affect the Completion Time. However, the system overhead was computed separately from the other tests using Windows Performance Analyzer (WPA), available by default on Windows.

5.3.6 *Analysis of the Collected Data and Discussion*

BCAS Effectiveness

We are now able to discuss the effectiveness of BCAS in terms of error rates. Analyzing the results of Table 16, we can see that during the tests the face recognition subsystem had a FRR of 4,61%, the fingerprint a FRR of 25,20% and the keystroke recognition subsystem showed a FRR of 19,78%. If considered individually, the FRR of our three subsystems are slightly higher than the error rates declared by other approaches in literature as [35, 36, 37]. In some of our tests, a high FRR is found, probably because the users were busy completing the tasks and did not focus on how well the biometric trait was presented; this may have caused imperfect traits acquisition and consequent errors in the recognition process.

Another important measurement taken for each test, in addition to logging the three subsystems' FRR, is *system false rejection*, which corresponds to any unexpected session termination. In other words, we have a system false rejection whenever the user trust level is below $\mathtt{trust}_{\mathtt{min}}$. As a consequence, the system FRR is obtained dividing the number of false rejections by the total number of identity verification attempts.

As shown in Table 16, the system FRR is 0,61%. This means that, despite the high FRR of the individual subsystems, our algorithm for trust level calculation properly integrates the three subsystems decisions in order to: i) reduce the rejection errors; and ii) let the legitimate user remaining authenticated.

Furthermore, if we consider the users that performed three tasks with the real BCAS (*real system group*) separately from the 12 users that completed only one with it (*placebo group*), we obtain a FRR of 0,40% and 1,44% respectively. As expected, for the placebo group the FRR is higher than for the real system group, because performing three tasks with the placebo version of the application means that they used the real BCAS for a short period. However, the results are good if compared with [30], in which FRR was 0,86% for the real system group and 3,13% for the placebo group.

Performing the attacks, we are also able to measure the individual FAR of face and fingerprint subsystems. It is the number of times that each of the traits, belonging to the impostor sitting in front of the computer, was erroneously recognized as legitimate. In order to recreate the same conditions of trust level decreasing for all participants, the substitution of the legitimate user with the attacker happens when $\mathtt{trust}(t_i)$ is 0.98. This is realized asking the legitimate user to look at the webcam and use the mouse right after the third task was completed, and before leaving the workstation unattended for a short break.
As explained previously, during the attack scenario the impostor sits in front of the workstation looking at the screen and using the mouse until session expiration, and this has been rigorously repeated for all the 60 tests in order not to alter conditions. As a consequence, we were not able to calculate the keystroke FAR during the experiments. However, we know from [22] that with the selected algorithm, the average number of keystrokes needed to lockout an intruder varies between 79 and 348, that means about 30 words.
We considered a *system false acceptance* any iteration in which the user was erroneously recognized by at least one of the subsystems, so when the trust decaying was average or slow. Results regarding acceptance rates of the system are shown in Table 16: the face subsystem has a FAR of 3,43%, and the fingerprint subsystem has an interesting FAR of 0,00%. Therefore, the system FAR is the ratio of the number of false acceptances divided by the number of identification attempts; for the BCAS it is 3,33%. We cannot compare our result with [30] because the authors did not report the FAR of their system.

BCAS Efficiency and Resources Utilization

Regarding efficiency, we first comment the results in Table 17. Analyzing the log files of each experiment, we calculate the Time to Impostor Rejection (TIR) as the time needed by the BCAS to determine the instant of session expiration from the substitution of the legitimate user. In other words, it is a measure of the time necessary to decrease the trust level under the $\mathtt{trust}_{min}$ threshold.
As expected, the MTIR is proportional to the threshold (see Table 17). For instance, the group I of participants, which has the lowest threshold of 0.3 and the most usable k parameter configuration, shows a MTIR of 2 minutes. Conversely, the MTIR of group VI is 1 minute and 18 seconds, because of the most secure configuration of k parameter, and the highest $\mathtt{trust}_{min}$ of 0.7 designed for this group of users.

However, the MTIR proportionality, and more in general all the analysis, applies as long as our system is not modified. With other implementation

Table 17: System efficiency measures

	Mean Time to Impostor Rejection (MTIR) (mm.ss)	Mean Authentication Time (MAT)
group I	02.00 ± 0.32	100.00%
group II	01.55 ± 0.25	99.36%
group III	01.48 ± 0.49	97.96%
group IV	01.47 ± 0.47	96.09%
group V	01.28 ± 0.28	99.85%
group VI	01.18 ± 0.37	98.09%

choices, as for instance a trust computation which distinguish between a missing trait and a not legitimate one, we may have different results in terms of usability and security measures.

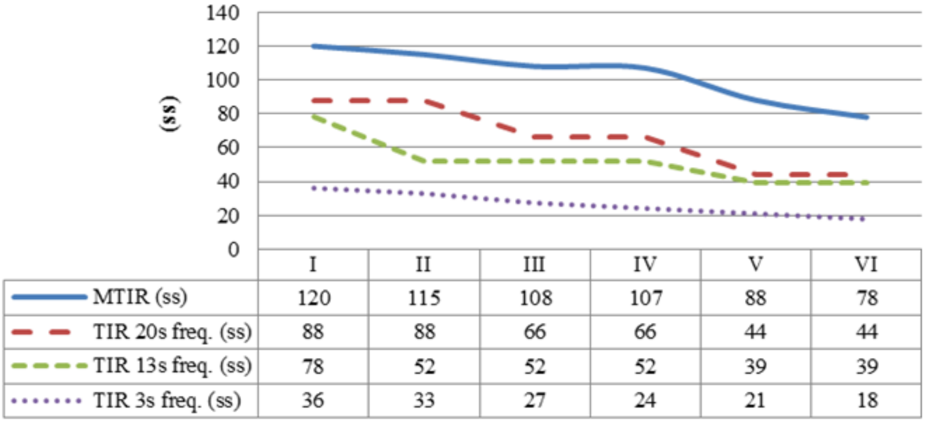
Generally speaking, the TIR is influenced by the false acceptances of the recognition subsystems: between the tests that did not show false acceptances, the lowest TIR is 44s. This measure corresponds to the time needed by the BCAS to close the session if the workstation was left unattended with the configuration of group VI.

Regarding the number of unexpected session terminations, we have that 75% of the participants were able to complete the tasks without any session termination, so the BCAS execution was actually transparent to them. We can considered it satisfying, especially if compared with the result of [30], where the percentage of completion without logout was 48%. It is also interesting to observe that the most usable parameters configuration let 100% of group I users to complete the tasks without interruptions.

We also report in Figure 26 the simulation of the lowest TIRs of the six groups, in correspondence to different intervals $[t_{i-1}, t_i]$. In other words, this shows the time needed to reject an impostor if we increase the frequency of continuous authentication iterations. The intervals varies from about 20 s (as in the tests with real users), to 3 s (referred as "3s Freq." in the Figure). Repeating the continuous authentication (maintenance phase) every 3s, and having the parameters set as in configuration of group VI, would make the impostor to be rejected in only 18 s.

In Figure 27 we show how many tests would have been completed without session expiration with the corresponding trust_{min} value varying between 0.9 and 0.3.

The Authentication Time (AT) is calculated as a fraction of the total time that the user remains authenticated [31]. For instance, suppose that the total time



	I	II	III	IV	V	VI
MTIR (ss)	120	115	108	107	88	78
TIR 20s freq. (ss)	88	88	66	66	44	44
TIR 13s freq. (ss)	78	52	52	52	39	39
TIR 3s freq. (ss)	36	33	27	24	21	18

Figure 26: MTIR and expected TIR of the six groups

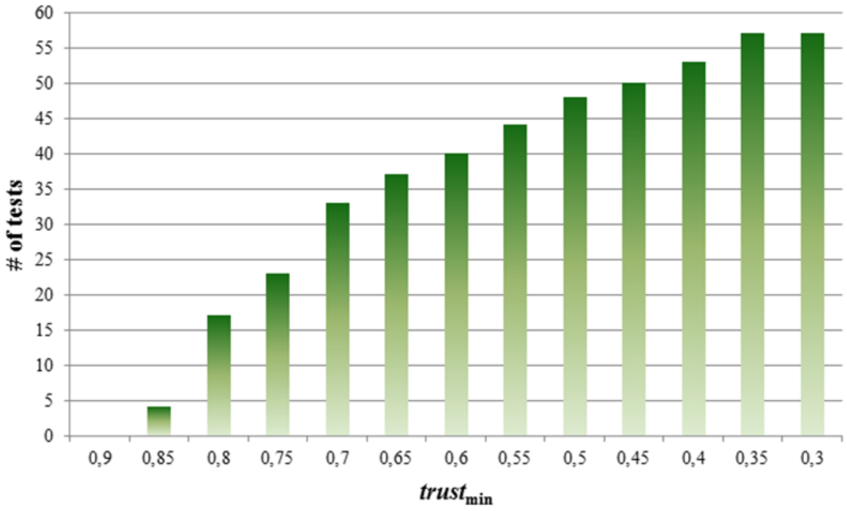


Figure 27: Expected number of tests without expirations

needed for a user to complete the tasks with the real BCAS running is T seconds and, during this time, the system rejects the user once or more times, preventing him/her accessing the protected resources a seconds. Thus, the AT is calculated as *(T-a)/T*.

As shown in Table 17, the Mean Authentication Time (MAT) of group I was 100% because they did not have any unexpected expiration. We are not able to formally compare our results in term of MAT with [31], because of the differences in terms of tasks, length of the experiment and number of participants. However, our MAT seems to be very similar to the one obtained in [31].

In order to determine whether our system had any significant effect on Completion Time (CT), we executed a paired t-test on the difference between the tasks CT with the BCAS, versus the CT of the placebo version. It is generally

Table 18: Paired t-test results for task completion time

Task	N	Paired differences: BCAS- Placebo version					t	df	Sig-2 tailed
		Mean Time	Std. Dev	Std.Err. Mean	95% Confidence Interval of the Difference				
					Lower	Higher			
Word	20	32.80	5.34	1.20	-7.40	73.00	1.7076	19	0.1040
P.Point	20	6.95	61.43	13.74	-60.42	74.32	0.2159	19	0.8314
Excel	20	42.75	-37.62	-8.41	-104.1	18.51	1.4605	19	0.1605

used to compare two population means to test the null hypothesis that the true mean difference is zero.

The combined paired t-test results are in Table 18.
N is the number of tests executed with the BCAS and then repeated with the placebo version, and it is 20 for each task.
The *Mean Time* is the mean difference between tests with BCAS and tests with the placebo version, *Std. Dev* is the standard deviation of the differences, and *Std.Err.Mean* is the standard error of the mean difference. Table 18 also shows the 95% confidence interval of their difference.
Under the null hypothesis, the t-statistic follows a t-distribution with n-1=19 degrees of freedom (indicated with *df* in the table). Comparing the values obtained for t with the t19 distribution, we obtain the p-value for the test. The result is that at the p< .01 level:

- There was no significant difference between time taken to complete the Word task with the BCAS and the placebo version $(p = .1040)[t_{19} = 1.7076, p > .05])$.
- There was no significant difference between time taken to complete the PowerPoint task with the BCAS and the placebo version $(p = .8314)[t_{19} = .2159, p > .05])$.
- There was no significant difference between time taken to complete the Excel task with the BCAS and the placebo version $(p = .1605)[t_{19} = 1.4605, p > .05])$.

We can conclude that there is no significant difference in completion time for all tasks. This gives evidence that there is not any significant impact on task performance, which is the same result obtained by the authors of [30].

As explained previously, in order to avoid introducing any additional overhead that could affect the Completion Time, the system overhead has been computed as a separate test and without the involvement of participants. The observer executed the PowerPoint task twice, with the real BCAS running and then

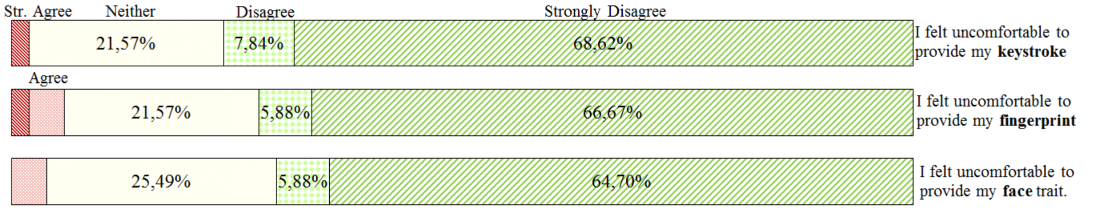


Figure 28: Questionnaire results about acceptability

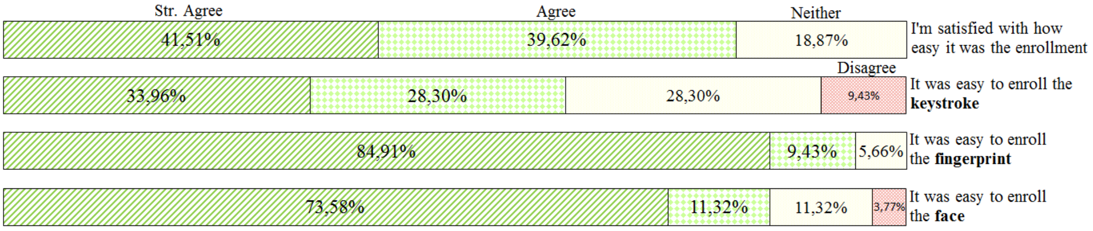


Figure 29: Questionnaire results about enrollment

repeating it with the placebo version, obtaining a comparative analysis of CPU usage with Windows Performance Analyzer ®(WPA). The resulting overhead for our machine, in terms of CPU usage, is 2,06%. This result is promising if compared with [31] and [30], who declared an overhead of 25% and 42% respectively, and is also an indication that nowadays, thanks to the technological progress of the last decade, biometric continuous authentication be actually integrated without slowing down the operating system.

User Satisfaction

Analyzing Figure 28, we can discuss to what extent the participants are willing to provide each of the biometric traits in order to perform the enrollment. It is interesting to compare our results with [10], in which the authors perceived keystroke, fingerprint and face characteristics having respectively medium, medium and high acceptability, and the same result has been reported in our Table **??**.

The result of the test is that we have the highest acceptability for the keystroke trait; in fact 80% of users said that they did not felt uncomfortable in providing it.

We find a high acceptability also for fingerprint (76,67%), and 70% of the participants felt comfortable in providing their face.

We report in Fig. 29 the users' opinion about the enrollment. Generally speaking, 83,33% of them were satisfied with its easiness. Between the three traits, as expected they felt more uneasy with the keystroke acquisition (11,67%), probably because of the 15 minutes length of the process.

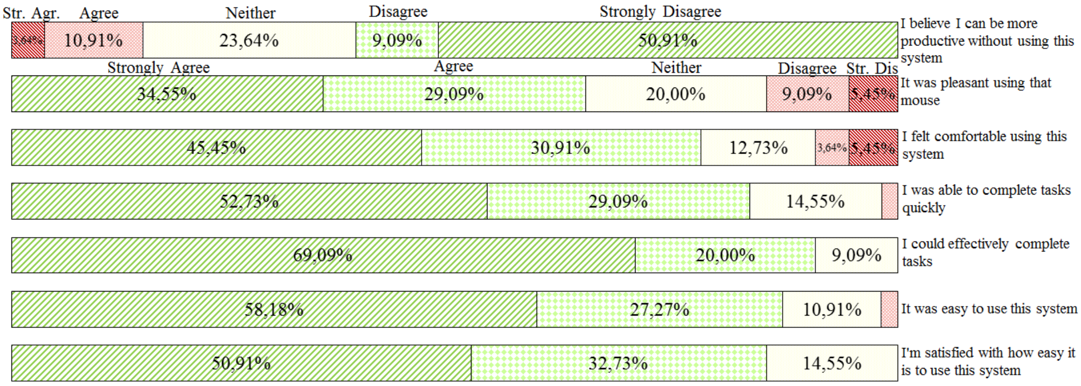


Figure 30: Questionnaire results about usability of the system

The results about users' satisfaction regarding system usability are shown in Figure 30. A consistent amount of participants (28,33%) found the keyboard unpleasant: the notebook used for the tests had a column of special keys on the left that the users often pressed unintentionally. Also the mouse, with the optical sensor for the right thumb fingerprint acquisition, was not pleasant to use for 20% of the users. These two elements probably influenced the perception on the system's usability; still it was comfortable for 78,34% of them. 13,33% of them believed they could be more productive without this system, probably because of the higher comfort and the familiarity they have with their own system, and also because they were forced to change their usual interaction with mouse and keyboard (10%).

Nevertheless, the participants found the system easy to use (88,33%) and were satisfied with it (86,66%). They also said to be able to complete tasks effectively (91,66%) and quickly (86,33%), and this was one of our main objectives thinking about our system's usability. A proper comparison with [30] is impossible, because the authors did not report the complete results of their questionnaire nor all the questions; we only know that their users were satisfied with system's responsiveness, with the overall system, and with the comfort they felt.

The Trade-off Between Security and Usability

In order to analyze the trade-off between usability and security, we follow the approach of [31]. As discussed in Section 5.3.2, we call Probability of Time to Impostor Rejection (PTIR) the probability that the TIR is lower than a vulnerability window (W). Vulnerability windows can be seen as the minimum time frames needed by an impostor to damage the critical system. In the ideal situation, the PTIR is 1, meaning that the impostor is certainly rejected.

In Figure 31 we report the PTIR of the six groups of users, for different values of vulnerability window. The higher the PTIR, the higher is the security provided. As we can see, configuration VI is the most secure for almost all the windows of

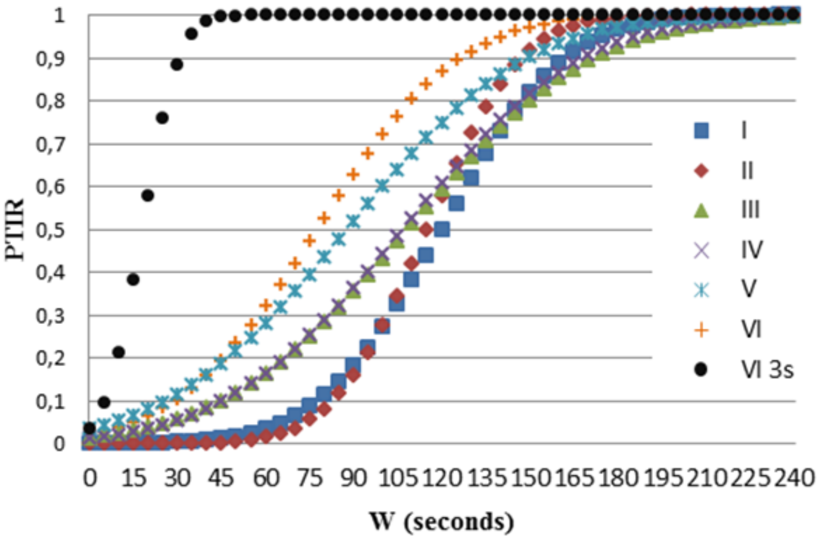


Figure 31: Plot of PTIR versus W for the six groups of users

vulnerability, and this confirms our expectations. As discussed, we designed our BCAS to execute authentication iterations every 20s (plus a delay of 2-3 seconds basically for acquiring the fingerprint). For this reason, comparison with [31] is not straightforward: their system acquires 10 frames per second and requires less than 3s to reject an impostor. With our configuration, after 150s we can see that for all the curves the reached PTIR is between 0.82 and 0.97.

However, as discussed before and shown with the simulation of Figure 26, increasing the frequency of user verification can easily reduce the TIR. When the interval $[t_{i-1}, t_i]$ is set to 3s, the TIR decreases to 18s.
The resulting PTIR for configuration VI is shown in Fig. 31 (called "VI 3s" in the legend). It is obtained synthetically and just for comparison, but clearly demonstrates that with this configuration an impostor is rejected after 25s with a probability of 76%, and the curve tends to 1 after about 30s.

Another interesting evaluation can be done analyzing the trade-off between usability and security with the same approach of [31]. We can discuss the PTIR and the MAT, being respectively a measure of system's security and usability. A PTIR of 1 is obtained when the impostor is always rejected, and this, for a low vulnerability window, means a MAT close to 0%. If instead we have a high MAT, it should be more unlikely to have the impostor rejected, especially with a low W.

The results show that the decreasing of usability was always very low if compared with [31]. In fact, the MAT of BCAS varies from 96% to 100%. However, even if group I had the best results in terms of MAT, for the six configurations of BCAS we tested, a higher security did not correspond linearly to lower usability. This anomaly is probably due to the influence of FRR and FAR. As explained in

Section 5.2.4, in our prototype the trust computation is only influenced by the number of successful verifications. A different implementation choice, which may distinguish between a missing trait and a not legitimate trait, would have direct impact on TIR, AT and on the tradeoff between usability and security in general. In fact, we can easily imagine that it would probably reduce the TIR. On the other side, it would be interesting to study if and to which extent the modification causes any side effect, for instance in terms of FRR, AT and user satisfaction.

Part III

TOWARDS CONTINUOUS NON-REPUDIATION

6 NON-REPUDIATION OF REMOTE SERVICES

As we have introduced in Section 2.5, non-repudiation is an important security service: it provides evidences of actions, protects against their denial, and helps solving disputes between parties. In this Chapter, we first review biometrics-based approaches for the point of view of the provision of non-repudiability property (Section 6.1). Then, in Section 6.2, and Section 6.3 we propose two solutions for non-repudiation of critical remote services based on multi-biometric continuous authentication. Finally, in Section 6.4 we present an application scenario that discusses how users and service providers are protected with such solutions. We also discuss, in the same Section, the technological readiness of biometrics for being employed in non-repudiation services.

6.1 NON-REPUDIATION THROUGH BIOMETRICS: RELATED WORKS

This Section presents an analysis of non-repudiation through biometrics, showing that in the literature it is an open question. For biometric authentication, as for all the other authentication mechanisms, non-repudiation depends on [89]:

- the ability of the selected traits to discriminate between individuals;
- the strength of binding between the trait and the individual in question;
- technical and procedural vulnerabilities that could undermine the intrinsic strength of the binding.

The discrimination capabilities of biometrics depend on the technology used and on other application-related factors, which are quantified in terms of error rates (FAR and FRR). According to [6], unlike passwords and tokens, biometrics – because of its strong binding to a specific person – is the only authentication factor capable of guaranteeing that authentication cannot subsequently be denied by a user. Despite biometric traits are sometimes presented in the computer security literature as an authentication factor that may solve the repudiation problem [6, 89], other works like [90, 91] draw a completely different conclusion: according to their authors, biometrics is not a security mechanism able to provide non-repudiation. Analyzing the state of the art, we can state that answers to this research question are contradictory. However, the situation changes [92, 93, 94] if biometric authentication is coupled with another security mechanism like digital signature, which is commonly considered as the standard approach to achieve non-repudiation [56]. In fact, public key infrastructure, or PKI, and biometrics can well complement each other in many security applications, giving birth to biometric cryptosystems [91, 87], and to the so-called biometric signature [93, 94]. Biometric signature is defined as the process of deriving a private key from a biometric trait and using the private key to sign an e-document. According to [92], this eliminates the problem of vulnerability of private key storage, which resolves the key management issue. The dynamically generated private key lets the user to perform signatures without carrying a disk or smart card [92]. One example is [94], in which the authors propose a fingerprint based signature scheme that uses a biometric trait to generate a key string, and then exploits the string to create a public key and the corresponding private key.

The analysis of the state of the art shows that no solutions capable of providing multi-biometric continuous non-repudiation of remote services exist. To our knowledge, this work is the first in this field. It is based on:

- *Multi-biometrics*, which as discussed in Section 2.4, solves the problem of authentication factor loss or steal. It can also avoid authenticating an impostor in case a single biometric trait is forged or compromised, and yield a higher performance than using only one trait.

- *Continuous authentication*, to guarantee the actual presence of the user of the system/device. Noteworthy, we exploit authentication as a requirement for non-repudiation: our solution binds the generated evidence of an action to the user identity only if identity verification is successful.

- *Digital signature* (for both the solutions we propose in this Chapter) and biometric signature (for the BS-CNR), continuously applied for the entire session duration in which a user remotely accesses an Internet service.

6.2 DS-CNR: DIGITAL SIGNATURE FOR CONTINUOUS NON-REPUDIATION

In order to continuously provide non-repudiation, our idea is to introduce digital signature in the biometric continuous authentication system designed in [19]. We believe that biometrics and digital signature are two security mechanisms that well complement each other. In fact, biometric continuous authentication contributes to strengthen authentication. Vice versa, as discussed in the previous Section, it is not so clear if, and to which extent, biometric authentication provides non-repudiation if employed alone. As a remark, the qualitative risk assessment of CASHMA system conducted in Section 5.1, highlighted that the introduction of digital signature is an effective countermeasure to several threats – as spoofing, forgery, or message corruption – not only for repudiation.

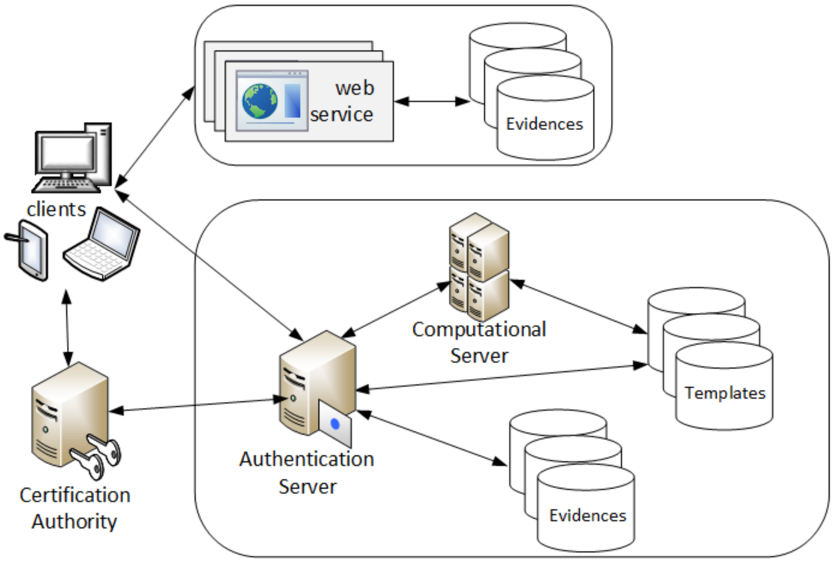


Figure 32: Overall view of the DS-CNR architecture.

6.2.1 *The Architecture*

The overall system, shown in Figure 32, is obtained from CASHMA [19] adding a Certification Authority (CA), as featured by many PKI schemes. The other main entities that compose the system are: an Authentication Server, the clients and the web services. All of them are connected through TLS[1]. The CA provides digital certificates in order to certify the ownership of a public key by each client and to guarantee that the client has sole control and access to the corresponding private key. We denote the couple of cryptographic keys as K_c^- and K_c^+, being the client's private and public key respectively, certified by the CA. We also assume that the Authentication Server has its own couple of keys: K_{AS}^- and K_{AS}^+.

As in [19], the Authentication Server is in charge of transmitting a certificate, called *DS-CNR certificate*, to the client. As for the CASHMA certificate in [19], the DS-CNR certificate is composed of:

```
Time stamp, sequence number, ID, Decision, expiration time.
```

The `Decision`, which represents the outcome of the verification procedure carried out on the server side and the user is not considered legitimate if the trust level is below the $\mathtt{trust}_{min}$ threshold. The higher is $\mathtt{trust}_{min}$, the higher are the security requirements of a specific application or web service. Note that, differently from [19], $\mathtt{trust}_{min}$ varies during the same session depending on the criticality of the operation that the client is asking to perform on a specific moment.

[1] SSL is now deprecated, thus from here we refer to its successor: the Transport Layer Security protocol

Further, we introduce two databases of evidences: one is for the storage of the messages received by the Authentication Server and the related signatures, while the other contains the digital signatures of all the requests and DS-CNR certificates received by the web service.

The other elements in Figure 32 are: high-performing computational servers that perform comparisons of biometric data for verification of the enrolled users, and databases of templates containing the biometric templates of the enrolled users.

6.2.2 *The Protocol*

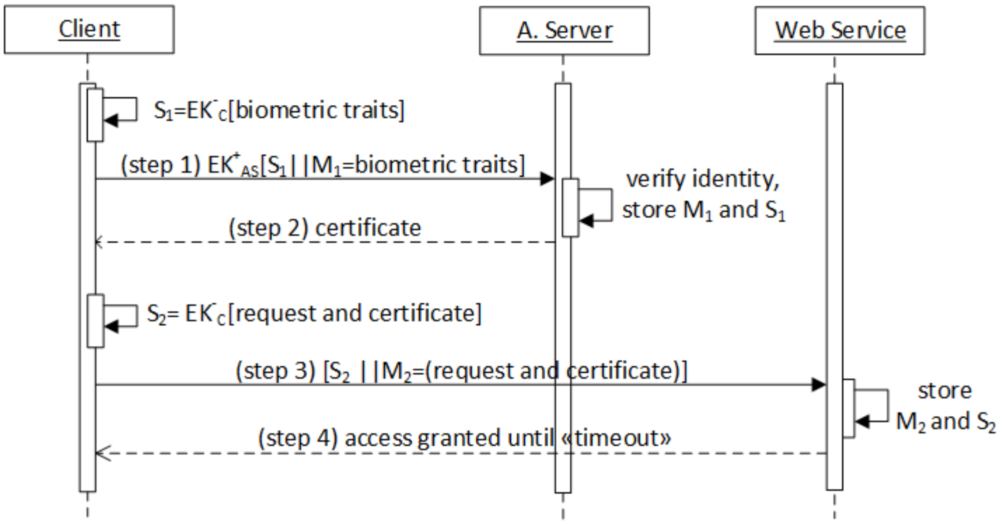

Client
A. Server
Web Service
S_1=EK^-_c[biometric traits]
(step 1) EK^+_{AS}[S_1||M_1=biometric traits]
verify identity,
store M_1 and S_1
(step 2) certificate
S_2= EK^-_c[request and certificate]
(step 3) [S_2||M_2=(request and certificate)]
store
M_2 and S_2
(step 4) access granted until «timeout»

Figure 33: Initial Phase of DS-CNR protocol in case of successful identity verification.

During the enrollment, the client provides the selected biometric traits that are stored by the Authentication Server. We propose a continuous non-repudiation protocol (shown in Figure 33) integrating the digital signature in the existing CASHMA protocol. In detail, two different signatures are used during Steps 1 and 3 (and during the maintenance phase on Steps 5 and 7). The following description highlights the differences between the new version of the protocol and the original one, discussed in Section 4.1.2.

Initial phase

This phase, shown in Figure 33, is now structured as follows:

First of all, exploiting the client's private key K_c^-, the application on client side performs the signature S_1 of the message containing the biometric traits of the user.

Step 1 - Message M_1, with the biometric traits, is sent at this step concatenated with its signature S_1. The resulting message is then sent after adding a further encryption layer obtained using the Authentication Server public key K^+_{AS}.

Step 2 - The Authentication Server is the only entity able to decrypt the message, using its private key K^-_{AS}. Then it analyzes the biometric traits received and performs the authentication procedure as usual. As in [19], the user is not considered legitimate if the trust level is below the trust threshold $\mathtt{trust}_{min}$. In the DS-CNR protocol, depending on the criticality of the action to be performed, the $\mathtt{trust}_{min}$ threshold may vary during the same session, and the Decision is directly influenced. If the user identity is successfully verified, the server authenticates the user, computes an initial timeout of length T_0 for the user session, sets the expiration time at $T_0 + t_0$, creates the DS-CNR certificate and sends it to the client. The Authentication Server also stores the received message M_1 and its signature S_1 in its database of evidences: these data can be accessed to solve possible disagreements between client and web service providers. In this way, the Authentication Server can act as a Trusted Third Party.

At the end of Step 2, exploiting its own private key K^-_c, the client performs the signature S_2 of the message M_2, containing request and the DS-CNR certificate.

Step 3 - The client forwards M_2 to the web service concatenating it with its signature S_2, which constitutes the NRO token. In fact, the web service stores M_2 and S_2 (the NRO token) in its database of evidences. This solves disagreements; in particular, it protects the web service providers against repudiation of origin.

Step 4 - As in [19], the web service reads the certificate and authorizes the client to use the requested service until expiration time. The client stores the message received on Step 4, which constitutes a NRD token and protects it from disagreements with the web service.

Maintenance phase

A maintenance phase is then started. It is composed of four steps (Steps 5-8), analogous to Steps 1-4 of Figure 33. This phase is repeated iteratively: the client sends fresh biometric traits (Step 5), the server repeats the identity verification, takes the related Decision, renews the certificate and saves the evidences (Step 6); the client sends the new request and certificate to the web service, in order to have the timeout expiration postponed by the web service (Step 7); the web service stores the NRO token and then sends back the *"access granted until timeout"* confirmation, which is saved by the client and constitutes the NRD evidence (Step 8).

6.2.3 *Security Considerations*

As discussed in Section 4.1.2, we assume that all data exchanged with this protocol are transmitted using TLS, which has been designed to provide privacy and data integrity between communicating parties [86]. However, TLS cannot protect against the insecurities introduced into the client system: if the user leaves his/her computer logged in to a secure web service and someone gets possession of the client computer without permission, the TLS protocol cannot contrast the intrusion. For this reason, the continuous authentication protocol is necessary to guarantee the authenticity of the user.

In addition, even if TLS – exploiting a MAC – can guarantee that a message has not been changed during its transmission, it cannot provide non-repudiation. In our protocol, this is obtained with the integration of digital signatures in the communications where sensitive data is exposed.

In other words, the protocol is now able to continuously guarantee user authenticity (thanks to multi-biometric authentication), privacy and data integrity (thanks to the TLS-based communications) and non-repudiation (thanks to digital signature) of the exchanged messages.

Non-repudiation is obtained digitally signing the messages from client to Authentication Server (Steps 2 and 6), and the messages from client to web service (Steps 3 and 7) and back (Steps 4 and 8). With this solution, if a client tries to deny having accessed a web service, a judge can ask the web service for the client's message M_2 – that contains request and certificate – and the related signature S_2; then retrieve the client public key K_c^+ from the Certification Authority and compare the signed message with the original one to verify if the signature is valid or not. In other words, consulting the evidence (NRO token) the judge establishes if the client is lying or not. Similarly, the NRD token and the help of the Authentication Server (which has M1 and S1 stored) are the evidences that can protect the client from possible disputes with the web server.

In general, in a direct digital signature scheme, which means without the participation of a TTP, non-repudiation depends on the security of the sender's private key. If a sender wishes to deny the access, the sender can claim that the private key was lost or stolen and that someone else forged his/her signature [95]. The solution we proposed in this Section tackles this vulnerability but still does not eliminate it completely. In fact, in DS-CNR the digital signature only works as long as the private key K_c^- remains secret. However, if the key is disclosed (or if the client discloses it), an attacker can produce the signature S_1 only if it also possesses the biometric traits of the user. Similarly, the signature S_2 can be forged only if the attacker possesses a valid DS-CNR certificate, which contains a new sequence number and in which the *expiration time* has not been reached yet.

With the DS-CNR solution, the problem of usability of continuous non-repudiation is solved. In fact, even if the client has its private key stored on an external device, it does not need to provide a password or secret for each single transaction. It is sufficient to insert the secret only at the first login, without losing security: thanks to the multi-biometric identity verification, a secure authentication is guaranteed for the whole session. If, instead, the private key is stored on the client main computer/device, the multi-biometric continuous authentication will guarantee protection against insiders. In fact, even if an attacker succeeds in violating the client's computer or device, it still has to continuously send multiple biometric traits to the Authentication Server if desires to access the web service and perform the operations that it offers.

In other words, without the multi-biometric authentication, the continuous non-repudiation would not be possible at the same degrees of usability and security.

6.3 BS-CNR: BIOMETRIC SIGNATURE FOR CONTINUOUS NON-REPUDIATION

To address the problem of private key loss [95], we present an alternative solution based on biometric signature.

In the literature largely investigated [91, 92, 87, 93, 94], biometric signature is a process of deriving a private key from a biometric sample and using the key to digitally sign a document or a message. We exploit biometric signature to prevent client's claim of having lost the private key or that the key has been stolen. The idea is to generate a new, additional, private key, denoted as $\mathtt{BioK}_c^-$, during the first iteration (Initial Phase) of the protocol. Analogously, the corresponding public key $\mathtt{BioK}_c^+$ should be generated concurrently.

The advantage of our solution is that the client does not store the private key when the session terminates, because it is valid for only one session and, assuming that the key is securely transmitted through TLS, the client cannot claim that it has been lost or stolen. In addition, the <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> key pair is generated only if the user identity is verified (the trust level $\mathtt{trust}(u,t)$ is above the $\mathtt{trust}_{min}$ threshold).

6.3.1 *The Architecture*

The overall architecture of the BS-CNR system is shown in Figure 34. The main difference with DS-CNR is that for BS-CNR the Authentication Server is also responsible for the generation of a couple of biometric-derived keys <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> for each client during the initial phase of the protocol.

The remainder of the architecture does not change from what has been discussed in Section 6.2.1. As in DS-CNR, an external Certification Authority is involved, which provides digital certificates in order to certify the ownership of the couple of cryptographic keys K_c^- and K_c^+ by the client. We also assume that the Authentication Server has its own couple of keys K_{AS}^- and K_{AS}^+. The other entities are: an Authentication Server, the clients and the web services. All of them are connected through TLS. Again, the other elements in Figure 34 are a set of high-performing computational servers for the biometric traits com-

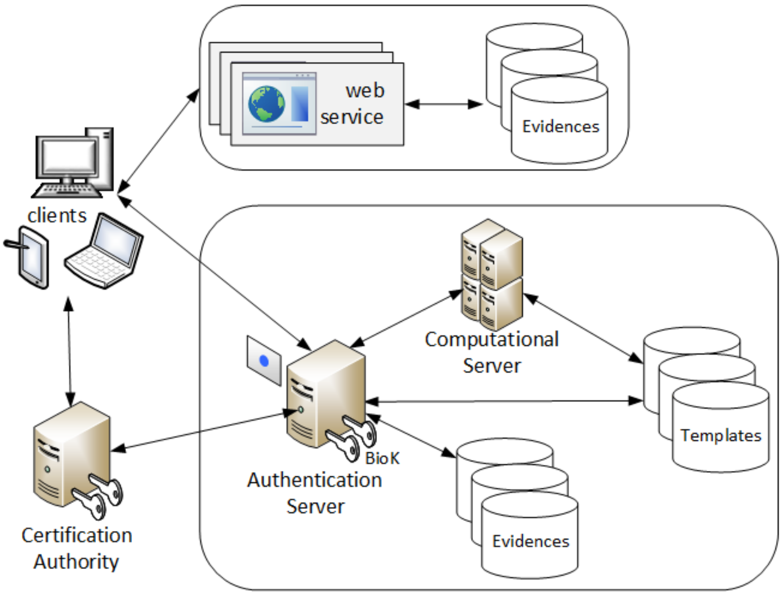


Figure 34: Overall view of the BS-CNR architecture.

parison useful for verification of the enrolled users, and databases of templates containing the biometric templates of the enrolled users.

As in the solution of Section 6.2.1, the BS-CNR certificate is composed of:

`Time stamp, sequence number, ID, Decision, expiration time.`

Finally, the considerations of Section 6.2.1 about the `Decision` are still valid here: $\mathtt{trust}_{\mathtt{min}}$ varies during the same session depending on the criticality of the operation that the client is asking to perform on the web service at a specific moment.

6.3.2 *The Protocol*

During the registration phase, the client provides the selected biometric traits that are stored by the Authentication Server.

In this protocol, the Authentication Server exploits the biometric traits not only for user identity verification, but also to generate a couple of cryptographic keys $<\mathtt{BioK}_c^+, \mathtt{BioK}_c^->$, following one of the existing approaches, e.g., from [92, 93].

Initial phase

This phase, shown in Figure 35, is now structured as follows:

First of all, exploiting the client's private key K_c^-, the application at the client side performs the signature S_1 of the message containing the biometric traits of the user.

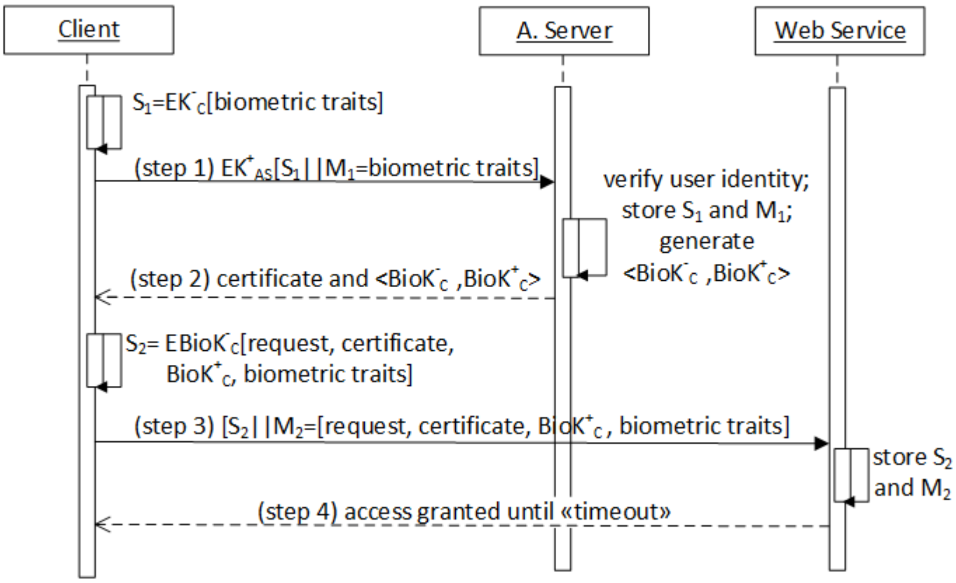

Client
A. Server
Web Service
S1=EK-c[biometric traits]
(step 1) EK+AS[S1||M1=biometric traits]
verify user identity;
store S1 and M1;
generate
<BioK-c ,BioK+c>
(step 2) certificate and <BioK-c ,BioK+c>
S2= EBioK-c[request, certificate,
BioK+c, biometric traits]
(step 3) [S2||M2=[request, certificate, BioK+c, biometric traits]
store S2
and M2
(step 4) access granted until «timeout»

Figure 35: Initial Phase of BS-CNR protocol in case of successful identity verification.

Step 1 - Message M_1, with the biometric traits, is sent at this step concatenated with its signature S_1. The resulting message is then sent after adding a further encryption layer obtained using the Authentication Server public key K^{+}_{AS}.

Step 2 - The Authentication Server decrypts M_1 with its private key K^{-}_{AS}. Then, it analyzes the biometric traits received and performs the authentication procedure as usual. As in DS-CNR, depending on the criticality of the action to be performed, the security requirements can demand a trust threshold $trust_min$ set to significantly high values. So, during the same session, the threshold varies based on the operation that the client is going to accomplish, and the Decision is directly influenced by that.

If the criteria are not completely satisfied (the message in incomplete or the trust is below the threshold), the user is forced to return to Step 1.

Instead, if the user identity is successfully verified, the server authenticates the user, computes an initial timeout of length T_0, sets the expiration time at $T_0 + t_0$, and creates the BS-CNR certificate. Then, the Authentication Server derives the $<BioK^{+}_{c}, BioK^{-}_{c}>$ couple from the biometric traits. Together with M1 and S1, it also stores the key pair in its database of evidences: these data can be accessed to solve possible disagreements between clients and web service providers. In this way, the Authentication Server can act as a Trusted Third Party.

The Authentication Server sends to the client a message containing the BS-CNR certificate, and also the couple of biometric keys <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> assigned to the user for this session.

At the end of Step 2, the client, using its new private key $\mathtt{BioK}_c^-$, computes $S2$. It is the digital signature of message $M2$, which is composed of: the request, the BS-CNR certificate, the public key $\mathtt{BioK}_c^+$, and the biometric traits.

Step 3 - The client forwards $M2$ to the web service concatenating it with the signature $S2$ which constitutes the NRO token. The web service reads the BS-CNR certificate and verifies its validity: it also checks if the $\mathtt{BioK}_c^+$ has been already used before the current session. The web service stores $M2$ and $S2$ (the NRO token) in its database of evidences. This may be used to solve disagreements, in particular protects the web service providers against repudiation of origin.

Step 4 - As in [19] and in DS-CNR, the web service reads the certificate and authorizes the client to use the requested service until expiration time. On its side, the client stores the message received on Step 4, which constitutes a NRD token and protects it from disagreements with the web service.

As in DS-CNR, after the initial phase there is a Maintenance Phase composed of four steps (Steps 5-8), analogous to Steps 1-4 of Figure 35. This phase is repeated iteratively: the client sends fresh biometric traits (Step 5), the server repeats the identity verification, takes the related Decision, renews the certificate and stores the evidences (Step 6); the client sends the new request and certificate to the web service, in order to have the timeout expiration postponed by the web service (Step 7); the Web service stores the NRO token and then sends back the *"access granted until timeout"* confirmation, which is saved by the client and constitutes the NRD (Step 8).

Note that the <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> couple is generated (and stored) only during Steps 1-2 of the Initial Phase: these operations are not repeated during the Maintenance Phase.

6.3.3 *Security Considerations*

This solution addresses the problem of continuous non-repudiation trying to reduce the weaknesses showed by the DS-CNR, including the potential claim of private key loss by the client. This solution introduces a strong binding between the private key and the client user: $\mathtt{BioK}_c^-$ is generated in case of successful identity verification only through the biometric traits.

In addition, the biometric-based keys can be considered as one-time keys: they remain valid only for one session. However, after the expiration of the <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> keys, the web service or a judge can still use the $\mathtt{BioK}_c^+$ to compare the message $M2$ and its signature $S2$, thus guaranteeing non-repudiation of the access. What cannot be done with these keys is using a private key for more than one session: in this way the client cannot repudiate having accessed the web service. In fact, if a couple of keys <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> is used for more than one session to send a request to the web service (Step 3), the web service would receive a message M_2 with a $\mathtt{BioK}_c^+$ key that is already stored, and would deny the access.

Moreover, in this protocol we assume that the transmissions are protected by TLS protocol, so that the privacy and the integrity of the information exchanged are guaranteed. For this solution, privacy is even more important because the message M_2 contains also the biometric traits of the user. This choice has been made to give even more strength to the NRO token stored by the web service. However, to protect client's biometrics, their templates are not sent to the web service in raw format: they are transformed using client specific parameters [96].

Apart from security, we also consider that the computational and storage resources needed are higher with respect to the previous solution: the Authentication Server generates a couple of biometric-based keys for each client, once for each session. Further, it stores the <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> couple with the corresponding certificates in case a TTP is requested to solve future disputes.

As in CASHMA, for the two versions of the protocol described here, and in the solution in the following of this dissertation, the certificate is issued based on the decision from multi-biometric recognition. Thus, our approach does not include a certificate automatically generated as a result of the comparison, as in some relatively recent schemes [155]. However, the benefits of biometric-generated keys are actually included in the BS-CNR protocol, as well as in the solution of next Chapter, and this happens in the same step of certificate generation. In other words, we think that our solutions do take advantage of the so-called biometric crypto-schemes, namely in the context of <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> generation, but it has to be clarified that a significant further improvement may be generating also the certificate with a similar approach, as a biometric key-binding scheme.

6.4 AN ONLINE BANKING SCENARIO AND ITS SECURITY ANALYSIS

The two solutions presented in this Chapter are general enough to guarantee a non-repudiation service for many scenarios in which users and remote Internet services are involved in a continuous information flow. In the following, we present a detailed description of how the BS-CNR is applied in a sample application scenario (Section 6.4.1) in order to prevent repudiation from a customer (Section 6.4.2) and from an online banking service operator (Section 6.4.2). The same analysis, which we do not report here in order to avoid excessive redundancy, can be easily performed also for DS-CNR.

6.4.1 *BS-CNR System Design and Assumptions*

Let us consider an online banking service which distinguishes between *Risky Actions* (RA_n) and *Basic Actions* (BA_n). An example of risky action is RA_1: transferring an amount of money that exceeds a specified value (e.g., five thousand dollars) to an account of a different bank and/or belonging to a different customer. An example of a basic action is BA_1: consulting services. The online banking service exploits BS-CNR to permit Risky Actions only to the legitimate client which is highly trusted. That is, in order to get a valid BS-CNR certificate, the $\mathtt{trust}_{min}$ threshold required by the Authentication Server is set to a relatively high value. Instead, the $\mathtt{trust}_{min}$ threshold is set to a lower value in order to get a BS-CNR certificate valid to perform Basic Actions.

We assume a customer intends to perform operations on his/her online banking account. The customer has been previously enrolled to the BS-CNR system. On the client side, the customer device is a desktop computer, which integrates a webcam for the acquisition of face and iris traits, and a mouse with an optical scanner for the acquisition of fingerprints [9].

We assume that on the Authentication Server the BS-CNR system depends on Sys_1, Sys_2, and Sys_3, being respectively three subsystems with the state-of-the-art biometric verification algorithms:

- Sys_1 with Google FaceNet [97], for face recognition, which showed at FAR 10^{-6} a True Acceptance Rate (TAR) of 86.473 in Megaface challenge [98], and the impressive accuracy result of 99.63%± 0.09 in LFW benchmark [99];

- Sys_2 with IRITECH algorithm for iris recognition, which showed at FAR 10^{-6} a FRR of 0.002 in the context of NIST Iris Exchange IREX I [100];

- Sys_3 with Neurotechnology algorithm for fingerprint recognition, which had at FAR $\% \leqslant 10^{-2}$ a FRR of 0.083 in the FVC-onGoing online evaluation [101].

Thus the subsystems possess a relatively high *subsystem trust level* $m(S_k, t_0)$ thanks to their low FAR_k.

If during an iteration of the protocol the client wants to perform a Risky Action (RA_n) but the Authentication Server does not provide the BS-CNR certificate, the action would not be permitted. If it happens when the operation is already ongoing, because during the previous iteration it has been permitted, the operation will be aborted.

6.4.2 *Repudiation Attempts and Security Analysis*

Before describing two repudiation attempts and performing the security analysis, we summarize the assumptions under which the analysis is conducted. The assumptions are:

1. all the communications between the entities are secure (e.g., under TLS);
2. the Authentication Server and the Certification Authority are both Trusted Third Parties;
3. the subsystems Sys_1, Sys_2, and Sys_3 adopt specific anti-spoofing and liveness detection measures so that the biometric traits are not forgeable.

The following analysis considers only attacks in terms of fraudulent behaviors of the parties and describes how the BS-CNR solution neutralizes them guaranteeing non-repudiation.

Fraudulent Customer

Consider the situation in which a customer sends money to a second account, then denies the transaction and asks for a refund in order to deliberately fraud the bank.

At time t_i, Step 2 of the protocol has been just completed, the Authentication Server has verified the identity of the user, produced a BS-CNR certificate containing a timeout of $T_i = 20$ seconds, thus setting the expiration time at $t_i + T_i$. The customer sends a RA_1 action request, providing on Step 3 all the data needed. The banking service checks the validity of the certificate, of the $\mathtt{BioK}_c^+$ key, and grants the operation until $t_i + T_i$ is reached. The customer spends more than 20s to complete the RA_1 operation, but the BS-CNR protocol continuously verifies the user identity and checks if $\mathtt{trust}(u,t)$ is always above the $\mathtt{trust}_{min}$ threshold requested by the web service to perform RA_1 operations. Otherwise, the operation is aborted. In this example, the expiration time is extended, and the customer is able to complete the operation.

A short time later, the customer repudiates the transaction, claims having been robbed and asks for the judge intervention in order to get a refund.

The online banking service is protected against this attempt of repudiation: it can provide the NRO token received on Step 3 (and Step 7) of the protocol. This protection also invalidates a possible claim of private key loss by the user: the Authentication Server can provide the evidences stored at the end of Step 1, showing that the private key $\mathtt{BioK}_c^-$ has been generated on that moment and cannot be used during subsequent sessions. Moreover, even assuming the loss of the private key, if all Sys_1, Sys_2, and Sys_3 subsystems have successfully verified the user, the probability that the multi-biometric system has authenticated an impostor is:

$$P(FA_{system}) = P(FA_1) \cap P(FA_2) \cap P(FA_3) = 10^{-14} \approx 2^{-46}. \tag{6.1}$$

Its value is comparable with the probability of inverting 2048-bit RSA used for digital signature, which is $\leqslant 2^{-60}$ [102]. Thus, we believe that biometrics constitutes an important additional layer of security, and the $P(FA_{system})$ value is sufficiently low to cause the customer's attempt of repudiation to be useless.

Besides, such a claim would be counter-productive if an investigation discovers the fraud attempt.

Fraudulent Operator

Let us examine the opposite situation, with a dishonest operator of the online banking service, who somehow obtains access to the money transferring system. We assume the bank employee succeeds in sending 20 thousand Dollars to his own account from the one of a customer. The customer who has been ripped off asks for a refund and for the judge intervention. When questioned, the online banking service is not able to produce a valid NRO token (S_2 and M_2 evidences), related to a hypothetical RA_1 operation executed by the customer on that time interval. Thus the judge solves the dispute in favor of the innocent customer and the bank is forced to refund.

7 CONTINUOUS AND TRUSTLESS NON-REPUDIATION

In this Chapter, Section 7.1 introduces the basic technological aspects constituting the background required to fully understand if and how a blockchain can contribute to improve our continuous non-repudiation mechanism. We also present a requirements-driven methodology which helps in this process, and guides the selection of the proper blockchain category. Then, in the same Section, we describe the assessment conducted to highlight risk related to this technology, especially focusing on threats and countermeasures related to the non-repudiation use case. Section 7.2 proposes a blockchain-based continuous non-repudiation mechanism, its architecture design, protocol definition, and implementation. Finally, we discuss this solution and advantages offered in terms of security and privacy.

7.1 INTRODUCING BLOCKCHAIN TECHNOLOGY

While at the beginning it has been known primarily for being the underlying technology of Bitcoin [117], and often associated with finance and *cryptocurrencies*, blockchain is rapidly becoming one of the most exciting areas of interest for researchers, investors, and companies in many other domains: scientific, social, humanitarian, political and more [118].

Recently, in 2018, Sierra Leone has been announced [119] as the first country in the World to use blockchain in a national government election, and this will be probably followed soon by West Virginia for U.S. federal elections [120]. Meanwhile, in 2017 the number of investments in blockchain startups increased by 300% [121], as well as the list of big companies involved as large investors, deployers of Blockchain-as-a-Service (BaaS), or patent owners: IBM, Microsoft, Bank of America, and MasterCard can all be considered just the tip of an iceberg. Moreover, it has been predicted that, by 2025, blockchain platforms will store 10% of the global Gross Domestic Product (GDP) [122].

With these news and numbers, we want to give an idea of the enthusiasm surrounding blockchain, considered by many not only a disruptive technology, but a real revolution for business, economies, and society [118, 123].

The reason of this huge excitement is explained by its properties of enabling mistrusting entities, applications and systems to interact in a fully decentralized fashion, without the need for any trusted third party. Blockchains are, indeed, *trustless* and decentralized databases of records, distributed applications or smart contracts that can be shared and executed among peers.

Important properties are provided by such technology, including but not limited to integrity, pseudo-anonymity, fault tolerance as a consequence of redundancy, transparency and, especially, non-repudiability.

7.1.1 *What is a Blockchain and how does it work?*

Blockchain is a technology characterized by a shared database (or ledger) distributed across a peer-to-peer network. The term recalls its structure, a chained sequence of blocks, where each block contains a set of transactions [1] and, except for the first one called genesis block, it is linked to its predecessor by means of a cryptographic hash. Blocks are linearly and chronologically added to the chain and can be seen as links of a constantly growing chain, hence the name blockchain.

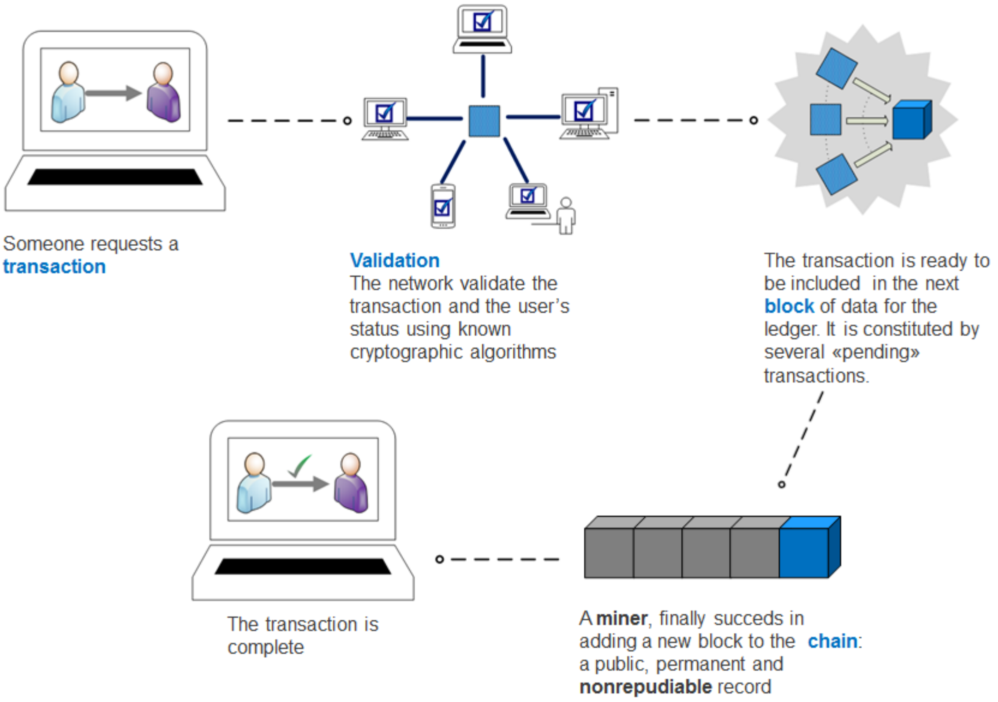

Someone requests a **transaction**
Validation
The network validate the transaction and the user's status using known cryptographic algorithms
The transaction is ready to be included in the next **block** of data for the ledger. It is constituted by several «pending» transactions.
The transaction is complete
A **miner**, finally succeds in adding a new block to the **chain**: a public, permanent and **nonrepudiable** record

Figure 36: Overall view of the steps executed when a transaction is requested in a typical blockchain network.

Each node of the network possesses a local replica of the blockchain, which is updated every time after appending a block to the chain. The process of committing a block takes the name of *mining*, and the nodes which are taking care of validating transactions, collecting them into blocks, and appending the blocks on the ledger are called miners. Nodes are typically independent peers

[1] Here, as in Section 2.5, the term transaction indicates a message exchange in general, which may or may not include a cryptocurrency transfer.

capable of reaching an agreement on the status of the blockchain, that is, the latest block to be appended, without the involvement of any central authority. This agreement is called *consensus*, and there are many different algorithms designed for reaching it.

The first application of blockchain has been financial transactions of cryptocurrencies, known as *blockchain 1.0* [118], and Bitcoin [117] is the most widespread and famous implementation. More recent alternatives enable systems and applications to record other kind of information on the ledger.
One example are distributed applications or smart contracts, executed and shared among participating entities, in which case the transaction includes the result of a function call. The smart contracts can be self-executing and general purpose, thanks to the Turing-completeness property provided in some cases, as for Ethereum [127]. These fundamental extensions of capabilities gave rise to the so-called second generation of blockchains, or *blockchain 2.0*.

Fundamental Properties

Hereafter, we describe the fundamental properties typically provided in distributed ledgers; however, their provision may be only partial for some categories of blockchain.

- *Immutability*. Due to the presence of cryptographic hashes in the blocks, transactions stored in the distributed ledger cannot be later tampered, reversed, or deleted without altering the hash values, thus without being detected.

- *Integrity*. Cryptography, together with algorithmic constraints, provide integrity on messages from users or between nodes, and ensures that operations are only performed by authorized entities. In fact, Public Key Infrastructure (PKI) and digital signatures provide accounts identification and transactions authorization.

- *Transparency*. Each participating entity has access to the distributed ledger, and can verify transactions without a central intermediary.

- *Decentralization*. There is no central authority deciding on the recording or not of a particular data in the ledger. Also, decentralization avoids single points of failure.

- *Pseudo-anonymity*. In general, each user can interact with the blockchain with a generated address. The address is a pseudonym which does not reveal the real identity of the user.

- *Non-repudiability*. We have already defined non-repudiability in the previous Chapters. In the blockchain context, its initiator digitally signs a transaction; in this way, the origin of every transaction is traced so that

Table 19: Blockchain Categories

	Permissionless	Permissioned
Public	Reading is open Writing is open	Reading is open Writing is restricted
Private/Consortium	*Not used in practice*	Reading is restricted Writing is restricted

there is no dispute about it neither on their sequence in a distributed ledger.

Permissions and Blockchain Categories

Although the properties of decentralization, transparency and the absence of a central authority are fundamental characteristics of blockchains, in some cases the participants in the network may have different authorizations and play different roles [124, 125, 126, 128]. Thus, the provision of these properties can be only partial.

Let us consider the two main operations applicable to every database: *reading* and *writing*. In a blockchain, we may call reading the operations of consulting the current state of the ledger and creating transactions. Instead, writing means validating transactions, aggregating transactions in blocks, appending blocks to the chain, and participating in the consensus protocol.

A *permissionless* blockchain is a decentralized and open system in which every node has reading and writing abilities. Examples of permissionless blockchains include Bitcoin [117] and Ethereum [127].

Permissioned blockchains, on the other hand, have been proposed as an alternative in which only a set of known and identifiable participants, previously enrolled and admitted to the blockchain, are allowed to read, write or perform both operations. Some state-of-the-art permissioned blockchains available today are Hyperledger Fabric [129], Ripple [134], Multichain [130], Kadena [133], Tendermint [131], and Chain [132].

The distinction between these classes is often combined with the notion of *public* and *private* blockchain, used to refer to reading permissions. The four resulting categories of blockchain are shown in Table 19.

Between them, public permissionless blockchains are diffused in applications involving the general population, as well as public permissioned ones but with the attribution of writing only to some priviledged nodes. Private permissioned blockchains are typically owned by one institution (or a combination of institutions, in which case they can be referred as *consortium* blockchain). Private permissionless blockchain are not used in practice as they would restrict reading while permitting writing to any node; though, some applications may exist, e.g., a shared black box for air traffic, where writing is open to every aircraft and reading is restricted to officers.

Choosing between permissionless and permissioned blockchains is not trivial, as there are often trade-offs including scalability, interoperability, cost, performance, availability, anonymity, privacy, confidentiality, transparency, and censorship resistance.

Analyzing the characteristics and the properties of blockchains, it is evident that a typical blockchain may be useful for designing a non-repudiation mechanism. However, in order to decide if and how to integrate such technology in our protocol, we perform a more detailed analysis, exploiting a requirements-driven methodology that we have proposed in [161]. In the following, we introduce the part of the methodology for the proper selection and configuration of blockchain that is relevant for the Thesis, applying our requirements.

7.1.2 *Do We Need a Blockchain?*

We have to acknowledge that nowadays there is a tendency towards considering blockchain a solution for many issues, mainly because of the wide range of crucial properties it can provide, or just because of its popularity. It is one of the emerging technologies at the peak of inflated expectations, as well as artificial intelligence or augmented reality [135]. However, a system designer, having in mind the system requirements, should consider whether this technology is actually applicable to the specific problem and convenient for its solution. The approach we propose in [161] to support the designer in this task is described with the help of flow diagrams. In part it can be considered an extension of [126], with the introduction of more criteria in the requirements analysis step and, especially, because it addresses not only the general choice of the blockchain category, but also its configuration. An overview of the whole methodology is shown in Figure 37.

The flow begins at the *Start* block and goes through the following sub processes:

1. *Project Requirements Analysis*, for the dichotomous choice between blockchain recommended or not recommended.

2. *Blockchain Choice*, the step in which, based on blockchain-specific criteria, the designer is guided on the choice of the most suitable blockchain category.

3. *Specific Configuration*, which assists the designer throughout the decision-making process for the configuration of the blockchain compliantly with the chosen category and the given project requirements.

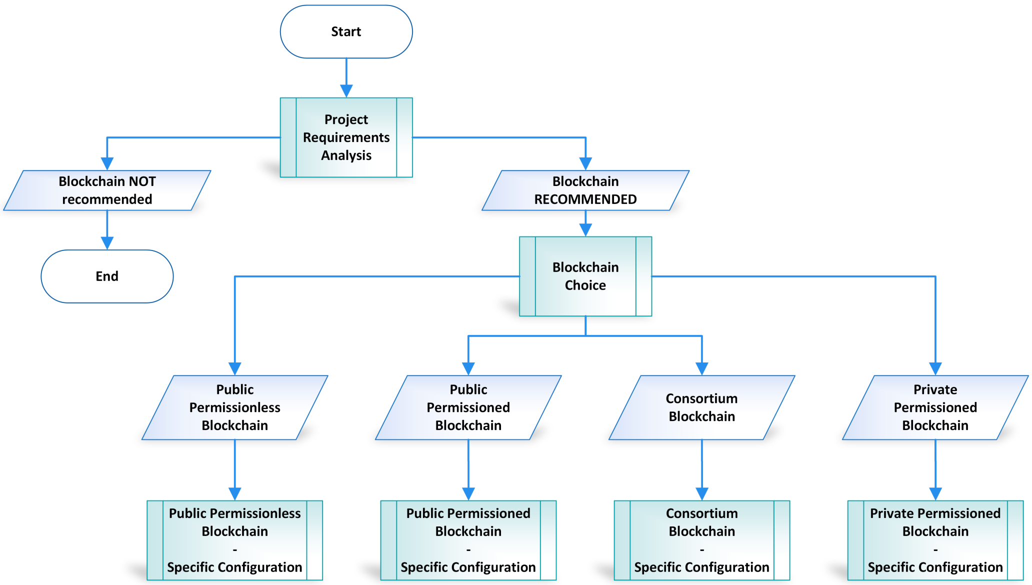

Start
Project Requirements Analysis
Blockchain NOT recommended
End
Blockchain RECOMMENDED
Blockchain Choice
Public Permissionless Blockchain
Public Permissioned Blockchain
Consortium Blockchain
Private Permissioned Blockchain
Public Permissionless Blockchain - Specific Configuration
Public Permissioned Blockchain - Specific Configuration
Consortium Blockchain - Specific Configuration
Private Permissioned Blockchain - Specific Configuration

Figure 37: Overview of the requirements-driven methodology for the proper selection and configuration of blockchains

Project Requirements Analysis. Blockchain: Yes or No?

This Section describes the first step of the proposed methodology, which consists in determining whether a blockchain is or is not an appropriate solution to address a specific problem, given its requirements. In other words, it consists in answering the question: *"Blockchain: Yes or No?"*. It is called *Project Requirements Analysis*, and the related flow diagram is shown in Figure 38.

It extends [126], introducing immutability, integrity and non-repudiability as additional criteria. It should to be noticed that some of the criteria directly drive to the result of recommending blockchain, or advising against its usage.

- *Data or State Storage*. The first criterion considered is the requirement for designing a system capable of storing data or system state. If in the project no information needs to be stored, no blockchain is needed at all, neither any traditional database. Thus, if this is not a requirement, the diagram brings to the answer *"Blockchain NOT recommended"*.

- *Immutability and Data Integrity*. If immutability is a requirement, then we can recommend using a blockchain, as this is probably the most distinctive property of any blockchain. At the same decision block of the diagram we also analyze Integrity, as these two requirements are closely related to cryptography. If a project requires data protection from unauthorized modifications, than this requirement can be met with a blockchain.

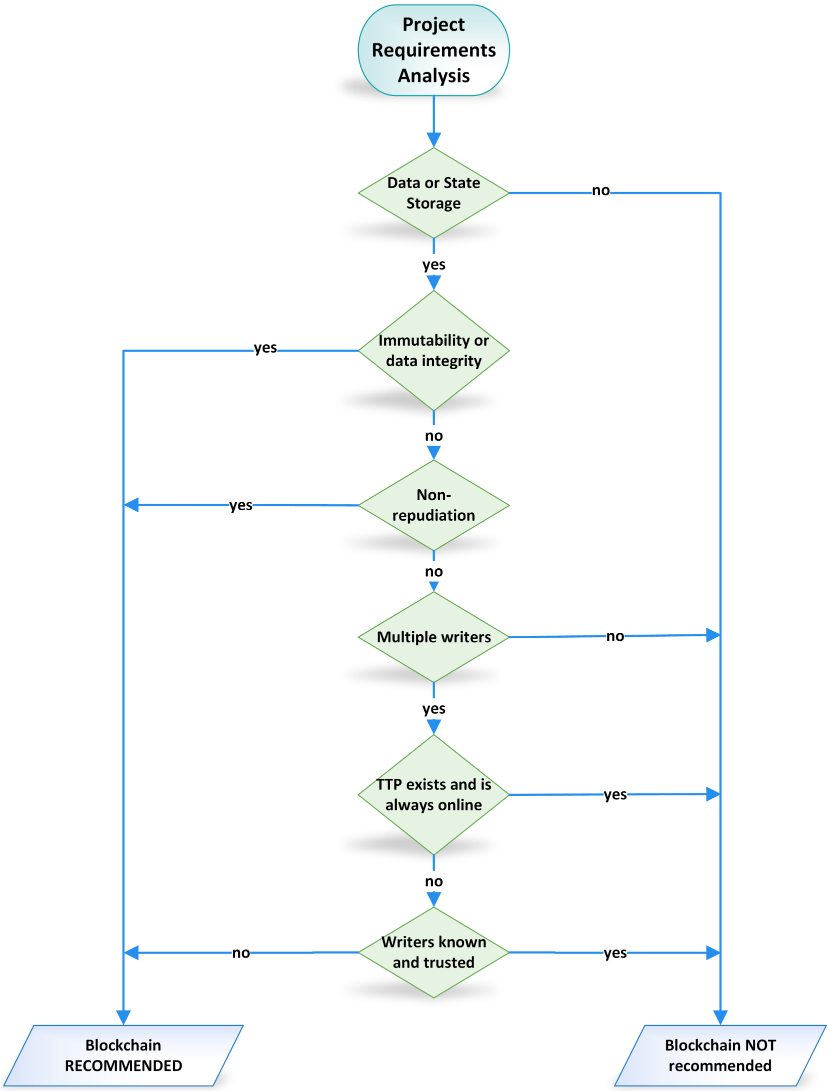

Project Requirements Analysis
Data or State Storage
no
yes
Immutability or data integrity
yes
no
Non-repudiation
yes
no
Multiple writers
no
yes
TTP exists and is always online
yes
no
Writers known and trusted
no
yes
Blockchain RECOMMENDED
Blockchain NOT recommended

Figure 38: Project Requirements Analysis: Blockchain YES or NO?

- *Non-repudiation*. Non-repudiation is another fundamental service that can be provided using blockchains; thus, if it is considered a requirement, the diagram directly produces the output: *"Blockchain RECOMMENDED"*.
- *Multiple writers*. This block of the diagram is related to the decision regarding data or state storage, and in particular it considers the multiplicity of entities in charge of writing. If only one entity is a writer, thus a common database is probably most appropriate than a blockchain, in terms of throughput and latency.
- *Trusted Third Party always online*. A Trusted Third Party (TTP) is an entity that facilitates interactions between mutually mistrusting entities. If a TTP is required in the system, and if it is planned to be always online, then entities can delegate to TTP writing operations, such as transactions or state changes. Therefore, a TTP plays the role of a trusted deliverer and verifier. In this case, a blockchain, known to be a trustless technology, becomes useless, and the methodology brings to the related output. It may happen that the involvement of a TTP is planned, but not to be always online. In this case, it could play the role of an authority giving authorizations for permissioned blockchains. Alternatively, a TTP may not exist at all. In both situations, we cannot exclude the recommendation for using a blockchain, and the analysis of the following decision block becomes necessary.
- *Writers are known and trusted*. If all entities interested in writing know and mutually trust each other, a blockchain is superfluous and, therefore, not recommended. Our advice, in this case, is to consider a traditional database shared among the writers.

Regarding the continuous non-repudiation protocol, it is quite straightforward to see that blockchain is recommended. In the following, we discuss all the criteria in Figure 38 with regard to this use case.

- *Storage of Evidences*. The entities (client, web service, and potentially also the authentication server) may need to store evidences of their interactions in order to protect from a denial.
- *Immutability and Integrity of Evidences*. Immutability of the evidence is mandatory, otherwise a malicious entity may alter it, after having committed a fraud or misbehaviour and, thus, removing traces of its involvement. According to the methodology, this would be already enough to consider a blockchain useful for our purposes.
- *Non-repudiation*. In the blockchain context, digital signature is performed every time a transaction is proposed by its initiator, so that the origin

of the transaction is traced and, therefore, there is no dispute about it. For this reason, the methodology considers it as another determining criterion to understand if blockchain is recommended. And, in our case, non-repudiability (or non-repudiation service) is exactly what we are trying to obtain.

- *Multiple entities writing evidences*. We do have multiple writers. In fact, at least one client and one web service need to store evidences of their interactions. But the authentication server may also be interested to it. And several of these entities may potentially share a single blockchain.
- *Removing the Certification Authority*. A TTP is always online? The assumption that, in order to provide continuous non-repudiation we need a TTP, is actually an assumption that we may try to relax. In fact, the architecture proposed in the previous Chapter, both for the DS-CNR and for the BS-CNR mechanisms, involve a Certification Authority. Our idea is to replace it for a blockchain. In a sense, we may consider the authentication service as a trusted third party; however, it cannot provide evidences that the client actually accessed the web service, but only that it requested a certificate. Moreover, a blockchain may be useful even if the authentication server is still present, and the flow in Figure 38 already confirms it with the analysis of second and third criterion.
- *Known writers, but trustless*. In our case, the writers are known, but they may not trust each-other (otherwise the problem of repudiation would not be raised).

This analysis confirms that the blockchain technology may help in providing continuous non-repudiation. In the following, we first present the subsequent step and then, we apply it again to our use case.

Choosing the Blockchain Category

Supposing that the outcome of the Project Requirements Analysis is that adopting a distributed ledger is recommended, the subsequent step guides the designer in the choice of the most suitable blockchain category between the possible alternatives described in Table 19.

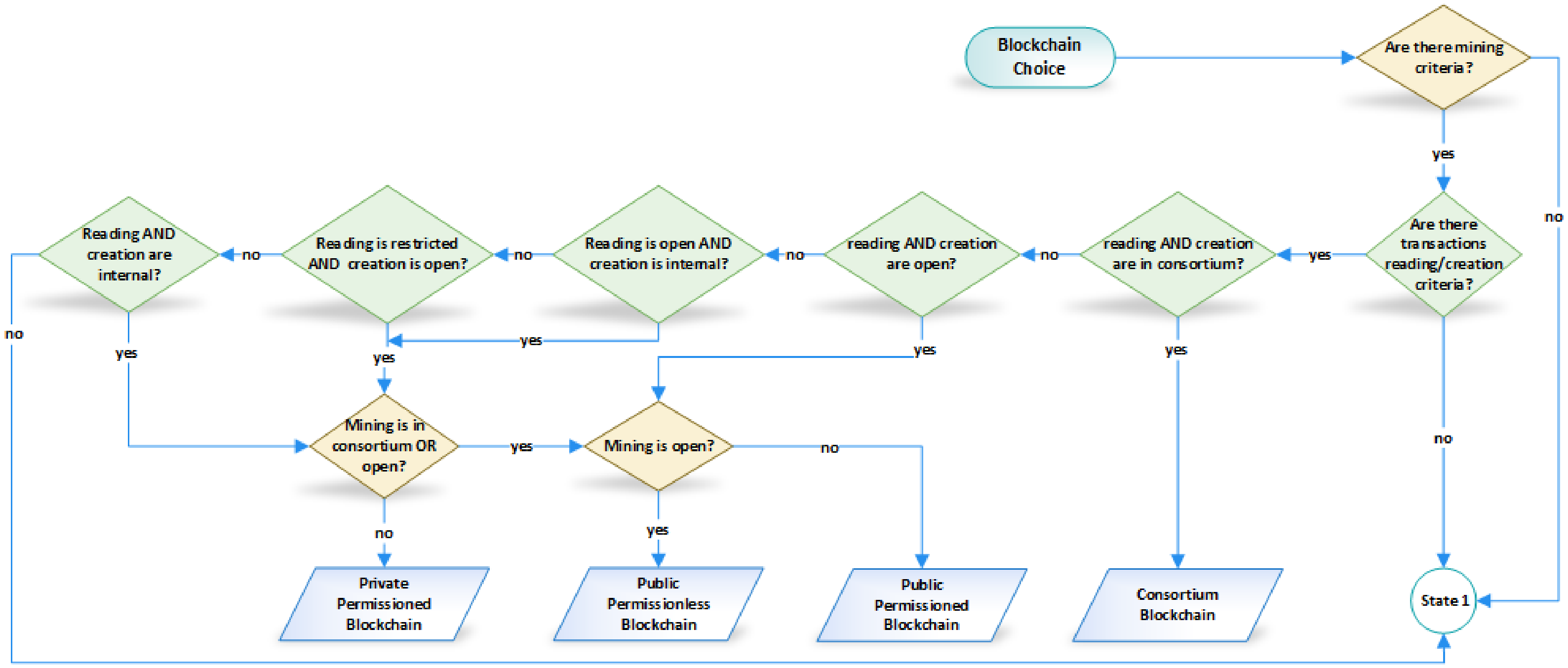

Blockchain Choice
Are there mining criteria?
yes
no
Are there transactions reading/creation criteria?
yes
no
reading AND creation are in consortium?
no
yes
reading AND creation are open?
no
yes
Reading is open AND creation is internal?
no
yes
Reading is restricted AND creation is open?
no
yes
Reading AND creation are internal?
no
yes
Mining is in consortium OR open?
yes
no
Mining is open?
yes
no
Private Permissioned Blockchain
Public Permissionless Blockchain
Public Permissioned Blockchain
Consortium Blockchain
State 1

Figure 39: Blokchain Category Choice

The *Blockchain Choice* step first considers reading and writing operations as the main criteria for the choice. As described in Section 7.1.1, these terms indicate multiple operations. In the flow diagram of Figure 39, we split reading in two criteria, separately considering:

- consulting the state of the ledger (referred to as reading in the diagram);
- creating transactions (referred to as creation).

The possible configurations are: *open*, *restricted*, and *internal*, meaning respectively that the operation is permitted to any node, permitted to some privileged nodes only, and permitted to some privileged nodes belonging to a unique organization only.

We define as mining criterion a project requirement that establishes the entities allowed to perform writing operations as referred to in Section 7.1.1; the possibilities we consider, as shown in the diagram, are: *open* mining, meaning that every peer node can be a miner; *consortium mining*, where only some nodes are allowed to write, but they can be entities belonging to different organizations, or *internal mining*, where the miners are just a subset of the nodes of the same organization. However, it may happen that, for a given project, the mining criterion is not known a priori; thus, in order to conclude the choice step, we have to consider more criteria. When this is the case, the flow brings to a sub-diagram, indicated by *State 1*; the same situation occurs if reading or creation criteria are missing.

The next *states* of this step, as well as the final step of the methodology, are not discussed here because they are slightly out of the scope of this Thesis; however, more details can be found in [161].

Applying the methodology to the continuous non-repudiation use case, we have as an outcome a *Consortium Blockchain*. In fact, our requirement is that reading, and especially writing[2] transactions are operations that have to be possible only for known entities. Another compatible outcome would be "Private Permissioned Blockchain": however, in our case the entities are not part of a single organization (as in private permissioned blockchains), but have to be considered as a group of different entities.

To resume, after having applied the requirements-driven methodology, we are now able to claim that our continuous non-repudiation mechanism may benefit from the adoption of a consortium blockchain.
However, before designing the blockchain-based continuous non-repudiation mechanism, we want to be fully aware of the possible known risks related to the adoption of this technology, and the necessary mitigations in our use case.

[2] to be intended as publishing evidences on the ledger and mining blocks

7.1.3 *Risk Assessment of Blockchain Technology*

In this Section, we identify the main threats to blockchain technology, and assess the related impacts. Then, applying a NIST-compliant approach[3], we perform a qualitative risk assessment. We review the possible countermeasures, where existing, for each threat analyzed. Finally, we discuss the consequences of their occurrence in the context of continuous non-repudiation.

Goals of The Assessment

- Giving a broad view to users and designers on possible risks in adopting this technology;
- Understanding strengths and weaknesses of a typical blockchain and how the latter may be exploited by attackers;
- Exploring solutions and countermeasures to make the blockchain more secure and resilient.

Assessment Methodology

Our analysis [162] surveys the most relevant threats, trying not to focus on a specific platform, assesses the risk related to each of the threats, and describes the respective countermeasures available so far, or desirable in the future.

Threat Sources. During the assessment, as for the assessment in Section 5.1, we choose to address adversarial threat sources only, as opposed to non-adversarial like human errors, structural failures, or natural disasters.

Threat Agents. The characteristics of the identified attackers are summarized in Table 20, and this characterization is based on the Threat Agent Library (TAL) in [146].

- *Criminal Organization (CO)* represents an external attacker, with government level resources (e.g., a corporation, a terrorist organization, or an adverse nation-state entity), and having good technological skills. This attacker maps agents 6, 7, 10, 15, and 18 in [146].
- *Hacker (Hk)* represents an external individual having high technological skills, and moderate/low resources. This attacker maps agents 5, 8, 14, 16, and 21 in [146].

In each assessment, we specify the threat agent that we consider most likely capable of conducting the attack. As an example, some threats involve the control of multiple nodes by the same entity, and we think it would be easier for a corporation, such as a CO, having at its disposal more resources. On the

[3] as in Section 5.1 for CASHMA continuous authentication protocol.

Table 20: Attackers and their characteristics

	Criminal Org. (CO)	*Hacker (Hk)*
Resources	Government	Moderate
Capabilities	Operational	Adept

other hand, other threat events require fewer resources but more capabilities, and those are the attacks for which we indicate Hk as a threat agent.

In this Thesis, we only present the risk assessment for the threats that are relevant for the continuous non-repudiation protocol. However, the complete assessment and the full description of all the threats we analyzed can be found in [162] and in [164].

The approach we followed to determine the likelihood of occurrence, as in Section 5.1, is based on our experience and confirmed by the CVSS framework [84]. Again, as proposed by NIST, the overall likelihood is expressed in a qualitative scale: from Very Low (VL), to Very High (VH). Similarly, the impacts determination and assessment, as well as the risk derivation, follow the strategy of NIST SP 800-30 [55].

Network Threats Analysis

Distributed Denial-of-Service (DDoS).
This threat is characterized by an explicit attempt to prevent the legitimate use of a service. A DDoS deploys multiple attacking entities – constituting a botnet – to attain the goal, and is typically targeted to a single point of failure. This type of attack is highly more complex to be conducted against a blockchain, thanks to the decentralized nature of this technology; in fact, by eliminating a middle man between the nodes of the network, the single point of failure disappears. Moreover, according to many sources (including for example [147, 148]), a blockchain itself can be considered as a mitigation for DDoS, or even capable of making this attack obsolete.
However, examples of successful DDoS already exist. Some are initiated by miners and targeted against adversary mining pools to eliminate their chances of winning the next competition. In [136], the authors empirically analyzed DDoS attacks, and documented 142 unique events against 40 Bitcoin services; they also found that 7% of all known operators were attacked.
Also, blockchains 2.0, as Ethereum, suffered from DDoS attacks. One example is a computational DDoS [140]: an attacker took advantage of a combination of underpriced operations and not efficient client (known as geth) implementation to create "bad transactions". The effects were a reduction in the rate of block creation, leading to a slowdown in the network, and an increase of I/O on the

clients, some of which were unable to synchronize with the blockchain. It has been fixed with a hard fork and changes in the protocol and gas costs, making the transaction spamming more expensive and ineffective [141].
All considered, the assessment is the following:
Attacker: Criminal Organization.
Targets: Adversary Mining Pool, Users.
Likelihood: High.
Impact: Moderate.
Risk: Moderate.
Countermeasures: Proof-of-Activity (PoA) protocol, fast verification, and signature based authentication [143].

In the context of continuous non-repudiation, if the blockchain is temporary unavailable, the entity is not able to store the evidences of interactions. However, timeliness is not a fundamental requirement for non-repudiation; it is sufficient to collect the evidences and have them stored on the ledger, when this is accessible. It is fundamental that the evidence is provided in case of a dispute. If one entity is isolated and cannot deposit the evidences on the ledger, it is necessary to provide a solution in case this happens (e.g., temporarily storing evidences on the device itself).

Other network threats, similar to DDoS, are: *Partitioning Attack* and *Delay Attack (or Tampering)*.

Partitioning Attack (as known as BGP hijacking [145]).
It consists in deviating and cutting all the connections between a set of nodes and the rest of the network, in order to disconnect and isolate them. In [139], the authors describe how an attacker can verifiably isolate a selected set of nodes in the Bitcoin network. The impact of a partitioning attack depends on the number of isolated nodes and on their mining power. Isolating a few nodes essentially constitutes a DoS attack. Disconnecting a considerable amount of mining power can lead to two different versions of the blockchain, with consequences as: block discarded, transactions reversed, as well as loss for the miners, risk of double spend and selfish mining attacks [139]. In the last case, the Impact is High. Likelihood is High as well, and its feasibility has been demonstrated in [139].
Attacker: Criminal Organization.
Target: Users, Miners and Network.
Likelihood: High.
Impact: High.
Risk: High.
Countermeasures: BGP security extensions, monitoring systems [145].

For the continuous non-repudiation use case, isolating an entity consists in another example of DoS, and can be addressed as we explained for DDoS. The possibility that a huge amount of nodes is disconnected may indeed lead to a creation of a different versions of the ledger, with serious consequences such as altering the contents of an evidence, or completely deleting it. However, thanks to the consortium nature of the blockchain we are going to introduce in the architecture, and considering that only authorized entities are allowed to read/write and mine the blockchain, the risk is significantly reduced.

Delay Attack or Tampering.
A typical blockchain network assumes that information about mined blocks is broadcast to the other nodes and quickly reaches them. The goal of a delay attack is to slow down this propagation. Studies that demonstrate the feasibility of this attack already exist. In [138], the authors proved that the adversary can reach the goal by either introducing congestion in the network, or making a victim node busy by sending requests to all its ports [143]. Similarly, in [139] the authors describe an attack procedure against Bitcoin in which the delivery of blocks is delayed by up to 20 minutes. This is obtained modifying some key messages while making sure that the connections are not disrupted. This attack, also known as tampering [143], can prepare for other subsequent threats, as double spending or DoS.
Attacker: Criminal Org.
Target: Users, network.
Likelihood: High.
Impact: Moderate.
Risk: Moderate.
Countermeasures: possible mitigations are, e.g., monitoring round-trip time (RTT), or using UDP heartbeats [139].

Table 21: Risk Assessment For Network Threats in Blockchains

Threat	Short Description	Threat Agent	Targets	Assessment			Countermeasures	Relevance for non-repudiation
				Likelihood	Impact	Risk		
DDoS	Exhaust network resources sending large number of requests	CO	Mining pools and users	H	M	M	Proof-of-Activity (PoA) protocol, fast verification, and signature -based authentication [143]	Timeliness not necessary: evidences will be included in blocks when possible.
Partitioning	Disconnect and isolate nodes from the network. Forks, prepare for other attacks	CO	Mining pools users, miners, network	H	M/H	M/H	Monitoring systems, BGP security extensions [145]	
Delay/Tampering	Slow down propagation of info about mined blocks, prepare for DoS	CO	Blockchain network	H	M	M	Monitoring round-trip time (RTT), use UDP heartbeats [139]	

Private Key/Wallet Threats Analysis

The private key of a blockchain user is critically important, as it constitutes the authentication factor. Since there is no TTP, if a user loses the key, irreversibly loses the crypto-coins – or any asset and confidential information – linked to the key. Moreover, if the key is stolen, it may also cause identity theft and account tampering. In fact, once a criminal steals the account, it becomes very difficult to track their behavior and recover the modified blockchain information.

Wallet Theft.
Many users maintain their cryptographic keys associated with the account with the help of *wallets*. Most of them are online, or hosted software wallets, while others are different kinds of implementation, such as hardware, paper and brain wallets [143]. Their thefts can occur because of system hacking, bugs in the software, malwares, or incorrect usage. Even if recent studies are proposing more secure alternatives to software wallets, the usage of the latter ones is still high. Therefore, in general, the likelihood of a wallet theft is High, while the impact is Moderate, considering that typically one theft affects only one user. The resulting risk is thus Moderate.
Attacker: Hacker.
Target: User.
Likelihood: High.
Impact: Moderate.
Risk: Moderate. Countermeasures: Possible solutions, apart from hardware wallets, are Password-Protected Secret Sharing (PPSS), and threshold signature based two-factor security [143].

Man-in-the-middle (Address Attack).
In some cases, the attacker does not target private keys directly. Instead, it acts as a "man in the middle", altering the recipient address of a transaction before it is signed. The malware replaces it with the thief's address. In our opinion, the man in the middle attack for the purpose of address altering, has High likelihood, as it can be conducted in many ways, some of which have been already detected [150]. Again, the impact is Moderate because limited to a single victim. The risk is thus Moderate.
Attacker: Hacker.
Target: User.
Likelihood: High.
Impact: Moderate.
Risk: Moderate.
Countermeasures: Prevention strategies for man-in-the-middle, e.g., IDS (Intrusion Detection Systems).

Vulnerabilities in the Cryptography.
To authorize transactions, many blockchains (e.g., Bitcoin and Ethereum) use the Elliptic Curve Digital Signature Algorithm (ECDSA). As well as SHA-256, ECDSA is currently also considered very strong, but this does not guarantee that it might not be broken in the far future. Moreover, mathematical complexity does not necessarily guarantee the security of a cryptographic algorithm when it is implemented in a real-world situation. Implementation can dramatically lower the security level [142], and this may have a dramatic impact in private key security. In [142], the authors discovered a vulnerability in the ECDSA scheme, through which an attacker can recover the user's private key because it does not generate enough randomness during the signature process.
Thus, the likelihood of breaking the cryptography is, in our opinion, Very Low, at least for the computational power available, even for a CO. However, for this assessment we consider the likelihood of exploiting a vulnerability in the implementation of a cryptographic algorithm as slightly higher: Low. The impact, instead, is High because the vulnerability can cause the loss of private keys for every user of a blockchain with that particular implementation.
Attacker: Criminal Organization.
Target: User.
Likelihood: Low.
Impact: High.
Risk: Low. Countermeasures: they are specific for addressing implementation related vulnerabilities. Some examples are in [142].

Considering the continuous non-repudiation use case, every entity interested in storing an evidence on the blockchain has its own private key. In this context, the loss or theft of the private key is a risk that has to be eliminated; otherwise, repudiation would be possible, undermining all the efforts.
Therefore, we assume that the entities involved in our scenario are not using online wallets, and that the countermeasures presented in this Section (e.g., prevention strategies for man-in-the middle, and IDS), necessary to face the problems of private key loss and theft, are integrated in the continuous non-repudiation mechanism.
Moreover, as already addressed in Section 6.3, the same solution is viable also here: usage of biometric signature to be included in the evidence. As discussed for the BS-CNR, this measure significantly reduces the effectiveness of a private key loss claim.

Table 22: Risk Assessment For Private Key/Wallet Threats in Blockchains

Threat	Short Description	Threat Agent	Targets	Assessment			Countermeasures	Relevance for non-repudiation
				Likelihood	Impact	Risk		
Wallet Theft	Tampering the sw that manages user keys associated with the account	Hk	User	H	M	M	Password-Protected Secret Sharing, and threshold signature based two-factor security [143]	Include a Biometric Signature in the evidence stored on the chain
Man-in-the-middle	Altering the address of a transaction before it is signed	CO	Mining pools users, miners, network	H	M	M	Typical prevention strategies for m-i-t-m, as IDS [145]	Assuming that typical prevention measures are taken.
Breaking the Cryptography	Slow down propagation of info about mined blocks, prepare for DoS	CO	Blockchain network	L	H	L	Specific for implementation related vulnerabilities. Examples are in [142]	Assuming that known vulnerabilities are removed from implementation

7.2 Block-CNR: a blockchain-based approach for continuous non-repudiation

In order to continuously provide non-repudiation, our idea is to introduce blockchain technology in the architecture designed for previous versions of the CNR protocols, as described in in the previous Chapter. This solution, in particular, builds on the BS-CNR solution presented in Section 6.3, with the aim of maintaining all its strengths, while avoiding the involvement of an additional Certification Authority.
Moreover, we believe that the usage of a blockchain, thanks to its intrinsic properties (especially immutability and integrity), enhances the storage of evidences with regard to a traditional database. As previously explained, evidences are the key elements for the provision of a non-repudiation service.
After having considered the usefulness and main risks regarding the adoption of this technology for continuous non-repudiation, we present in this Section the approach that we called *Block-CNR*.

7.2.1 *Design of the Architecture*

The overall system, shown in Figure 40, is obtained from BS-CNR, after removing the Certification Authority (CA), and substituting it with a consortium blockchain. As in the previous versions, the other main entities that compose the system are: an Authentication Server, the clients and the web services.
Again, we assume that all the connections are under TLS, and this is mandatory in order to reduce the risks highlighted during the assessments (e.g., man-in-the-middle). The couple of cryptographic keys, denoted here as K_c^- and K_c^+, are the client's private and public key, respectively: in this case they indicate the keys necessary also for interacting with the blockchain network.
As in the BS-CNR, we assume that the Authentication Server has its own couple of keys: K_{AS}^- and K_{AS}^+. As in [19], the Authentication Server is in charge of transmitting a certificate to the client. The certificate is composed of:

```
Time stamp, sequence number, ID, Decision, expiration time.
```

It should be noted that, in Figure 40, the two databases of evidences, present in the DS-CNR and BS-CNR architectures, have been removed here. The other elements, instead, are not changed.

As a pre-requirement, we assume that the entities allowed to perform read and write operations to the blockchain, as well as mining the blocks, have been previously enrolled and are considered part of a consortium. In general, each entity that interacts through Block-CNR possesses the cryptographic keys necessary to participate in the consortium blockchain. This means that the Web Service is also equipped with its own keys.

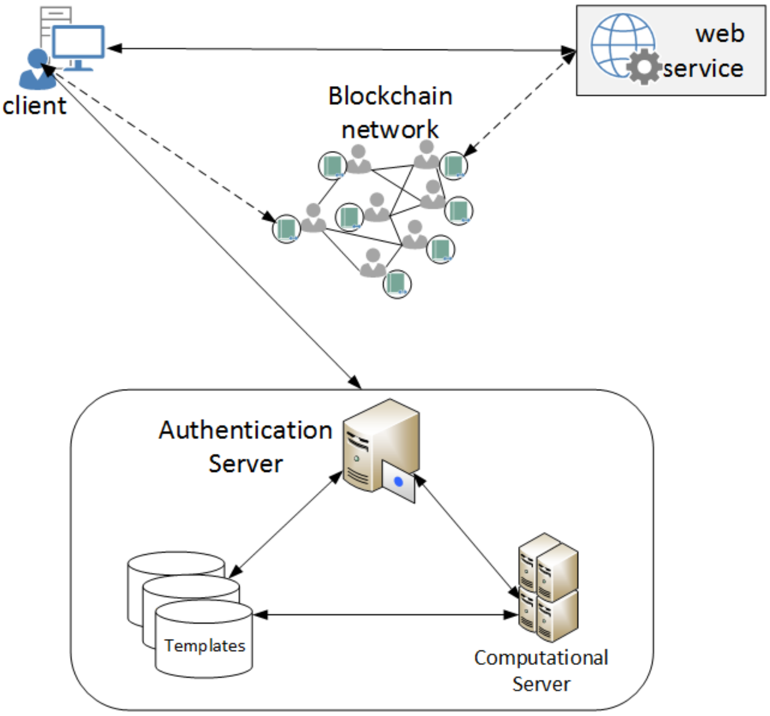


Figure 40: Overall view of the Block-CNR Architecture

7.2.2 *The Protocol*

As in the BS-CNR, in this protocol the AS exploits the biometric traits obtained from the client during the enrollment, not only for user identity verification, but also to generate the $<\mathtt{BioK}_c^+, \mathtt{BioK}_c^->$ keys. This is necessary to face the problem of private key loss, which has already been raised during the risk assessment of the blockchain technology. The biometric key pairs may be generated following one of the existing approaches e.g., from [92, 93].

Initial phase

This phase, shown in Figure 41, is now structured as follows:

Step 0 - The client performs an *Access request* to the Web Service, in order to obtain access to the critical functions. The Web Service replies with a *Certificate Request*, asking for its provision; otherwise, the access is not granted.

Exploiting the client's private key K_c^-, the application on the client side performs the signature S_1 of the message containing the biometric traits of the user.

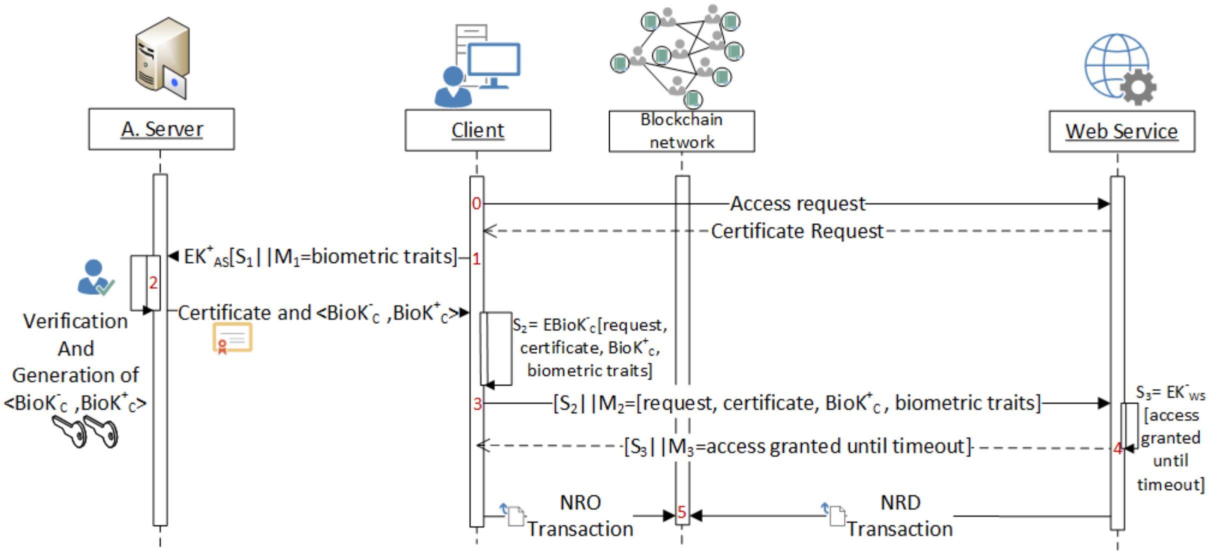

A. Server
Client
Blockchain network
Web Service
0
Access request
Certificate Request
EK+AS[S1||M1=biometric traits]
1
2
Certificate and <BioK-c ,BioK+c>
Verification And Generation of <BioK-c ,BioK+c>
S2= EBioK-c[request, certificate, BioK+c, biometric traits]
3
[S2||M2=[request, certificate, BioK+c, biometric traits]
[S3||M3=access granted until timeout]
4
S3= EK-WS [access granted until timeout]
NRO Transaction
5
NRD Transaction

Figure 41: Initial Phase of the Block-CNR Protocol

Step 1 - Message M_1, with the biometric traits, is sent at this step concatenated with its signature S_1. The resulting message is then sent after adding a further encryption layer obtained using the AS public key K^+_{AS}.

Step 2 - The Authentication Server decrypts M_1 with its private key K^-_{AS}. Then, it analyzes the biometric traits received and performs the authentication procedure as usual. As in the BS-CNR, depending on the criticality of the action to be performed, the security requirements may demand a trust threshold $\mathtt{trust}_{min}$ significantly high. So, during the same session, the threshold varies based on the operation that the client is going to accomplish, and the Decision is directly influenced by that.

If the criteria are not completely satisfied (the message in incomplete or the trust is below the threshold), the user is forced to return to Step 1.

Instead, if the user identity is successfully verified, the server authenticates the user, computes an initial timeout of length T_0, sets the expiration time to $T_0 + t_0$, and creates the certificate. Then, the Authentication Server derives the $<\mathtt{BioK}^+_c, \mathtt{BioK}^-_c>$ couple from the biometric traits.

One of the main differences here, with regard to BS-CNR, is that the storage of messages M_1 and S_1 in a database of evidences is not necessary. In fact, thanks to the involvement of a blockchain, AS is not asked to play the role of a TTP and solve possible disagreements between clients and web service providers.

AS, instead, is now only responsible for sending to the client a message containing the certificate, and the couple of biometric keys <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> assigned to the user for this session.

At the end of Step 2, the client, using its new private key $\mathtt{BioK}_c^-$, computes S_2. It is the digital signature of the message M_2, which is composed of the request, the certificate, the public key $\mathtt{BioK}_c^+$, and the biometric traits[4].

Step 3 - The client forwards M_2 to WS, concatenating it with the signature S_2 that constitutes part of the NRO token. The web service analyzes the certificate and verifies its validity; it also checks if the $\mathtt{BioK}_c^+$ has been already used before the current session.

Step 4 - As in [19], and in the previous versions of the CNR protocol, the web service reads the certificate and authorizes the client to use the requested service until expiration time.

Step 5 - This step is introduced in this version of the protocol, and constitutes the main innovation with regard to the previous versions. Both the client and the Web Service propose a transaction to be included on the next block of the ledger. These are the evidences useful in case of a repudiation attempt operated by the counterpart, and constitute the core of non-repudiation service.
NRD, is a transaction proposed by WS that contains message M2 and its signature S2. This is to protect the WS against C's false denial of having originated the message on Step 3, that is, denying the (request of) access to the web service and to the critical functions offered by it.
NRO, is a transaction proposed by the client that contains "access granted until timeout" message. This is to protect C against WS's false denial of having originated the message on Step 4, that is, having provided the access to the client.
Note that this is in agreement with the definitions of NRO and NRD in [56], as described in Section 2.5.1.
The structure and detailed content of NRO and NRD transactions are discussed in Section 7.2.3.

As in BS-CNR, after the initial phase there is a *Maintenance Phase* composed of five steps (Steps 6-9), analogous to Steps 1-5 of Figure 41. This phase is repeated iteratively: the client sends fresh biometric traits (Step 6), the server repeats the identity verification, takes the related Decision, and renews the certificate (Step 7); the client sends the new request and certificate to the web service, in order to

[4] this information, indicated as biometric traits in Figure 41, is not intended as raw biometrics, but as a transformation of their templates, e.g., performed following one of the template protection approaches indicated in [88]

have the timeout expiration postponed by the web service (Step 8); finally, new NRO and NRD transactions are proposed for inclusion on the ledger (Step 9).

Note that, as happened for BS-CNR, the <$\mathtt{BioK}_c^+$, $\mathtt{BioK}_c^-$> couple is generated only during Steps 1-2 of the Initial Phase; these operations are not repeated during the Maintenance Phase.

7.2.3 *Implementation*

In order to test if the adoption of a blockchain is actually a viable alternative for the provision of continuous non-repudiation, we created our own consortium blockchain. However, we think that the approach presented in the previous Sections of this Chapter is general enough to be adapted to different platforms and implementations.
Our implementation is based on Multichain [130, 151]. This is an open platform for building and deploying blockchain applications, already discussed in Section 7.1.1 as one of the alternative permissioned blockchains. One of the reasons for choosing Multichain is the possibility to set many fine-grained permissions. However, the objective of this implementation is only to show that it is easy to create a blockchain and set it up to effectively store the evidences necessary in our continuous non-repudiation protocol.
The architecture is composed of two nodes: *C* and *WS*, representing Client and Web Service, respectively.

Creating a Blockchain

After having downloaded and installed Multichain on both our servers, the first step is the creation of the chain, whose name in our case is `chain1`.
On WS, we run the command

```
multichain-util create chain1
```

and the output contains the message

```
[...]
Blockchain parameter set was successfully generated.
[...]
```

WS here plays the role of the *seed* node, being the node that created the chain. However, the parameters can be later set in order to give exactly the same permissions to nodes that are part of the consortium. In this way, we will not have differences between nodes, and our blockchain could be actually considered peer to peer.

Initialization

Then, we initialize the blockchain with the command:

```
multichaind chain1 -daemon
```

This step, which also includes the mining of the genesis block, produces the output:

```
MultiChain 1.0.2 Daemon (latest protocol 10009)
MultiChain server starting
Looking for genesis block...
Genesis block found
Other nodes can connect to this node using:
multichaind chain1@11.11.11.39:9731

Node started
```

Connecting and Permissions

By default, the blockchain we have created so far is a private one. This means that it has been sucessfully initialized, but any other node needs permissions in order to connect to the chain.
In fact, this is what happens if, on the C server, we control the existence of `chain1` chain without having the permission.
Executed command:

```
multichaind chain1@11.11.11.39:9731
```

Output:

```
MultiChain 1.0.2 Daemon (latest protocol 10009)
Retrieving blockchain parameters from the seed node 11.11.11.39:9731 ...
Blockchain successfully initialized.
Please ask blockchain admin or user having activate permission to let you
    connect and/or transact:
multichain-cli chain1 grant 1CW42yEtA4bLQ8saCAtmdxG8oradVPB7PQKwny connect
```

The meaning of the message is that, checking on WS (the seed node), the blockchain `chain1` that C is looking for, actually exists and has been initialized. The Multichain output also suggests the command to be executed in order to give permissions is `grant`; it has to be noticed that

`1CW42yEtA4bLQ8saCAtmdxG8oradVPB7PQKwny`

is the address (public key) of C server, and the permission we have to provide is `connect`.

Thus, what we do is running the command

```
multichain-cli chain1 grant 1CW42yEtA4bLQ8saCAtmdxG8oradVPB7PQKwny connect
```

Table 23: Permissions in Our Consortium Blockchain

Node	Role	Permissions
Web Service	Seed Node	mine, admin, activate, connect, send, receive, issue, and create
Client	-	mine, admin, activate, connect, send, receive, issue, and create
Any other node	-	none

obtaining:

```
{"method":"grant","params":["1CW42yEtA4bLQ8saCAtmdxG8oradVPB7PQKwny","
    connect"],"id":1,"chain_name":"chain1"}
5c95375b6d862e61f658eb3f7555d35e4548359feec7d12cee83beb50cbcfeac
```

The connection from C side can be easily completed with

```
multichaind chain1 -daemon
```

and the output is:

```
MultiChain 1.0.2 Daemon (latest protocol 10009)
MultiChain server starting
Retrieving blockchain parameters from the seed node 11.11.11.39:9731 ...
Other nodes can connect to this node using:
multichaind chain1@11.11.11.11:9731

Node started
```

As can be seen in this output, a third node (e.g., another Client, or another WS), which is part of the consortium and has already obtained the permissions, can easily connect to the same chain specifying the chain name, followed by either the IP of the seed node, or the IP of another node already connected to it.

The permissions, as set in our implementation, are summarized in Table 23

Where to Store Evidences

The next step is to investigate if and how we can store and retrieve evidences on the chain.
There is an item called `stream`, which can be created in a Multichain blockchain in multiple instances. Each stream acts as an independent append-only collection of items, and possesses the following features:

- `publishers` - One or more publishers who have digitally signed that item.
- `key` - An optional key (between 0 and 256 bytes in length) for convenient later retrieval.

- `data` - Some data, which can range from a small piece of text to many megabytes of raw binary. It must be an hexadecimal string.
- `timestamp` - A timestamp, which is taken from the header of the block in which the item is confirmed.

Also for streams, there are permissions that can be set in order to control entities that are allowed to publish data on a stream and manage it.

We execute the following command, where `evidences` is the name of the stream, and `false` is the setting for "open" parameter of this stream, based on which publishers must be explicitly granted per-stream write permissions.

```
create stream evidences false
```

Publishing NRO and NRD Evidences

The last step is to understand which will be the format of evidences to be published on the stream. As discussed in Section 7.2.2, the core of the evidences will be:

- NRD, on WS side, containing message M2 and its signature S2.
- NRO, on client side, containing "access granted until timeout" message.

An exemplary log for the NRD is

```
{
"client_id": "1CW42yEtA4bLQ8saCAtmdxG8oradVPB7PQKwny",
"ws_id": "1CKN5knCoPniPnsGKrFBHyAu2e4BxPFATJPBcM",
"m_2": {
                request:{
                                function_1(parameters_list)
                                ...
                                function_n(parameters_list)
                                }
                certificate: Timestamp, seq_num, ID, Decision, exp_time
                biok+: public_biometric_key
                biometric_traits:{
                                transformed_template_1
                                ...
                                transformed_template_n
                                }
"s_2": signature_of_m2
}
```

where the client_id and ws_id correspond to the respective blockchain address (or any other identifier). Therefore, this evidence can be published on a stream called evidences, with the key value expressed in the form of client_id_NRD_ws_id, followed by the content of the NRD itself, after the application of a hash function[5].

The command to be executed is the following:

```
publish evidences client_id_NRD_ws_id d94553547c00025df6036278249cc19baf39e
    910781543759450696c59019bda
```

With an analogous approach, we also publish the NRO evidence.

In case of a dispute, a stream can be queried in many different ways, e.g., indicating the stream name, or specifying also the key value. An exemplary output is the following:

```
[
    {
        "publishers" : [
            "1CKN5knCoPniPnsGKrFBHyAu2e4BxPFATJPBcM"
        ],
        "key" : client_id_NRD_ws_id,
        "data" : "d94553547c00025df6036278249cc19baf39e910781543759450696c
            59019bda",
        "confirmations" : "86",
        "blocktime" : "1515755332",
        "txid" : "23bc4302ddf8a8d8ced5663b6cf954fccedb75dd051343b299b
            8304578120912"
    }
]
```

The output shows that, together with some metadata (as number of confirmations obtained, timestamp, and transaction id), we obtained the id of the publisher entity, the key, and data, as needed.

7.2.4 *Security Considerations and Discussion*

In this Section, we analyze advantages and disadvantages obtained with this new version of the continuous non-repudation mechanism, with regard to the CNR solutions described in the previous Chapter.

This solution addresses the problem of continuous non-repudiation with the aim of taking advantage of the characteristics and intrinsic properties provided by blockchain technology.

[5] In this example we applied the SHA256 hash function the the content of NRD

Trustless Scheme

First, the involvement of a Certification Authority is avoided. Typically, its role is to provide trusted digital certificates to certify the ownership of a public key by an entity. In our scheme, the legitimacy of the entities is granted by the consortium blockchain: the cryptographic keys, necessary for the encryptions and signatures of the protocol - $<K_c^-, K_c^+>$, $<K_{AS}^-, K_{AS}^+>$ and $<K_{WS}^-, K_{WS}^+>$ - are not managed by an external authority. Instead, the key pairs are generated when an entity connects to the consortium blockchain for the first time. And this is provided only to known nodes. Moreover, in the case of the client, the assurance that the owner of $<K_c^-, K_c^+>$ is the legitimate user, is corroborated by multi-biometric verification.

Thus, the only third party that client and web service have to trust now is the Authentication Server, in charge of performing the multi-biometric continuous authentication of client users.

Biometric Signature

Then, the Block-CNR mechanism also includes the advantages of biometric signature, introduced with BS-CNR; in fact, the couple of biometric-derived keys $<\mathtt{Bio}K_c^+, \mathtt{Bio}K_c^->$ is generated also during the Initial Phase of the Block-CNR protocol. This is because there may be a claim from a client to have the private key lost, or even stolen. Moreover, having the CA removed, a client may repudiate his/her K_c^-, and there is no trusted entity which may intervene to disprove it. However, when K_c^- is used at Step 1 for the signature of M_1, that message contains multiple traits of the user; and, if they are verified by the AS and considered authentic, the repudiation of K_c^- has to be followed by a repudiation of biometric traits in M_1. Thus, if we assume the global accuracy of the multi-biometric authentication to be as described in 6.4.2, in our opinion the probability that the system has authenticated an impostor is sufficiently low to guarantee non-repudiability (see equation 6.1).

Availability, Decentralization and Fault Tolerance

Evidences of non-repudiation, in this mechanism, are included in blocks of the consortium blockchain. This removes the need of storing them in specific databases which is a single point of failure. Instead, in Block-CNR, thanks to *decentralization*, every node participating in the consortium blockchain possesses a copy of the evidences stored on the chain. This redundancy also provides *fault tolerance*. Now, in case of a dispute, a judge may always obtain evidences even if one or more copies of the evidences are not accessible for some reason (e.g., natural disasters) or temporarily unavailable.

The risk assessment performed in [162] and discussed in Section 7.1.3 with regard to the non-repudiation use case, highlighted a non-negligible risk of network threats, e.g., DDoS or other threats that can be considered a different version of DoS. In our system, we assume that known countermeasures (e.g., monitoring systems) are present. However, in the protocol, even if the blockchain is under attack and temporary unavailable, or if one node is disconnected and isolated from the network, there is no need to complete step 5 right at the moment in which the entity attempts to publish the evidences of interactions. In fact, it is enough to publish the NRO and NRD transactions when possible, and to have them available in case of repudiation attempt. The timeout information, and timestamp present in the certificate, respectively included in messages M3 and M2, will constitute a demonstration of the moment in which the access to critical functions has been granted, or has happened.

Integrity

Key properties of blockchain technology, as *immutability* and *integrity*, are very useful here: blocks containing evidences are cryptographically hashed, and evidences cannot be altered or deleted without being detected. Instead, integrity in previous versions of CNR solutions is provided thanks to TLS communications only.
As highlighted by our risk assessment [162] and discussed in Section 7.1.3, the risk of breaking the cryptography is currently negligible. However, it has to be taken into account in the future. For the moment, we assume that known vulnerabilities are removed from the implementations of cryptographic algorithms.

Confidentiality and Privacy

Regarding privacy, the presence of biometric traits of the user in message M_2, is treated as in BS-CNR: templates are not sent to the WS in raw format, but transformed using non-invertible functions (e.g., based on client specific parameters, or following one of the approaches in [88]). Thus, even if M_2 is included in the NRD and stored on the chain, there is no risk that raw biometrics of the user are stolen from the chain.

Size and Scalability

When addressing blockchains, one of the main issues which slows down its adoption is size. In fact, when a blockchain grows in size, needs more space for storage. This may be a problem, especially in case new nodes connect to the ledger for the first time and need to copy the entire chain, or nodes that come back online and have not been updated for a long period of time.
In our scenario, however, we are considering consortium blockchains, which size is considerably lower than for other categories, e.g., public permissionless ones. Moreover, the dimension of NRO data is negligible; the size of NRD token may

vary depending on the transformation applied to the biometric traits it contains, but is still limited to few bits.

Another typical issue to be taken into account is scalability. Generally, blockchains do not scale well when the volume of transactions to process grows (e.g., more than 15 per second). However, as already explained before, in our scenario timeliness is not a fundamental requirement: it is sufficient to have NRD and NRO tokens on the ledger when a dispute arises. Thus, in case the writing time grows because of a network traffic congestion, or temporal unavailability, a node may maintain its evidences in a buffer and postpone the publication on the stream. Then, with the help of metadata, a judge may query the ledger and trace back the exact position of the evidence. To conclude, we think that the expected workload on the blockchain is sustainable considering the requirements of Block-CNR mechanism, and the design and implementation choices made.

8 CONCLUSIONS

Nowadays, ICT plays an important role in our society, and security services are getting increasingly fundamental to protect users and entities involved.
In many critical systems and applications, it is mandatory that only authorized users are allowed to have services and functionalities at their disposal. In some working environments, in fact, users are in charge of analyzing potentially sensitive data, taking decisions for which they are directly responsible, and which may have serious implications on company's assets or even on citizen's safety. Their workstations should be properly protected in order to prevent undesired consequences.
Some existing solutions based on multi-biometrics addressed the problem. However, their evaluation is often conducted as a simulation, rarely with human involvement, and almost never through a proper usability assessment

In this Thesis, we presented our design, implementation and experimental evaluation of a multi-modal biometric continuous authentication system conducted taking into account user needs and behaviors, and having end users in mind in all phases of the work. Towards this end, we designed a solution which integrates face, fingerprint and keystroke recognitions and removes the necessity of explicit interactions to prove the user identity.
We defined a protocol that improves security based on the trust in the user, which is continuously computed by an authentication server. The security provided by the proposed solution can be managed through a wide set of configuration parameters.
A significant number of experiments with human participants has been performed to assess usability and security of our approach. The tests clearly stated that our system is usable and incurs in little system overhead.
Evaluations also showed that the system is satisfyingly effective and efficient. The number of users who completed the test without unexpected expirations (75%) is very high if compared with the previous studies. However, it could be further improved, e.g., replacing the three biometric subsystems and their recognition algorithms with high accurate ones.
Participants declared to be satisfied with the solution, and 91,66% of them said to be able to complete tasks effectively. As expected, with the change in the parameters, we were able to obtain a highly usable configuration, or a more secure one, without markedly decreasing usability.
As a further contribution, we publish a repository of the log files recorded by the system during experiments, which, as far as we know, is the first public dataset on logs of a continuous authentication service available to researchers.

The supplementary data consists also in detailed questionnaire results regarding user satisfaction.
We also observe that this solution has been integrated in the prototype of the Secure! crisis management system [156], in which users have to command intervention teams during emergencies.

Non-repudiation is another essential security service: it provides evidences of actions, protects against denial of involvement, and helps solving disputes between parties. Traditional non-repudiation mechanisms, as digital signature, are widely applied to prevent denial of past behaviors such as having sent or received messages. However, if the information flow is continuous, evidences should be produced for the entirety of the flow and not only at specific points. Further, non-repudiation should be guaranteed by mechanisms that do not reduce the usability of the system or application.

To meet these challenges, in the last part of the Thesis we proposed three solutions for continuous non-repudiation of remote services based on multi-biometric verification of user identities. Each of them introduces a fundamental improvement to the previous solution: they are respectively coupled with digital signature, biometric signature, and blockchain technology. As for the continuous authentication mechanism, we choose the CASHMA approach [19] as a starting point. In this Thesis, it has been evaluated from a qualitative perspective by mean of a risk assessment. The study identified the main threats for the transmission and the biometric system levels. Most of the selected countermeasures, capable of reducing the risk for the detected threats, have been integrated in the CNR solutions.

The DS-CNR and BS-CNR approaches are a demonstration that, under specific assumptions, continuous authentication mechanisms can actually be complemented with non-repudiation. The proposed approaches are able to provide continuous non-repudiation for the entire information flow between a client and a remote Internet service. The NRO and NRD evidences generated, and the Authentication Server involvement guarantee protection for both parties.
We also showed that some biometric algorithms found in the literature already have high verification accuracy. In our opinion, if those algorithms (or different ones equally accurate) are properly integrated in a multi-biometric system, with high quality sensors and anti-spoofing measures, biometric-based continuous non-repudiation is possible. The probability of authenticating an impostor is relatively low and constitutes a first evidence of user involvement. This is particularly valid for traits which inherently possess high distinctiveness. Then, coupling biometrics with other security mechanisms makes continuous non-repudiation very effective.

In the last Chapter we investigated if and how a blockchain can contribute to improve our continuous non-repudiation mechanism. We also presented a requirements-driven methodology which helps in this process, and guides the selection of the proper blockchain category. We described the assessment conducted to highlight risk related to this technology, especially focusing on threats and countermeasures relevant for the non-repudiation use case. The result of this activity is Block-CNR, a consortium blockchain-based continuous non-repudiation mechanism. We designed the architecture and defined a protocol that leverages distributed ledger technology and its properties. We provide an implementation to show that creating a blockchain and configuring it for the storage of non-repudiation evidences is feasible and constitutes an interesting alternative. Finally, we discuss this solution and its advantages, especially in terms of security and dependability attributes, taking also into account the issues that it may raise.

A concrete further development, is to apply a biometric key-binding scheme to the certificate generation step of the CNR protocols: this would constitute a significant improvement. Besides that, our future research direction is aimed at further assessing the CNR approaches, possibly through quantitative model-based evaluation and with a usability testing campaign, as happened for the continuous authentication mechanism proposed in this Thesis.

For the future of ICT, we envision an increased involvement of biometrics and blockchain technology in the seamless and trustless provision of security services, including but not limited to the ones addressed in this work.

BIBLIOGRAPHY

[1] Avizienis, A., Laprie, J. C., Randell, B., & Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. IEEE transactions on dependable and secure computing, 1(1), 11-33. (Cited on pages xi, 9, 10, 11, 13, 16, and 20.)

[2] Quality Concepts and Terminology, part 1: Generic Terms and Definitions, Document ISO/TC 176/SC 1 N 93, Feb. 1992. (Cited on page 9.)

[3] Industrial-Process Measurement and Control-Evaluation of System Properties for the Purpose of System Assessment, Part 5: Assessment of System Dependability, Draft, Publication 1069-5, International Electrotechnical Commission (IEC) Secretariat, Feb. 1992. (Cited on page 9.)

[4] Information Technology Security Evaluation Criteria, Harmonized criteria of France, Germany, the Netherlands, the United Kingdom, Commission of the European Communities, 1991. (Cited on page 10.)

[5] Pfleeger, C.P., Data Security, in Encyclopedia of Computer Science, Ralston A. et al., eds., Nature Publishing Group, pp. 504-507, 2000. (Cited on page 10.)

[6] Li, S. Z., & Jain, A. K. (Eds.). (2009). Encyclopedia of Biometrics: I-Z (Vol. 1). Springer Science & Business Media. (Cited on pages 29, 53, and 95.)

[7] Stallings, W., Brown, L., Bauer, M. D., & Bhattacharjee, A. K. (2012). Computer security: principles and practice (pp. 978-0). Pearson Education. (Cited on pages xi, 14, 15, 16, 17, and 30.)

[8] Bishop, M. A. (2005). Introduction to computer security (Vol. 50). Boston: Addison-Wesley. (Cited on page 15.)

[9] SecuGen OptiMouse Plus, `http://www.secugen.com/products/po.htm` (Cited on pages 65, 70, 72, and 106.)

[10] Jain, A. K., Ross, A., & Prabhakar, S. (2004). An introduction to biometric recognition. IEEE Transactions on circuits and systems for video technology, 14(1), 4-20. (Cited on pages 23, 24, 29, 31, 36, 65, 68, 77, and 88.)

[11] Jain, A. K., Nandakumar, A., Ross, A., (2016). 50 years of Biometric Research: Accomplishments, Challenges, and Opportunities. Pattern Recognition Letters, Volume 79, Pages 80-105, ISSN 0167-8655, August 2016. (Cited on page 23.)

[12] Delac, K., & Grgic, M. (2004, June). A survey of biometric recognition methods. In 46th International Symposium Electronics in Marine (Vol. 46, pp. 16-18). (Cited on pages 23 and 24.)

[13] Jain, A. K., Flynn, P., & Ross, A. A. (Eds.). (2007). Handbook of biometrics. Springer Science & Business Media. (Cited on page 24.)

[14] Jain, A. K., Ross, A., & Pankanti, S. (2006). Biometrics: a tool for information security. IEEE transactions on information forensics and security, 1(2), 125-143. (Cited on pages xi, 30, 32, and 33.)

[15] Reid, P. (2004). Biometrics for network security. Prentice Hall Professional. (Cited on page 24.)

[16] Albrizio, A. (2007). Biometry and anthropometry: from Galton to constitutional medicine. Journal of Anthropological Sciences, 85, 101-123. (Cited on page 23.)

[17] International Organization for Standards, ISO/IEC JTC1 SC37 Standing Document 2, version 8, Harmonized Biometric Vocabulary, 1997. (Cited on page 23.)

[18] Tripathi, K. P. (2011). A comparative study of biometric technologies with reference to human interface. International Journal of Computer Applications, 14(5), 10-15. (Cited on pages 65 and 68.)

[19] Ceccarelli, A., Montecchi, L., Brancati, F., Lollini, P., Marguglio, A., & Bondavalli, A. (2015). Continuous and transparent user identity verification for secure internet services. IEEE transactions on dependable and secure computing, 12(3), 270-283. (Cited on pages xi, 19, 35, 38, 39, 45, 46, 47, 51, 52, 64, 68, 96, 97, 99, 104, 128, 131, and 142.)

[20] Davison, A. (2005). Killer game programming in Java. " O'Reilly Media, Inc.". (Cited on pages 72 and 73.)

[21] Turk, M. A., & Pentland, A. P. (1991, June). Face recognition using eigenfaces. In Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on (pp. 586-591). IEEE. (Cited on page 73.)

[22] Bours, P., & Barghouthi, H. (2009, November). Continuous authentication using biometric keystroke dynamics. In The Norwegian Information Security Conference (NISK) (Vol. 2009). (Cited on pages 41, 73, 74, 75, and 84.)

[23] Burke, B., RESTful Java with JAX-RS, O'Reilly Media, 2009. (Cited on page 70.)

[24] Jersey Test Framework, https://jersey.java.net/. (Cited on page 70.)

[25] The OpenCV Reference Manual, Release 2.4.9.0, April 2014, `http://opencv.org`. (Cited on page 73.)

[26] JNativeHook, `https://code.google.com/p/jnativehook/` (Cited on page 73.)

[27] ISO 9241-100:2010- Ergonomics of human-system interaction - Part 100: Introduction to standards related to software ergonomics. (Cited on page 76.)

[28] Rubin, J., & Chisnell, D. (2008). Handbook of usability testing: how to plan, design and conduct effective tests. John Wiley & Sons. (Cited on page 76.)

[29] Nielsen, J. (1994). Usability engineering. Elsevier. (Cited on page 76.)

[30] Kwang, G., Yap, R. H., Sim, T., & Ramnath, R. (2009, June). An usability study of continuous biometrics authentication. In International Conference on Biometrics (pp. 828-837). Springer, Berlin, Heidelberg. (Cited on pages 38, 42, 77, 78, 80, 81, 84, 85, 87, 88, and 89.)

[31] Sim, T., Zhang, S., Janakiraman, R., & Kumar, S. (2007). Continuous verification using multimodal biometrics. IEEE transactions on pattern analysis and machine intelligence, 29(4), 687-700. (Cited on pages 38, 42, 77, 85, 86, 88, 89, and 90.)

[32] Likert, R. (1932). A technique for the measurement of attitudes. Archives of psychology. (Cited on page 78.)

[33] Lewis, J. R. (1995). IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. International Journal of Human-Computer Interaction, 7(1), 57-78. (Cited on page 78.)

[34] Prümper, J., & Anft, M. (1993). Beurteilung von Software auf Grundlage der Internationalen Ergonomie-Norm ISO 9241/10. FHTW-Berlin, Fachgebiet Wirtschaftspsychologie (Cited on page 78.)

[35] Alsultan, A., & Warwick, K. (2013). Keystroke dynamics authentication: a survey of free-text methods. International Journal of Computer Science Issues (IJCSI), 10(4), 1. (Cited on pages 80 and 83.)

[36] Jain, A. K., Klare, B., & Park, U. (2011, March). Face recognition: Some challenges in forensics. In Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on (pp. 726-733). IEEE. (Cited on page 83.)

[37] Maltoni, D., Maio, D., Jain, A. K., & Prabhakar, S. (2009). Handbook of fingerprint recognition. Springer Science & Business Media. (Cited on page 83.)

[38] ISO/IEC 18014-2:2009 - Information technology - Security techniques - Time-stamping services - Part 2: Mechanisms producing independent tokens. (Cited on page 17.)

[39] ISO/IEC JTC 1/SC 27, 2011 - Internet Security Glossary. Information technology- Security techniques - Entity authentication assurance framework. (Cited on pages 17 and 19.)

[40] Tipton, H. F., & Nozaki, M. K. (2007). Information security management handbook. CRC press. (Cited on page 19.)

[41] Talbot, E., Peisert, S., & Bishop, M. (2014). Principles of Authentication. (Cited on page 19.)

[42] Daugman, J. (2009). How iris recognition works. In The essential guide to image processing (pp. 715-739). (Cited on pages xi and 25.)

[43] Borgen, H., Bours, P., & Wolthusen, S. D. (2008, August). Visible-spectrum biometric retina recognition. In Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP'08 International Conference on (pp. 1056-1062). IEEE. (Cited on pages xi and 25.)

[44] Joyce, R., & Gupta, G. (1990). Identity authentication based on keystroke latencies. Communications of the ACM, 33(2), 168-176. (Cited on page 25.)

[45] Ross, A., Jain, A., & Pankati, S. (1999, March). A prototype hand geometry-based verification system. In Proceedings of 2nd conference on audio and video based biometric person authentication (pp. 166-171). (Cited on pages xi and 26.)

[46] Kong, A., Zhang, D., & Kamel, M. (2009). A survey of palmprint recognition. pattern recognition, 42(7), 1408-1418. (Cited on pages xi and 26.)

[47] Veeraraghavan, A., Roy-Chowdhury, A. K., & Chellappa, R. (2005). Matching shape sequences in video with applications in human movement analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(12), 1896-1909. (Cited on pages xi and 27.)

[48] Drosou, A., Ioannidis, D., Moustakas, K., & Tzovaras, D. (2012). Spatiotemporal analysis of human activities for biometric authentication. Computer Vision and Image Understanding, 116(3), 411-421. (Cited on pages xi and 27.)

[49] Feng, T., Liu, Z., Kwon, K. A., Shi, W., Carbunar, B., Jiang, Y., & Nguyen, N. (2012, November). Continuous mobile authentication using touchscreen gestures. In Homeland Security (HST), 2012 IEEE Conference on Technologies for (pp. 451-456). IEEE.

[50] Frank, M., Biedert, R., Ma, E., Martinovic, I., & Song, D. (2013). Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. IEEE transactions on information forensics and security, 8(1), 136-148. (Cited on pages 38 and 40.)

[51] Lieber, R. Disputing a Charge on Your Credit Card, 2013. `http://www.nytimes.com/2013/01/26/your-money/what-happens-when-you-dispute-a-credit-card-charge.html` (Cited on page 20.)

[52] Barboza, D. Online Brokers Fined Millions In Fraud Case, 2003. `http://www.nytimes.com/2003/01/15/business/online-brokers-fined-millions-in-fraud-case.html` (Cited on page 20.)

[53] ITU Baseline identity management terms and definitions, Recommendation ITU-T X.1252, April, 2010. (Cited on page 20.)

[54] NIST Special Pub. 800-53 Revision 4: Security and Privacy Controls for Federal Information Systems and Organizations, 2013. (Cited on pages 20 and 21.)

[55] NIST Special Pub. 800-30 Revision 1: Guide for Conducting Risk Assessments, 2012. (Cited on pages ix, 51, 53, 54, and 121.)

[56] ISO/IEC 13888-3:2009 - Information technology - Security techniques - Non-repudiation - Part 3: Mechanisms using asymmetric techniques, 2009. (Cited on pages xi, 21, 22, 95, and 131.)

[57] Diffie, W., & Hellman, M. (1976). New directions in cryptography. IEEE transactions on Information Theory, 22(6), 644-654. (Cited on page 21.)

[58] Hong, L., Jain, A. K., & Pankanti, S. (1999, October). Can multibiometrics improve performance?. In Proceedings AutoID (Vol. 99, pp. 59-64). Citeseer. (Cited on page 35.)

[59] Azzini, A., Marrara, S., Sassi, R., & Scotti, F. (2008). A fuzzy approach to multimodal biometric continuous authentication. Fuzzy Optimization and Decision Making, 7(3), 243. (Cited on pages 36, 38, and 42.)

[60] SitovÃ¡, Z., Sedenka, J., Yang, Q., Peng, G., Zhou, G., Gasti, P., & Balagani, K. S. (2016). HMOG: New behavioral biometric features for continuous authentication of smartphone users. IEEE Transactions on Information Forensics and Security, 11(5), 877-892. (Cited on pages 37 and 38.)

[61] Saevanee, H., Clarke, N., Furnell, S., & Biscione, V. (2015). Continuous user authentication using multi-modal biometrics. Computers & Security, 53, 234-246. (Cited on pages 38 and 39.)

[62] Crouse, D., Han, H., Chandra, D., Barbello, B., & Jain, A. K. (2015, May). Continuous authentication of mobile user: Fusion of face image and inertial measurement unit data. In Biometrics (ICB), 2015 International Conference on (pp. 135-142). IEEE. (Cited on pages 38 and 39.)

[63] De Marsico, M., Galdi, C., Nappi, M., & Riccio, D. (2014). Firme: face and iris recognition for mobile engagement. Image and Vision Computing, 32(12), 1161-1172. (Cited on pages 38 and 39.)

[64] Tsai, P. W., Khan, M. K., Pan, J. S., & Liao, B. Y. (2014). Interactive artificial bee colony supported passive continuous authentication system. IEEE Systems Journal, 8(2), 395-405. (Cited on pages 38 and 39.)

[65] Bailey, K. O., Okolica, J. S., & Peterson, G. L. (2014). User identification and authentication using multi-modal behavioral biometrics. Computers & Security, 43, 77-89. (Cited on pages 38 and 39.)

[66] Roth, J., Liu, X., & Metaxas, D. (2014). On continuous user authentication via typing behavior. IEEE Transactions on Image Processing, 23(10), 4611-4624. (Cited on pages 38 and 40.)

[67] Prakash, A., & Mukesh, R. (2014). A Biometric Approach for Continuous User Authentication by Fusing Hard and Soft Traits. IJ Network Security, 16(1), 65-70. (Cited on pages 38 and 40.)

[68] Draffin, B., Zhu, J., & Zhang, J. Y. (2013, November). KeySens: Passive User Authentication through Micro-behavior Modeling of Soft Keyboard Interaction. In MobiCASE (pp. 184-201). (Cited on pages 38 and 40.)

[69] Zhu, J., Wu, P., Wang, X., & Zhang, J. (2013, January). Sensec: Mobile security through passive sensing. In Computing, Networking and Communications (ICNC), 2013 International Conference on (pp. 1128-1133). IEEE. (Cited on pages 38 and 40.)

[70] Crawford, H., Renaud, K., & Storer, T. (2013). A framework for continuous, transparent mobile device authentication. Computers & Security, 39, 127-136. (Cited on pages 38 and 40.)

[71] Mondal, S., & Bours, P. (2013, September). Continuous authentication using mouse dynamics. In Biometrics Special Interest Group (BIOSIG), 2013 International Conference of the (pp. 1-12). IEEE. (Cited on pages 38 and 41.)

[72] Deutschmann, I., NordstrÃ¶m, P., & Nilsson, L. (2013). Continuous authentication using behavioral biometrics. IT Professional, 15(4), 12-15. (Cited on pages 38 and 41.)

[73] Meng, Y., Wong, D. S., & Schlegel, R. (2012, November). Touch gestures based biometric authentication scheme for touchscreen mobile phones. In International Conference on Information Security and Cryptology (pp. 331-350). Springer, Berlin, Heidelberg. (Cited on pages 38 and 41.)

[74] Shi, W., Yang, J., Jiang, Y., Yang, F., & Xiong, Y. (2011, October). Senguard: Passive user identification on smartphones using multiple sensors. In Wireless and Mobile Computing, Networking and Communications (WiMob), 2011 IEEE 7th International Conference on (pp. 141-148). IEEE. (Cited on pages 38 and 41.)

[75] Bu, S., Yu, F. R., Liu, X. P., Mason, P., & Tang, H. (2011). Distributed combined authentication and intrusion detection with data fusion in high-security mobile ad hoc networks. IEEE transactions on vehicular technology, 60(3), 1025-1036. (Cited on pages 38 and 41.)

[76] Xu, Y., Zhang, D., & Yang, J. Y. (2010). A feature extraction method for use with bimodal biometrics. Pattern recognition, 43(3), 1106-1115. (Cited on pages 38 and 41.)

[77] Niinuma, K., Park, U., & Jain, A. K. (2010). Soft biometric traits for continuous user authentication. IEEE Transactions on information forensics and security, 5(4), 771-780. (Cited on pages 38 and 39.)

[78] Kumar, A., Kanhangad, V., & Zhang, D. (2010). A new framework for adaptive multimodal biometrics management. IEEE transactions on Information Forensics and Security, 5(1), 92-102. (Cited on pages 38 and 42.)

[79] Toledano, D. T., Pozo, R. F., Trapote, A. H., & Gomez, L. H. (2006). Usability evaluation of multi-modal biometric verification systems. Interacting with Computers, 18(5), 1101-1122. (Cited on pages 38 and 42.)

[80] Altinok, A., & Turk, M. (2003, December). Temporal integration for continuous multimodal biometrics. In Proceedings of the Workshop on Multimodal User Authentication. (Cited on pages 38 and 42.)

[81] ETSI TS 102 165-1 V4.1.1 (2003). Telecommunications and internet protocol harmonization over networks (TIPHON) release 4; protocol framework definition; methods and protocols for security; part 1: Threat analysis. Technical Specification. (Cited on pages 53, 56, 57, 58, and 60.)

[82] Nostro, N., Bondavalli, A., & Silva, N. (2014, November). Adding Security Concerns to Safety Critical Certification. In Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on (pp. 521-526). IEEE. (Cited on pages 53, 56, and 58.)

[83] Chris, R. (2007). Biometric attack vectors and defenses. Computers & Security, 26, 14-25. (Cited on pages 53, 61, and 62.)

[84] CVSS Common Vulnerability Scoring System SIG `https://www.first.org/cvss/` (Cited on pages 51, 53, and 121.)

[85] Singhal, A., Winograd, T., & Scarfone, K. (2007). Guide to secure web services. NIST Special Publication, 800(95), 4. (Cited on page 47.)

[86] Dierks, T. (2008). The transport layer security (TLS) protocol version 1.2. (Cited on pages 60 and 100.)

[87] Rathgeb, C., & Uhl, A. (2011). A survey on biometric cryptosystems and cancelable biometrics. EURASIP Journal on Information Security, 1-25, 2011. (Cited on pages 62, 96, and 101.)

[88] Rane, S. (2014). Standardization of biometric template protection. IEEE MultiMedia, 21(4), 94-99. (Cited on pages 62, 131, and 138.)

[89] Bidgoli, H. (2006). Handbook of Information Security, Information Warfare, Social, Legal, and International Issues and Security Foundations (Vol. 2). John Wiley & Sons. (Cited on page 95.)

[90] Kuhn, D. R., Hu, V. C., Polk, W. T., & Chang, S. J. (2001). Introduction to public key technology and the federal PKI infrastructure. National Inst of Standards and Technology Gaithersburg MD. (Cited on page 95.)

[91] Kholmatov, A., & Yanikoglu, B. (2006, November). Biometric cryptosystem using online signatures. In International Symposium on Computer and Information Sciences (pp. 981-990). Springer, Berlin, Heidelberg. (Cited on pages 95, 96, and 101.)

[92] Feng, H., & Choong Wah, C. (2002). Private key generation from on-line handwritten signatures. Information Management & Computer Security, 10(4), 159-164. (Cited on pages 95, 96, 101, 102, and 129.)

[93] Janbandhu, P. K., & Siyal, M. Y. (2001). Novel biometric digital signatures for Internet-based applications. Information Management & Computer Security, 9(5), 205-212. (Cited on pages 95, 96, 101, 102, and 129.)

[94] Burnett, A., Byrne, F., Dowling, T., & Duffy, A. (2007). A Biometric Identity Based Signature Scheme. IJ Network Security, 5(3), 317-326. (Cited on pages 95, 96, and 101.)

[95] Stallings, W. (2017). Cryptography and network security: principles and practice (p. 743). Upper Saddle River, NJ: Pearson. (Cited on pages 100 and 101.)

[96] Nagar, A., Nandakumar, K., & Jain, A. K. (2010, January). Biometric template transformation: a security analysis. In Media Forensics and Security II (Vol. 7541, p. 75410O). International Society for Optics and Photonics. (Cited on page 105.)

[97] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815-823). (Cited on page 106.)

[98] Kemelmacher-Shlizerman, I., Seitz, S. M., Miller, D., & Brossard, E. (2016). The megaface benchmark: 1 million faces for recognition at scale. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4873-4882). http://megaface.cs.washington.edu/ (Cited on page 106.)

[99] Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E. (2007). Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49, October, 2007. http://vis-www.cs.umass.edu/lfw/ (Cited on page 106.)

[100] Grother, P., Tabassi, E., Quinn, G., and Salamon, W. (2009). Irex I - Performance of Iris Recognition Algorithms on Standard Images, NIST Interagency Report 7629. NIST Information Access Division. https://www.nist.gov/itl/iad/image-group/irex-i (Cited on page 106.)

[101] Dorizzi, B., Cappelli, R., Ferrara, M., Maio, D., Maltoni, D., Houmani, N., Garcia-Salicetti, S., & Mayoue, A. (2009, June). Fingerprint and on-line signature verification competitions at ICB 2009. In International Conference on Biometrics (pp. 725-732). Springer, Berlin, Heidelberg. https://biolab.csr.unibo.it/FvcOnGoing/UI/Form/Home.aspx (Cited on page 106.)

[102] Koo, C. Y., Yakovenko, N., Blank, J., & Katz, J. 1 Digital Signature Schemes., Lecture 16 of Advanced Topics in Cryptography, 2004. (Cited on page 108.)

[103] Verissimo, P. E., Neves, N. F., Cachin, C., Poritz, J., Powell, D., Deswarte, Y., Stroud, R., & Welch, I. (2006). Intrusion-tolerant middleware: The road to automatic security. IEEE Security & Privacy, 4(4), 54-62. (Cited on pages xi and 14.)

[104] Chinchani, R., Iyer, A., Ngo, H. Q., & Upadhyaya, S. (2005, June). Towards a theory of insider threat assessment. In Dependable Systems and Networks, 2005. DSN 2005. Proceedings. International Conference on (pp. 108-117). IEEE. (Cited on page 16.)

[105] Hunker, J., & Probst, C. W. (2011). Insiders and Insider Threats-An Overview of Definitions and Mitigation Techniques. JoWUA, 2(1), 4-27. (Cited on page 16.)

[106] ITU-T, Rec. X.800, (1991). Security architecture for Open Systems Interconnection. (Cited on page 17.)

[107] Shirey, R., (2000). Internet Security Glossary, RFC Editor. (Cited on page 17.)

[108] Sasse, M. A., & Flechais, I. (2005). Usable security: Why do we need it? How do we get it?. In Security and Usability: Designing secure systems that people can use, O'Reilly.

[109] ThisData Verify, OneLogin, Inc. https://thisdata.com/solutions/continuous-authentication (Cited on pages 43 and 44.)

[110] NoPassword https://www2.nopassword.com (Cited on pages 43 and 44.)

[111] BioCatch https://www.biocatch.com/ (Cited on pages 43 and 44.)

[112] BehavioSec https://www.behaviosec.com (Cited on pages 43 and 44.)

[113] BioTracker, by Plurilock https://www.plurilock.com (Cited on page 44.)

[114] Kryptowire https://www.kryptowire.com/continuous-authentication.php (Cited on page 44.)

[115] SensifyID by Zighra https://zighra.com/ (Cited on page 44.)

[116] Espacenet https://worldwide.espacenet.com/

[117] Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. (Cited on pages 109, 111, and 112.)

[118] Swan, M. (2015). Blockchain: Blueprint for a new economy. "O'Reilly Media, Inc.". (Cited on pages 109 and 111.)

[119] Swiss-based Agora powers world's first ever blockchain elections in Sierra Leone, press release, March 2018. https://agora.vote/pdf/Agora_Press-release_SL2018.pdf (Cited on page 109.)

[120] West Virginia Secretary of State's Office, March 2018. https://sos.wv.gov/News-Center/Pages/Military-Mobile-Voting-Pilot-Project.aspx (Cited on page 109.)

[121] Fiedler, M., Sandner, P., (2017). Identifying leading blockchain startups on a worldwide level. FSBC Working Paper. (Cited on page 109.)

[122] Espinel, V. (2015). Deep shift, technology tipping points and societal impact. In New York: World Economic Forum-Global Agenda Council on the Future of Software & Society. (Cited on page 109.)

[123] Tapscott, D., & Tapscott, A. (2016). Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world. Penguin. (Cited on page 109.)

[124] Cloud Standards Customer Council, (2017). Cloud Customer Architecture for Blockchain. (Cited on page 112.)

[125] Xu, X., Weber, I., Staples, M., Zhu, L., Bosch, J., Bass, L., ... & Rimba, P. (2017, April). A taxonomy of blockchain-based systems for architecture design. In Software Architecture (ICSA), 2017 IEEE International Conference on (pp. 243-252). IEEE. (Cited on page 112.)

[126] Wüst, K., & Gervais, A. (2017). Do you need a Blockchain?. IACR Cryptology ePrint Archive, 2017, 375. (Cited on pages 112, 113, and 114.)

[127] Wood, G., (2014). Ethereum: A secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper, 151, 1-32. (Cited on pages 111 and 112.)

[128] Meijer, D. B. (2017). Consequences of the implementation of blockchain technology. Master's Thesis, Delft University of Technology. (Cited on page 112.)

[129] Hyperledger Fabric `https://www.hyperledger.org/projects/fabric` (Cited on page 112.)

[130] Multichain `https://www.multichain.com/` (Cited on pages 112 and 132.)

[131] Tendermint `https://tendermint.com/` (Cited on page 112.)

[132] Chain `https://chain.com/` (Cited on page 112.)

[133] Martino W., Kadena, The first scalable, high performance private blockchain, `http://kadena.io`, 2016. (Cited on page 112.)

[134] Schwarzt D., Youngs N., Britto A., The Ripple protocol consensus Algorithm, Ripple Labs Inc., 2014. (Cited on page 112.)

[135] Panetta, C. K. Top trends in the Gartner Hype Cycle for emerging technologies. 2017. (Cited on page 113.)

[136] Vasek, M., Thornton, M., & Moore, T. (2014, March). Empirical analysis of denial-of-service attacks in the Bitcoin ecosystem. In International Conference on Financial Cryptography and Data Security (pp. 57-71). Springer, Berlin, Heidelberg. (Cited on page 121.)

[137] Timejacking & Bitcoin http://culubas.blogspot.it/2011/05/timejacking-bitcoin_802.html

[138] Gervais, A., Ritzdorf, H., Karame, G. O., & Capkun, S. Tampering with the delivery of blocks and transactions in bitcoin. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (pp. 692-705). ACM. October 2015. (Cited on page 123.)

[139] Apostolaki, M., Zohar, A., & Vanbever, L. Hijacking bitcoin: Routing attacks on cryptocurrencies. In Security and Privacy (SP), 2017 IEEE Symposium on (pp. 375-392). IEEE. May 2017. (Cited on pages 122, 123, and 124.)

[140] The Ethereum network is currently undergoing a DoS attack https://blog.ethereum.org/2016/09/22/ethereum-network-currently-undergoing-dos-attack/ (Cited on page 121.)

[141] Defeating the Ethereum DDos Attacks https://medium.com/@tjayrush/defeating-the-ethereum-ddos-attacks-d3d773a9a063 (Cited on page 122.)

[142] Mayer, H. ECDSA security in bitcoin and ethereum: a research survey. CoinFaabrik, June, 28, 2016. (Cited on pages 126 and 127.)

[143] Conti, M., Chhagan L., and Sushmita R.,(2017). A survey on security and privacy issues of bitcoin." arXiv preprint arXiv:1706.00916. (Cited on pages 122, 123, 124, 125, and 127.)

[144] Ethereum Smart Contract Best Practices. Known Attacks https://consensys.github.io/smart-contract-best-practices/known_attacks/

[145] Li, X., Jiang, P., Chen, T., Luo, X., & Wen, Q. (2017). A survey on the security of blockchain systems. Future Generation Computer Systems. (Cited on pages 122, 124, and 127.)

[146] Casey, T. (2007). Threat agent library helps identify information security risks. Intel White Paper. (Cited on page 120.)

[147] Cyber Security with Blockchain. Prevention of DDoS attacks with Blockchain technology. Deloitte. https://www2.deloitte.com/de/de/pages/technology-media-and-telecommunications/articles/cyber-security-prevention-of-ddos-attacks-with-blockchain-technology.html (Cited on page 121.)

[148] Rodrigues, B., Bocek, T., Lareida, A., Hausheer, D., Rafati, S., & Stiller, B. A Blockchain-Based Architecture for Collaborative DDoS Mitigation with Smart Contracts. In IFIP International Conference on Autonomous Infrastructure, Management and Security (pp. 16-29). Springer, Cham, 2017. (Cited on page 121.)

[149] Douceur, J. R., (2002). The sybil attack. International workshop on peer-to-peer systems. Springer, Berlin, Heidelberg.

[150] Ledger Addresses Man in the Middle Attack That Threatens Millions of Hardware Wallets `https://news.bitcoin.com/ledger-addresses-man-in-the-middle-attack-that-threatens-millions-of-hardware-wallets/` (Cited on page 125.)

[151] MultiChain private blockchain-White paper. Greenspan, G. (2015). `http://www.multichain.com/download/MultiChain-White-Paper.pdf`. (Cited on page 132.)

[152] Chen, C. H., & Te Chu, C. (2006, January). Fusion of face and iris features for multimodal biometrics. In International Conference on Biometrics (pp. 571-580). Springer, Berlin, Heidelberg. (Cited on pages xi, 30, and 31.)

[153] Rattani, A., & Poh, N. (2013, June). Biometric system design under zero and non-zero effort attacks. In Biometrics (ICB), 2013 International Conference on (pp. 1-8). IEEE. (Cited on page 33.)

[154] Matsumoto, T., Matsumoto, H., Yamada, K., and Hoshino, S. Impact of artificial "gummy" fingers on fingerprint systems. In Proc. of SPIE Opt. Sec. Counterfeit Deterrence Tech. IV, pages 275-289, 2002. (Cited on pages xi, 33, and 34.)

[155] Uludag, U., Pankanti, S., Prabhakar, S., & Jain, A. K. (2004). Biometric cryptosystems: issues and challenges. Proceedings of the IEEE, 92(6), 948-960. (Cited on page 105.)

[156] SECURE! - Smart platform based on crowdsourcing and crowdsensing technologies for safety and for the management of crisis and emergencies. POR CReO 2007-2013 `http://secure.eng.it` (Cited on pages 64 and 142.)

[157] Schiavone, E., Ceccarelli, A., & Bondavalli, A. (2015, November). Continuous User Identity Verification for Trusted Operators in Control Rooms, In 15th International Conference on Algorithms and Architectures for Parallel Processing, (co-located with PRDC 2015, the 21st IEEE Pacific Rim International Symposium on Dependable Computing). Springer International Publishing, LNCS Volume 9532, Pages 187-200.

[158] Schiavone, E., Ceccarelli, A., Bondavalli, A., and Carvalho, A., (2016) Usability Assessment in a Multi-Biometric Continuous Authentication System, IEEE Proceedings of LADC 2016, 7th Latin-American Symposium on Dependable Computing, Pages 43-50.

[159] Schiavone, E., Ceccarelli, A., & Bondavalli, A., (2017). Risk Assessment of a Biometric Continuous Authentication Protocol for Internet Services,

ITASEC17, 1st Italian Conference on Cybersecurity. CEUR Workshop Proceedings, Vol. 1816, Pages 53-65.

[160] Schiavone, E., Ceccarelli, A., & Bondavalli, A., (2017). Continuous Biometric Verification for Non-Repudiation of Remote Services, ACM International Conference Proceedings of the 12th International Conference on Availability, Reliability and Security, ARES.

[161] Staderini, M., Schiavone, E., and Bondavalli, A. (2018). A Requirements-Driven Methodology for the Proper Selection and Configuration of Blockchains. IEEE Proceedings of SRDS 2018, the 37th International Symposium on Reliable Distributed Systems. (Cited on pages 113 and 119.)

[162] Morganti, G., Schiavone, E., and Bondavalli, A. (2018). Risk Assessment of Blockchain Technology. Proceedings of LADC 2018, the 8th Latin-American Symposium on Dependable Computing. (Cited on pages 120, 121, 137, and 138.)

[163] Continuous authentication log dataset `http://rcl.dsi.unifi.it/~enrico/BCAS`

[164] Appendix of blockchain risk assessment `http://rcl.dimai.unifi.it/~enrico/blockchain-ra` (Cited on page 121.)