

# An Automated Multi-Layered Methodology to Assist the Secure and Risk-Aware Design of Multi-Factor Authentication Protocols

Marco Pernpruner, Roberto Carbone, Giada Sciarretta, and Silvio Ranise

**Abstract**—Authentication protocols represent the entry point to online services, so they must be sturdily designed in order to allow only authorized users to access the underlying data. However, designing authentication protocols is a complex process: security designers should carefully select the technologies to involve and integrate them properly in order to prevent potential vulnerabilities. In addition, these choices are usually restricted by further factors, such as the requirements associated with the scenario, the regulatory framework, the dimensions to balance (e.g., security vs. usability), and the standards to rely on. We come to the rescue by presenting an automated multi-layered methodology we have developed to assist security designers in this phase: by repeatedly evaluating their protocols, they can select the security mitigations to consider until they reach the desired security level, thus enabling a security-by-design approach. For concreteness, we also show how we have applied our methodology to a real use case scenario in the context of a collaboration with the Italian Government Printing Office and Mint.

**Index Terms**—Authentication, risk analysis, security analysis, security methodology.

## 1 INTRODUCTION

THE National Institute of Standards and Technology (NIST) defines authentication protocols as «a defined sequence of messages between a claimant and a verifier that demonstrates that the claimant has possession and control of one or more valid authenticators to establish their identity» [1]. Authenticators represent the core elements within the authentication procedures, and can attest one or more authentication factors: something that the claimant *knows* (knowledge factors), *owns* (ownership factors), or *is* (inherence factors).

Considering that authentication potentially allows to access sensitive information and perform restricted operations, malicious agents frequently target this phase: in the first quarter of 2022, almost 113 million attacks have been performed against multi-factor authentication systems, with the main objectives being staffing/recruiting (4.45%), public (3.99%), and financial (3.86%) services [2]. This trend is confirmed when we focus on financial services: 80% of the organizations have suffered from at least one cyber breach due to authentication weaknesses, and 95% of total breaches could be presumedly ascribable to credential misuse or authentication vulnerabilities [3].

Given the considerable number of attacks, authentication protocols should be designed in order to be sufficiently resistant and guarantee a significant level of security. However, designing security protocols is not trivial: first of all, security designers need to understand the scenario in which the authentication protocol is deployed, as it usually im-

poses some constraints regarding the environment and/or the legal framework to comply with. Then, many other choices have to be made: the proper balance between different dimensions (e.g., usability and privacy), the standards to rely on, and the authenticators to use in the protocol, which must be properly configured and integrated between each other without exposing the protocol to vulnerabilities. During the process, security designers usually need to analyse the security of several configurations of the system to identify the most appropriate one. However, performing a manual analysis of each configuration would lead to greater efforts, longer development lifecycles and a higher probability of missing some vulnerabilities [4], [5]. Automated techniques can greatly support the analysis process of authentication protocols due to their reliance on advanced methods such as model checkers, which can turn the security analysis into large satisfiability problems [6]. As a consequence, these techniques are fundamental in detecting complex, uncovered vulnerabilities that affected common security protocols [7]. Unfortunately, such techniques are computationally very intensive as they suffer the state space explosion problem and may be difficult to exploit to quickly evaluate alternative configurations of a design.

To alleviate this problem and allow the usage of formal techniques during the design phase, we have developed a multi-layered methodology that supports security experts in the design of authentication protocols, thus fostering a security-by-design approach. Our methodology can be repeatedly employed as an oracle in the design process, with security designers giving in input different configurations of the protocols until they reach the desired security level. The multiple layers allow to satisfy some requirements that we have identified from our experience in protocol design and analysis: (i) *Efficiency*, to support security experts in reasonable time; (ii) *Classification*, to understand the riskiness

- The authors are with the Center for Cybersecurity, Fondazione Bruno Kessler, Trento, Italy. E-mail: {mpnpruner, carbone, g.sciarretta, ranise}@fbk.eu.
- Marco Pernpruner is also with the Department of Informatics, Bioengineering, Robotics and System Engineering, University of Genoa, Italy.
- Silvio Ranise is also with the Department of Mathematics, University of Trento, Italy.

of the successful attackers. In particular, the *combinatorial analysis* performs a fast, yet incomplete, high-level analysis to reduce the number of invocations of the *symbolic analysis*, which relies on formal frameworks and is connected with a higher computational complexity; this complies with the first requirement. Finally, to meet the second requirement, the *risk analysis* complements the list of attackers that are able to violate the protocols with the related risk, to understand which of them represent the most relevant threats.

For the sake of concreteness, we also show how we have applied our methodology to a real use case scenario: an authentication procedure based on QR codes and electronic documents that currently represents one of the main authentication procedures to access Italian Public Administration's online services. This activity has been performed in the context of a long-standing collaboration with *Poligrafico e Zecca dello Stato Italiano* (IPZS, the Italian Government Printing Office and Mint) and has allowed them to gradually refine the protocol until they reached the desired trade-off between security and usability.

## Structure of the Paper

Section 2 describes the methodology that we have developed to support the design of authentication solutions. Section 3 introduces a concrete authentication protocol that we use as a practical example. Section 4 shows the application of the methodology to the use case scenario. Section 5 identifies and discusses the role of security mitigations along with their implications from security and usability perspectives. Section 6 presents related works. Section 7 draws some conclusions and hints future works.

## 2 SECURITY ANALYSIS METHODOLOGY

To support the design of authentication protocols, we have developed a methodology – displayed in Fig. 1 – that aims at detecting all the (combinations of) attackers  $A^1$  that are able to violate the security goal  $G$ :

$$A = \{A \subseteq \mathcal{TM} \mid M_P, \mu(A) \not\models G\} \quad (\star)$$

with the following specifications required as input:

- a *model of the protocol* ( $M_P$ ) derived from the Message Sequence Chart (MSC, in blue), a detailed representation of the protocol to be analysed from which we can extract the authentication factors used, the entities, their initial knowledge, the messages they exchange, the communication channels used, and the security assumptions;
- a *model of the attackers* ( $M_A$ ) obtained by using a list of potential attackers (i.e., a *threat model*  $\mathcal{TM}$ ) equipped with a set of capabilities (in orange). In Section 4.2, we provide a reference model that can be extended by a security expert according to the needs. This model takes inspiration from the *Authenticator Threats* introduced

1. It is worth underlining that  $A$  can represent either single attackers or combinations of attackers: in the first case, attackers are successful by relying on their own capabilities; in the second case, they need to collude and combine their capabilities. For readability, single attackers  $A = \{a\}$  can be denoted as  $\{a\}$  or  $a$ ; combinations of attackers  $A = \{a_1, \dots, a_n\}$  can be denoted as  $\{a_1, \dots, a_n\}$  or  $a_1 + \dots + a_n$ .

by NIST [8]. To formalise the capabilities of a specific attacker, we introduce a function  $\mu : \mathcal{TM} \rightarrow M_A$ , which takes in input an attacker  $A \in \mathcal{TM}$  and returns the specification of the corresponding capabilities from the model of the attackers  $M_A$ .

In case of authentication protocols,  $G$  represents the fact that the intended service (called *Service Provider*) must authenticate the user with a given level of assurance, which is specified by the service itself depending on many factors.

From our experience in the design and analysis of authentication protocols, we have identified the following requirements that our methodology should meet:

- R1. Efficiency:** the methodology should be efficient enough to allow security experts to promptly obtain results during the design of authentication protocols;
- R2. Classification:** the methodology should provide a clear classification of the successful attackers to allow security designers to understand the related risk.

For this reason, we have structured our methodology in multiple layers: *combinatorial analysis*, *symbolic analysis*, and *risk analysis*; this way, we can enable security designers to make informed decisions while maintaining the cost of invoking automated security analysis techniques at a reasonable level. In particular, the combinatorial layer requires high-level specifications and performs a fast, yet incomplete, analysis whose role is to reduce the number of attackers to test in the following layer – which is more computationally expensive – and comply with *R1*. To provide complete results, the second layer is represented by the symbolic analysis, which relies on advanced frameworks requiring formal specifications with cryptographic details; this analysis suffers from the well-known state space explosion problem that is common to several state-based techniques [9]. Finally, the risk analysis collects the results of the previous analyses and complements them with a risk assessment that provides a basis to plan the mitigations, thus complying with *R2*.

### 2.1 Combinatorial Analysis

The first layer of our methodology aims at finding:

$$A_C = \{A_C \subseteq \mathcal{TM} \mid M_{P_C}, \mu_C(A_C) \not\models_C G_C\} \quad (\star_c)$$

with

$$\mu_C : \mathcal{TM} \rightarrow M_{A_C}$$

This is achieved through a high-level analysis based on authentication factors:  $G_C$  holds when attackers are not able to compromise all the authentication factors involved in the protocol. In fact, attackers who compromise all the authentication factors cause a violation of the whole protocol, and are reported by the combinatorial analysis. In this context,  $M_{P_C}$  represents the list of authentication factors involved in the protocol (inferred from the MSC), while  $M_{A_C}$  represents the attackers' capabilities in terms of compromised authentication factors: we can consider them as a table where rows represent attackers and columns represent authentication factors: each pair  $(a, af)$  specifies whether attacker  $a$  is able or not to compromise the authentication factor  $af$ . We may also use the notation  $a \xrightarrow{\blacksquare} af$ , where  $af$  can be a compromised authentication factor or  $\emptyset$  in case the attacker  $a$  has no effect on the protocol.

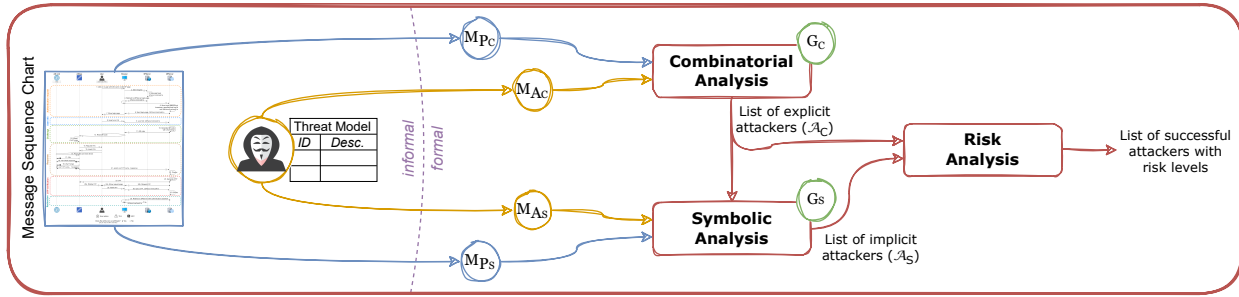


Fig. 1. A graphical representation of our methodology

TABLE 1  
Attackers' capabilities ( $M_{A_C}$ ) in Example 1

Attackers	Authentication Factors	
	Password	Smartphone
Thief		
Social Engineer		
Thief + Social Engineer		
Thief + Social Engineer + Any		

= safe = compromised

In particular,  $A_C$  is reported by the combinatorial analysis (and thus is a solution of  $\star_C$ ) iff for each authentication factor  $af \in M_{P_C}$  there exists an attacker  $a \in A_C$  so that  $a \xrightarrow{af}$ . These attackers, resulting from an explicit violation of the authentication factors, are called *explicit*.

**Example 1.** Let us consider an authentication protocol in which users need to insert a password and scan a QR code through an application on their smartphone (on which they must already be authenticated). The protocol is composed of two authentication factors: the password (i.e., a knowledge factor) and the smartphone (i.e., an ownership factor). Let us consider two attackers: thieves, who physically steal devices, and social engineers, who deceive people into performing operations or revealing secrets; therefore,  $\mathcal{T}\mathcal{M} = \{\text{Thief}, \text{Social Engineer}\}$ . Thieves manage to steal the smartphone, but they do not know the password, so they cannot compromise the protocol. Social engineers manage to know the password, but they do not possess the smartphone, so they cannot compromise the protocol. The only way they have to compromise the protocol is by combining their capabilities: together, they can both know the password and possess the smartphone, thus violating the protocol (all the padlocks are open in Table 1).

Following Example 1, once we detect *Thief + Social Engineer* as successful, any wider combination involving these attackers would be trivially successful too, as attackers colluding together result in an enrichment of the original capabilities. As a consequence, to avoid redundancy, we only consider *minimum sets of attackers* throughout our analyses: when a set of attackers is detected as successful, we never consider any larger combination involving the already detected attacker. Formally, given a non-empty set  $A_C \in \mathcal{A}_C$ , no combination  $A_C' \supseteq A_C$  will be considered.

Given a successful violation, we observe that the following two properties hold for the combinatorial analysis:

- it never reports false positives, meaning that the at-

tackers detected by the analysis do manage to violate the security goal. Therefore, the combinatorial analysis is *sound* w.r.t.  $\star_C$ ;

- it may miss some advanced attacks yielding to false negatives, thus it is *incomplete* w.r.t.  $\star_C$ .

These properties allow us to (dramatically) reduce the number of invocations of the precise and resource-intensive security analysis in the next layer.

## 2.2 Symbolic Analysis

From the combinatorial analysis, we obtain the list of all the explicit attackers violating the protocol ( $\mathcal{A}_C$ ). As a second layer of our methodology, we rely on the symbolic analysis that aims at finding:

$$\mathcal{A}_S = \{A_S \subseteq \mathcal{T}\mathcal{M} \setminus \mathcal{A}_C \mid M_{P_S}, \mu_S(A_S) \not\models_S G_S\} \quad (\star_S)$$

with

$$\mu_S : \mathcal{T}\mathcal{M} \rightarrow M_{A_S}$$

The symbolic analysis can also discover complex attacks where the attackers do not need to compromise all the authentication factors to violate the protocol, as they deceive the victim into implicitly compromising the remaining factors on their behalf; we have defined them as *implicit attacks* [10].

**Example 2.** Let us consider the protocol in Example 1. Social engineers could launch an authentication request, insert the user's password (which they do know), and send the QR code to the user, claiming that she might win a cruise by scanning it. As a consequence, although they cannot explicitly compromise all the authentication factors involved (they cannot possess the user's smartphone, as per Table 1), they anyway manage to implicitly compromise them as soon as the user scans the QR code.

By relying on the soundness of the combinatorial analysis, we employ the symbolic analysis only to evaluate the attackers who have not been detected by the combinatorial analysis (i.e.,  $\mathcal{T}\mathcal{M} \setminus \mathcal{A}_C$ ); for this reason, the combinatorial analysis shares the list of successful explicit attackers.

Given our expertise and past experience, for the symbolic analysis we have chosen to leverage ASLan++ [11] – the specification language of the AVANTSSAR Platform [6] – in combination with SATMC [9] – a model checker for security protocols relying on advanced SAT solvers. Anyway, the protocol could also be modelled in different formal specification languages and given in input to other model checkers

(e.g., ProVerif [12] or Tamarin [13]). With respect to the combinatorial analysis, here the inputs described in Section 2 have to be expressed in more refined specifications based on ASLan++. As a consequence,  $M_{P_S}$  does not represent just the authentication factors involved in the protocol, but consists in a refined model of the messages exchanged by the entities.  $G_s$  is modelled by specifying the security properties that must hold on the channel that gets established between the user and the Service Provider. Moreover, while  $M_{A_C}$  was specified in terms of compromised authentication factors,  $M_{A_S}$  is modelled in terms of channels' properties that get violated and knowledge that is acquired during the protocol. The Dolev-Yao model ( $M_{DY}$ ) [14] supported in SATMC can thus be extended by specifying custom attackers with additional capabilities (e.g., those belonging to our threat model), so that  $M_{A_S} = (M_{DY} \parallel M_{A_S}^*)$ .

**Example 3.** *From Example 2, social engineers can threaten the confidentiality between the user and the mobile application (and viceversa), as they can deceive the user into revealing whichever value she inserts in (or reads from) the application. Moreover, they can compromise the authenticity between the browser and the user, as the QR code that she scans would not really come from the browser (even though the user may think so).*

Once the model checker receives  $M_{P_S}$  and  $M_{A_S}$ , it verifies that they satisfy the security goal, reporting details about the attack in case a violation is found. We observe that the symbolic analysis is both sound and complete with respect to  $\star_s$  under suitable assumptions.

### 2.3 Risk Analysis

At the end of the combinatorial and symbolic analyses, a list of explicit ( $A_C$ ) and implicit ( $A_S$ ) attackers is generated and shared with the risk analysis layer, which evaluates the risks of each attacker as a combination between the likelihood and impact, according to some factors. Let  $\rho_L$  and  $\rho_I$  be two functions that, given an attacker in  $\mathcal{TM}$ , return tuples containing parameters that once combined together yield the likelihood or impact values, respectively; these values are defined by the security expert for each single attacker. Moreover, let  $\mathcal{R}$  be a function that computes the risk, by suitably combining the likelihood and impact. Although different methodologies can be used, we rely on an extended version of the OWASP Risk Rating Methodology [15]:  $\mathcal{R}$  computes the overall likelihood and impact by computing the average of the values returned by  $\rho_L$  and  $\rho_I$ , respectively; each of them is then assigned a label (Low, Medium, High). Finally, the likelihood and impact labels are combined according to a risk matrix (described in [15]) to obtain the overall risk.

Below, for concreteness, we consider that  $\rho_L$  returns five factors and  $\rho_I$  four factors, as specified in Table 2. We observe that the following discussion can be easily adapted to other methodologies considering different tuples of factors to characterise likelihood ( $\rho_L$ ) and impact ( $\rho_I$ ).

Depending on the considered attacker  $A$ , we now distinguish two cases:  $A$  is a single attacker, i.e.,  $|A| = 1$ ; or  $A$  is a combination of attackers, i.e.,  $|A| > 1$ .

*Single attacker:*  $A = \{a\}$

Given the set of factors that we have adopted:

$$\begin{aligned}\rho_L(A) &= \rho_L(a) = \langle v_{TD}^a, v_O^a, v_{AV}^a, v_{UI}^a, v_{SA}^a \rangle \\ \rho_I(A) &= \rho_I(a) = \langle v_{LSP}^a, v_{AS}^a, v_{AD}^a, v_{AP}^a \rangle\end{aligned}$$

where  $v_f^a = [0, 9]$  corresponds to the value assigned to the factor  $f$  for the attacker  $a$ .

Finally, the risk can be computed by combining the likelihood and impact through a suitable function  $\mathcal{R}$ :

$$Risk(A) = Risk(a) = \mathcal{R}(\rho_L(a), \rho_I(a))$$

*Combination of attackers:*  $A = \{a_1, \dots, a_n\}$

When considering a combination of attackers, each  $a_k \in A$  (with  $k = 1, \dots, n$ ) is associated with a set of likelihood and impact values:

$$\begin{aligned}\rho_L(a_k) &= \langle v_{TD}^{a_k}, v_O^{a_k}, v_{AV}^{a_k}, v_{UI}^{a_k}, v_{SA}^{a_k} \rangle \\ \rho_I(a_k) &= \langle v_{LSP}^{a_k}, v_{AS}^{a_k}, v_{AD}^{a_k}, v_{AP}^{a_k} \rangle\end{aligned}$$

As a consequence, for each likelihood and impact factor, we have  $n$  values. We now explain how to derive a single tuple for likelihood and one for impact from the  $n$  available tuples, respectively, so to apply the function  $\mathcal{R}$  to derive the risk value. For this, we define a function  $\mathcal{C}$  taking as input  $n$  values belonging to a given factor  $f$ :

$$V_f^A = \mathcal{C}(v_f^{a_1}, \dots, v_f^{a_n})$$

With respect to the case of the single attacker, the functions  $\rho_L$  and  $\rho_I$  have to be redefined accordingly:

$$\begin{aligned}\rho_L(A) &= \langle V_{TD}^A, V_O^A, V_{AV}^A, V_{UI}^A, V_{SA}^A \rangle \\ \rho_I(A) &= \langle V_{LSP}^A, V_{AS}^A, V_{AD}^A, V_{AP}^A \rangle\end{aligned}$$

Finally, the risk can be computed as follows:

$$Risk(A) = \mathcal{R}(\rho_L(A), \rho_I(A))$$

In our analysis, we have defined  $\mathcal{C}$  as follows:

- In general, we consider the minimum between all the values for the considered factor. For instance, a combination of a physical thief (which must act physically, thus  $v_{AV} = 1$ ) and a remote malware (which can act remotely, thus  $v_{AV} = 9$ ) needs a physical intervention anyway, hence the former value will be considered. Considering  $f \in \{TD, AV, UI, LSP, AS, AP\}$ :

$$\mathcal{C}(v_f^{a_1}, \dots, v_f^{a_n}) = \min\{v_f^{a_1}, \dots, v_f^{a_n}\}$$

- Beyond the previous consideration, when considering some particular factors such as the Opportunity and the Spread of Attack, the likelihood decreases as the number of attackers involved  $n$  increases. Considering  $f \in \{O, SA\}$ :

$$\mathcal{C}(v_f^{a_1}, \dots, v_f^{a_n}) = \min\{v_f^{a_1}, \dots, v_f^{a_n}\} - (n - 1)$$

- In addition, we can distinguish two classes of attackers: physical and remote (according to how they perform the attack). When attackers from both the classes are involved in a combination, they need not only to perform physical operations, but also to act remotely on the same user's devices, thus further reducing both

the Opportunity and the Spread of Attack. Considering  $f \in \{O, SA\}$ :

$$\mathcal{C}(v_f^{a_1}, \dots, v_f^{a_n}) = \min \{v_f^{a_1}, \dots, v_f^{a_n}\} - (n - 1) - 2$$

- Finally, as far as the Attack Detection is concerned, the number of physical devices stolen must be taken in consideration, since a higher number results in an easier detection of the attack. However, this consideration only applies when more than a single device is stolen. Therefore, given a combination of attackers that manage to physically steal  $k$  devices:

$$\mathcal{C}(v_{AD}^{a_1}, \dots, v_{AD}^{a_n}) = \begin{cases} \min \{v_{AD}^{a_1}, \dots, v_{AD}^{a_n}\} & \text{if } k = 0 \\ \min \{v_{AD}^{a_1}, \dots, v_{AD}^{a_n}\} - (k - 1) & \text{if } k \geq 1 \end{cases}$$

Remind that, in any case, values range from 0 to 9.

## 2.4 Relationship Between the Analyses

Our methodology involves two different layers for the security analysis (i.e., the combinatorial and symbolic analyses), which aim at identifying the explicit and implicit attackers that are able to compromise the considered protocol. The list of successful attackers is then shared with the risk analysis layer, which assigns a risk value to each of the attackers detected by the previous analyses.

The combinatorial and symbolic analyses are connected by a strong relationship that must be taken in consideration when providing the related models:  $M_{AS}$  extends  $M_{AC}$ , i.e.,  $M_{AS}$  models in the specification language at least the same capabilities on the authentication factors represented by  $M_{AC}$ . To ensure consistency between the two analyses, these models should be provided by a security expert.

Due to this relationship, the combinatorial analysis is not strictly necessary, as the symbolic analysis would be able to detect both explicit and implicit attackers by itself. In particular, for each attacker  $A_C$  detected by the combinatorial analysis, there exists a corresponding attacker  $A_S$  detected by the symbolic analysis, with  $A_C = A_S$ ; therefore, the combinatorial analysis is *sound with respect to the symbolic analysis*. Consequently, we rely on the combinatorial analysis to reduce the set of attackers to test in the symbolic analysis, so as to reduce the overall complexity of the process.

On the contrary, the symbolic analysis is mandatory: for each attacker  $A_S$ , there not always exist a corresponding attacker  $A_C$ , with  $A_S = A_C$ ; therefore, the combinatorial analysis is *incomplete with respect to the symbolic analysis*.

## 2.5 Computational Considerations

The symbolic analysis requires advanced computational capabilities, which result in a higher time of execution. On the other hand, it provides a higher level of confidence on the results, meaning that it reports also complex attacks that may have been missed during the combinatorial analysis. If we performed the symbolic analysis for all the  $n$  attackers (and combinations) belonging to the threat model, we would have to run it  $2^n - 1$  times. We can considerably reduce the set of attackers to test thanks to the following considerations:

**C1. Explicit attacks from the combinatorial analysis:** the main goal of the combinatorial analysis is to prune the set of attackers to test, given its speed in detecting explicit attackers. Consequently, by relying on its soundness, we check neither already detected attackers nor larger combinations involving sets of successful attackers.

**C2. Physical thieves:** when they steal a device, users cannot be deceived into approving a malicious authentication attempt on that device, because they no longer own it. Therefore, these attackers cannot be involved in implicit attacks.

**C3. Non-minimum combinations of implicit attacks:** once the symbolic analysis detects successful attackers, we do not consider further combinations with them (see Section 2.1 for a formal definition).

More details will be provided in Section 4.5.

## 2.6 Application Scenarios

In the next section, we describe a concrete authentication protocol to which we apply our methodology. This use case scenario just aims to provide a concrete example of how our methodology can be applied and how the results can be used to support the design phase. In fact, our methodology can be employed to analyse authentication protocols in many more scenarios. For example, in the financial sector, to highlight the differences between online banking authentication protocols before and after the PSD2 regulation on payment services [16]; in the corporate sector, to support the design of new authentication schemes or analyse existing ones according to internal needs.

## 3 USE CASE SCENARIO

We have a long-standing collaboration with *Poligrafico e Zecca dello Stato Italiano* (IPZS), the Italian Government Printing Office and Mint that is responsible for producing the Italian eID card, called *Carta d'Identità Elettronica* (CIE 3.0) [17]. This collaboration aims at developing and analysing cutting-edge identity management solutions to fully benefit from the capabilities of eID cards, which are equipped with a microchip that communicates through a contactless NFC interface. They are also provided with an X.509 certificate [18] containing the personal data of the owner, whose trustworthiness is ensured by the competent authorities through a digital signature. As a consequence, each card has a pair of keys that can be used for public-key cryptography (following the IAS ECC specifications [19]), whose use can be unlocked by inserting a PIN.

Involving eID cards in an authentication workflow may provide several benefits from a security perspective. In this context, we focused on *hybrid* scenarios allowing users to authenticate from a personal computer by leveraging their smartphone as an eID card reader. In particular, we now describe a concrete protocol that is currently used in the Italian ecosystem as one of the main authentication procedures to access Public Administration's online services. This protocol requires users who wish to authenticate themselves to launch a request from their personal computer's browser, which then displays a QR code. Once users scan the QR code

TABLE 2  
Factors considered for the Risk Analysis

Likelihood		
Technical Difficulty	TD	Deals with the technical difficulty to perform the attack from a technical perspective, with lower scores representing higher levels of difficulty. For instance, reading a password from a screen is not technically difficult (higher value), in contrast to programming a rooting malware (lower value).
Opportunity	O	Time or space restrictions limiting the attackers' opportunity to carry the attack out, with more restrictions resulting in lower scores. For instance, assuming that mobile devices are protected by security mechanisms, thieves need to steal them when they are unlocked, otherwise they become unusable (lower value). On the other hand, deceiving users into revealing their password is not subject to particular limitations (higher value).
Attack Vector	AV	The vector which the attack is carried out through. Attacks requiring a physical intervention result in a rating of 1, while those requiring only a network access result in 7.
User Interaction needed	UI	Whether the victims need to interact in order to make the attack successful, with lower scores (1 or 2) assigned when an interaction is needed in a specific interval of time, 4 when an interaction is needed with no time constraints, 7 when the attack can be carried out without any interaction with the users. For instance, in case the victim needs to authorize the attacker's malicious attempt in a few minutes (before the operation expires), this results in a lower score.
Spread of Attack	SA	Provides a measure of the popularity of an attack, based on official statistics concerning the specific (combination of) attackers analysed. Common attacks result in higher scores.
Impact		
Loss of Security Properties	LSP	Refers to the effects of the attack in terms of violated security properties. While OWASP suggests four different factors to estimate the loss of confidentiality, integrity, availability and accountability respectively, we chose to consider them jointly. As our protocols aim at authenticating users onto an external SP, we are not aware of the impact of a potential violation on the security properties, therefore we adopt a worst-case approach and always rate this factor with 9.
Attack Scale	AS	Depends on the number of users which could potentially be compromised, with more users leading to higher values. In particular, we assign 2 when only a specific target can be threaten, while we use 8 when everyone can be affected.
Attack Detection	AD	Whether (and how easily) the attack can be detected by the victims after it has been successfully carried out, with easy-to-detect attacks resulting in lower scores. For instance, if every successful authentication cause a notification to be sent to the legitimate user, a potential account violation will be discovered easily, thus resulting in lower scores.
Attack Persistence	AP	Provides a measure of the time exposure of the user's account in case it was compromised. For instance, if an attacker manages to steal the OTP or the response connected with the ongoing attempt, he can compromise only that specific authentication; on the other hand, if he steals the eID card and its PIN he can authenticate on the user's behalf until the theft is reported and the eID card blocked. In the former case we rate this factor with 2, while in the latter with 8.

via a custom mobile application (namely, eIDApp) on their smartphone, they are guided through the interaction with their eID card and finally authenticated on their personal computer. Therefore, the authentication factors involved are the eID card (📱) and the PIN (🔑).

### 3.1 Entities

The protocol involves the following set of entities:

- *User*: the claimant, who wants to authenticate on a specific service. Each user is supposed to have a `userId`, which is a uniquely identifying value that is contained in the eID card's certificate.<sup>2</sup>
- *eID card*: the eID card belonging to the user.
- *eIDApp*: the mobile application responsible for securely interacting with the eID card.
- *Identity Provider (IdP)*: responsible for managing users' online identities and ensuring proper authentication. It is composed of a front-end interface (accessible through a browser) and a back-end server (*IdPServer*).

2. In the Italian scenario, the serial number of the eID card is used as `userId`.

- *Service Provider (SP)*: any online service where users can authenticate. It is composed of a front-end interface (accessible through a browser) and a back-end server (*SPServer*).
- *Browser*: a web browser that users can interact with in order to access front-end interfaces of both the SP and the IdP. This browser runs on a personal computer belonging to the user.

The *IdPServer* and the *SPServer* are part of a trust relationship, obtained after a federation procedure [20].

### 3.2 Flow

Fig. 2 shows the message sequence chart of the protocol, which is composed of the following phases:

- 1) *Authentication request*: the user visits the SP webpage and launches a new authentication request, thus being redirected to the IdP. The *IdPServer* generates some fresh values associated with the authentication attempt: an identifier `opId` and an associated cookie `IdP-SessionCookie` that will be used during any communication between the browser and the *IdPServer*. After storing these values, the *IdPServer* displays the login page.



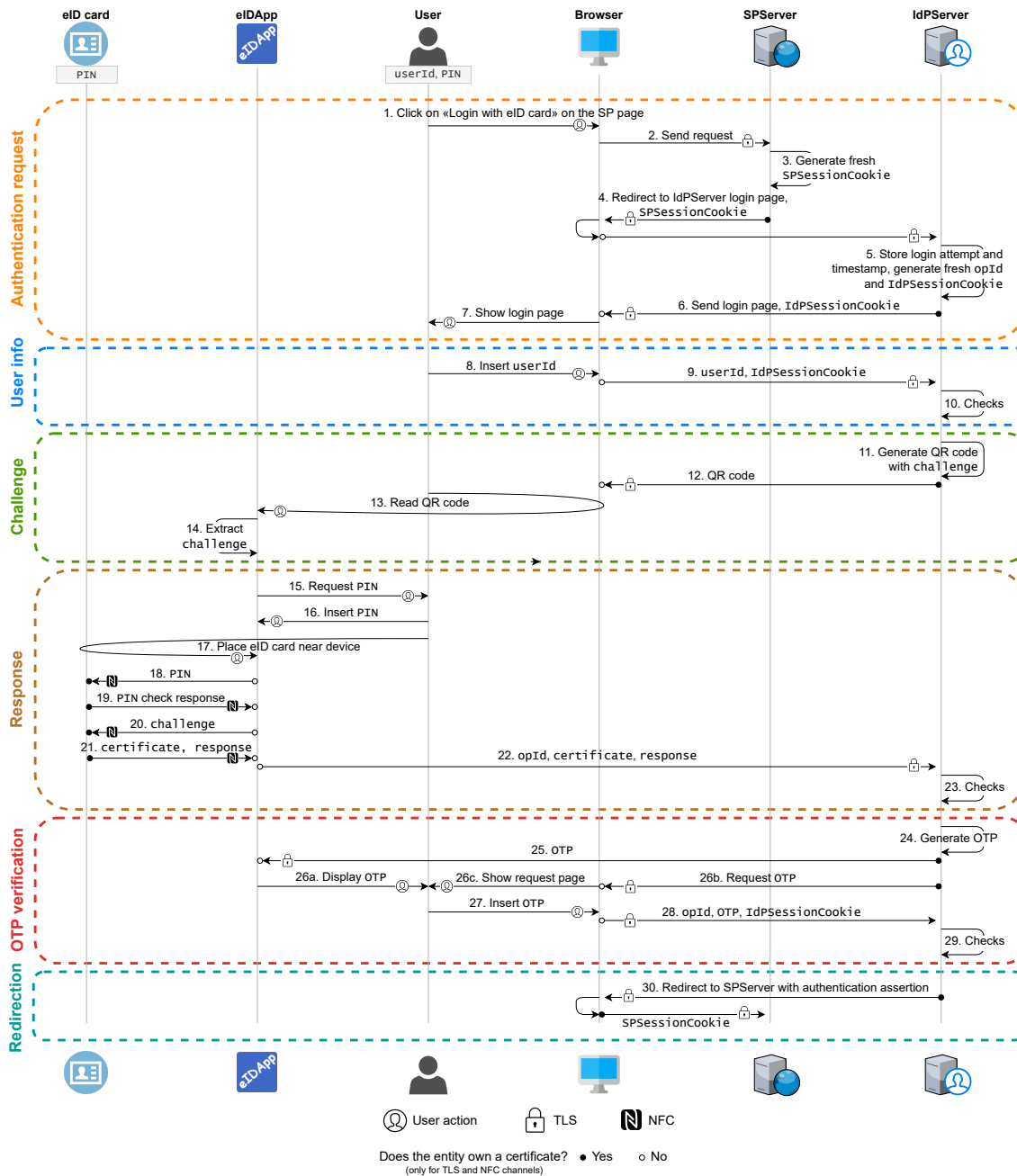


Fig. 2. Message Sequence Chart of the protocol

- 2) *User info*: the user fills her identifying information (*userId*) in the login page. The browser retrieves this value and sends it to the IdP Server.
- 3) *Challenge*: the IdP Server generates the challenge and displays it on the browser as a QR code, which the user is required to scan through the eIDApp to extract the plain challenge.
- 4) *Response*: after providing the PIN, the user is required to place her eID card near the mobile device for NFC scanning. In case the PIN is correct and has been provided within a fixed number of attempts, the eID card signs the challenge through its private key, thus generating the response that is finally sent back to the IdP Server through the eIDApp along with the eID

- card's certificate.
- 5) *OTP verification*: the IdP Server generates a fresh OTP and associates it with the current *opId*. The OTP, which is displayed to the user through the eIDApp, needs to be written back to the IdP login page on the browser. Finally, it is sent to the IdP Server.
- 6) *Redirection*: the IdP Server redirects the user to the SP Server with an authentication assertion.

During the protocol, the IdP Server verifies that:

- *Step 10*: the incoming *IdPSessionCookie* matches the one generated at *step 5*.
- *Step 23*: (i) the eID card's certificate has not been revoked; (ii) the *userId* previously inserted matches the one stored in the eID card's certificate; (iii) the value

obtained after applying the eID card's public key to response matches the challenge.

- *Step 29*: (i) the incoming `IdPSessionCookie` matches the one generated at *step 5*; (ii) the OTP inserted by the user on the browser at *step 27* matches the one generated by the `IdPServer` at *step 24*; (iii) the OTP has been provided within a fixed number of attempts; (iv) the operation has been completed in a fixed time interval.

In case one of these checks fails, the authentication procedure ends with an error.

### 3.3 Challenge

During authentication, the `IdPServer` needs to be sure that the involved eID card really belongs to the user who is going to be authenticated. To this end, a challenge–response procedure [1] occurs between the `IdPServer` and the eID card. In the considered protocol, the `challenge` is composed of the following parameters:

- *opId*: the operation identifier, which is randomly generated and represents the authentication attempt;
- *userId*: the user identifier, which the user fills in during the procedure;
- *IdPName*: the name of the IdP used for the authentication;
- *SPName*: the name of the SP which the user wishes to authenticate onto;
- *opText*: a textual description of the current operation, displayed on the mobile device before the approval.

## 4 APPLICATION OF THE METHODOLOGY

In this section, we show how we have applied our methodology to the use case scenario defined in Section 3.

### 4.1 Security Setup

In this section, we set the analyses up for our use case scenario by defining the security assumptions (Section 4.1.1) and the attackers' capabilities (Section 4.1.2).

#### 4.1.1 Security Assumptions

During the analyses, we consider the security assumptions that are described in Table 3. We categorize them in: *eID cards Assumptions* (EA), ensuring that the authenticator has been properly activated; *Procedural Assumptions* (PA), dealing with the authentication procedure itself; *Trust Assumptions* (TA), related to the trust between the entities involved in the protocol; and *Channels Assumptions* (CA), regarding the properties of the communication channels.

#### 4.1.2 Threat Model and Attackers' Capabilities

In order to analyze the security of the protocol, we have identified from [8] the *Authenticator Threats* that can violate it. Then, we have expanded and contextualized the related attackers to obtain the threat model in Table 4:

$$\mathcal{TM} = \{PCT, MDT, CT, D, ES, SS, SE, MB, MM\}$$

The relationship between the *Authentication Threats* identified by NIST and our threat model is detailed in the

complementary website [21]. In particular, we have not considered: “Assertion Manufacture or Modification” (since the authentication assertion is digitally signed by the `IdPServer` and cannot be tampered with); “Offline Cracking” and “Online Guessing” (due to the restricted number of possible attempts while inserting the eID card's PIN); “Side Channel Attack” (as in EA4 we assume that the eID card's private key is particularly difficult to extract); and “Unauthorized Binding” (as eID cards can be associated only to their legitimate owners, due to EA1).

Once defined the attackers, we have also defined their capabilities in terms of compromised authentication factor(s) in our scenarios (Table 5): closed padlocks (🔒) denote non-compromised factors, while open padlocks (🔓) represent compromised factors. In addition, we use an asterisk (🔓\*) to indicate a possession factor that is compromised *indirectly*. For instance, a malicious application does not physically violate eID cards, yet it manages to deceive victims into interacting with their eID cards, thus compromising that factor without physically possessing it.

### 4.2 Combinatorial Analysis

The combinatorial analysis, described in Section 2.1, discovered 5 attackers that are able to compromise the protocol explicitly:

$$\mathcal{A}_C = \{\{MM\}, \{CT, D\}, \{CT, ES\}, \{CT, SS\}, \{CT, SE\}\}$$

where the first corresponds to a malicious application, while the others correspond to a card thief combined with another attacker able to discover the user's eID card's PIN.

For instance, considering the combination CT+SS: CT manages to obtain the victim's eID card (🔓) by physically stealing it, while SS can compromise the PIN of the eID card (🔓) by looking at the victim while typing it. As a consequence, the combination violates all the authentication factors involved in the QR protocol and thus is able to authenticate onto an online service as the victim.

### 4.3 Symbolic Analysis

As explained in Section 2.2, we need to provide SATMC with all the parameters in input, which have been modelled in ASLan++ following the work in [22]. To better understand the formalisation, Table 6 shows some relevant predicates in ASLan++. The complete ASLan++ models are available on the complementary website [21].

#### 4.3.1 Protocol ( $M_{P_S}$ )

The model of the protocol formally describes the entities involved and the communications between them over the communication channels. This model should be as consistent as possible with the protocol it describes, in order to obtain a proper analysis; however, as often happens, models can also contain some approximations to reduce the computational complexity of the analysis, though without losing crucial details. For space reasons, the approximations adopted in our models are described on the complementary website [21].

Legitimate entities may be required to know specific information before the protocol execution. ASLan++ requires these values to be passed as arguments to the related entities. In the considered protocol:













TABLE 3  
Security Assumptions




#	Assumptions
EA1	eID cards are released by a specific municipality to their right owners, after a proper identification.
EA2	The eID cards' PIN has not been disclosed before being given to the citizen.
EA3	eID cards are provided with anti-tampering features, thus they cannot be altered or cloned.
EA4	The private keys of eID cards, related to their $x.509$ certificate, are safely protected and cannot be disclosed.
PA1	Users cannot have their physical devices (i.e., personal computer, mobile device or eID card) stolen during the authentication process. Attackers can hence possess such devices either for the whole protocol or not at all.
TA1	The IdPServer is recognized as an official IdP, thus releasing only valid and correct identity assertions and protecting secret values.
CA1	The browser and the IdPServer communicate over a unilateral TLS channel, established thanks to a valid certificate of the latter.
CA2	The browser and the SPSServer communicate over a unilateral TLS channel, established thanks to a valid certificate of the latter.
CA3	The eIDApp and the IdPServer communicate over a unilateral TLS channel, established thanks to a valid certificate of the latter.
CA4	The eIDApp and the eID card occurs over a unilateral secure channel, established via NFC thanks to a valid certificate of the latter.

TABLE 4  
Attackers belonging to the Threat Model

Personal Computer Thief	PCT	Steals the users' personal computer.
Mobile Device Thief	MDT	Steals the users' mobile device.
Card Thief	CT	Steals the users' eID card.
Duplicator	D	Copies or duplicates a legitimate authenticator or authentication factor.
Eavesdropping Software	ES	Intercepts the data exchanged between the claimant and the authenticator. Keyloggers are typical examples of this kind of attacks, since they either capture what users type or take screenshot while passwords are being filled in.
Shoulder Surfer	SS	Obtains secrets by physically looking at the claimant performing authentication operations. This attack can be implemented either by directly looking over the claimant's shoulder or by using longer-range tools like binoculars, video surveillance systems or hidden cameras.
Social Engineer	SE	Exploits human gullibility and confidence in others. Attackers deceive the claimant into revealing secret information or performing actions to their advantage.
Man in the Browser	MB	Manages to take full control of the claimant's browser, due to malware or malicious browser extensions which have previously been installed. By lying on the user's browser, these applications are thus able to read and tamper with every webpage and transaction, without the claimant's awareness.
Man in the Mobile	MM	As the previous attacker, but related to mobile devices. Whether malicious programs are able to obtain root privileges, they totally compromise the device: they are therefore able to read in other applications' sandboxes, overlay the interface with custom phishing windows and a lot more.

TABLE 5  
Attackers' capabilities in the combinatorial model

Attackers	Authentication Factors	
		
PCT, MDT, MB		
CT		
D, ES, SS, SE		
MM		

 = safe     = compromised     = indirectly compromised

- the user knows the PIN of the eID card and the `userId`;
- the eID card knows its PIN.

In addition to honest entities, also attackers (which in ASLan++ are referred to as *intruders* and indicated with *i*) may have some preliminary knowledge. In ASLan++, an initial knowledge  $k$  is given to the intruder by using

the expression  $i\text{knows}(k)$ . However, since attackers' initial knowledge depends on their capabilities, it will be specified directly in the attackers' model  $M_{AS}^*$ .

Beyond modelling the protocols, we also had to formalise the security assumptions that we have identified in Table 3. The formal specification of such assumptions can be found in Table 7, while the description of the predicates used is provided in Table 6.

#### 4.3.2 Model of the attackers' capabilities ( $M_{AS}^*$ )

To model the attackers' capabilities for the symbolic analysis, we modify some parts of the specifications according to the attacker that we are considering in the specific run of the symbolic analysis. Below you can find a description of how each attacker affects the protocol; the numbers in parentheses refer to Table 8, which displays the corresponding changes to make in the ASLan++ model. We also show the relationship between the symbolic and the combinatorial models of the attackers' capabilities, to underline that the

TABLE 6  
Relevant predicates in ASLan++

Predicate	Description
<code>weakly_authentic(ch)</code>	A channel <code>ch</code> is <i>weakly authentic</i> if its input can be accessed by a single, but unknown, sender.
<code>weakly_confidential(ch)</code>	A channel <code>ch</code> is <i>weakly confidential</i> if its output can be accessed by a single, but unknown, recipient.
<code>confidential_to(ch, R)</code>	A channel <code>ch</code> is <i>confidential</i> if its output can be accessed only by the specified recipient <code>R</code> . Therefore, this predicate is a strengthened version of <i>weakly confidential</i> .
<code>link(ch_A2B, ch_B2A)</code>	Two channels <code>ch_A2B</code> and <code>ch_B2A</code> are <i>linked</i> if the agent sending messages over the former is the same receiving messages on the latter.
<code>unilateral_conf_auth(ch_A2B, ch_B2A, B)</code>	The property of <i>unilateral confidentiality and authenticity</i> can be used to model a run of SSL/TLS in which an agent has a valid certificate, while the other has not. Specifically, this property implies that: <code>ch_A2B</code> is <i>confidential</i> to <code>B</code> and <i>weakly authentic</i> ; <code>ch_B2A</code> is <i>weakly confidential</i> and <i>authentic</i> for <code>B</code> ; <code>ch_A2B</code> and <code>ch_B2A</code> are linked.
<code>bilateral_conf_auth(ch_A2B, ch_B2A, A, B)</code>	The property of <i>bilateral confidentiality and authenticity</i> can be used to model a run of SSL/TLS in which both agents have a valid certificate. Specifically, this property implies that: <code>ch_A2B</code> is <i>confidential</i> to <code>B</code> and <i>authentic</i> for <code>A</code> ; <code>ch_B2A</code> is <i>confidential</i> to <code>A</code> and <i>authentic</i> for <code>B</code> ; <code>ch_A2B</code> and <code>ch_B2A</code> are linked.

TABLE 7  
Security assumptions in the symbolic model

Ass.	Formal Specification
EA1	Unless CT is considered, users possess their own eID card that will be used for proper identification.
EA2	The PIN of the eID card is set as <code>nonpublic</code> .
EA3	No attacker can tamper with an eID card.
EA4	Private keys are automatically kept confidential unless otherwise declared.
PA1	<code>link(ch_User2Browser, Ch_Browser2User)</code> <code>link(ch_User2EICApp, Ch_EICApp2User)</code>
TA1	The intruder <code>i</code> cannot impersonate the <code>IdPServer</code> in any session.
CA1	<code>unilateral_conf_auth(ch_Browser2IdPServer, ch_IdPServer2Browser, IdPServer)</code>
CA2	<code>unilateral_conf_auth(ch_Browser2SPServer, ch_SPServerS2Browser, SPServer)</code>
CA3	<code>unilateral_conf_auth(ch_EICApp2IdPServer, ch_IdPServer2EICApp, IdPServer)</code>
CA4	<code>unilateral_conf_auth(ch_EICApp2EIC, ch_EIC2EICApp, EIC)</code>

former ( $M_{As}$ ) can always be reduced to the latter ( $M_{AC}$ ); this leads to the discussion in Section 2.4.

It is important to notice that we provide instructions to model both how to consider and how *not* to consider a specific attacker. Therefore, a dash (—) means that the predicate in the other column of the same row does not apply.

**PCT, MDT, CT** Before the attacker steals the user's device (personal computer, mobile device or eID card), every interaction with it is surely made by the user herself, thus the channel between the user and the browser (1), the eIDApp (5) or the eID card (9) is *authentic*. Moreover, the fact `userOwnComputer` (2), `userOwnSmartphone` (6) or `userOwnEIC` (10) needs to be accordingly set to `true`, since the user physically owns its device.

On the contrary, when the user's device is possessed by the attacker, all the interactions are made by the same entity (the attacker himself). Therefore:

- since the browser (3), the eIDApp (7) and the eID card (11), respectively, do not have any guarantee on this entity's identity, the related channel is only *weakly authentic*;

- since the attacker is the only entity who can interact with the browser (4), the eIDApp (8) or the eID card (12), he uses the same communication channel in every session.

This corresponds to violating the personal computer, mobile device, or eID card, respectively. However, in our protocol, only the eID card is considered as an authentication factor ( $PCT, MDT \xrightarrow{\text{eID}} \emptyset; CT \xrightarrow{\text{eID}} \text{eID}$ ).

**D** The attacker manages to copy the eID card's PIN that may be written on paper ( $D \xrightarrow{\text{eID}} \text{eID}$ ), thus getting to know this value (13).

**ES, SS** Without considering these attackers, the following information can be known only to the intended recipient, thus the considered communication channels are *confidential*:

- what the user types in the eIDApp/browser can be known only by the eIDApp (14)/browser (15);
- what the browser shows to the user can be known only by the user (16).

Property (15) corresponds to violating the PIN, which the user is required to insert in the eIDApp ( $ES, SS \xrightarrow{\text{eID}} \text{eID}$ ). Instead, properties (15) and (16) are

TABLE 8  
Attackers' capabilities in the symbolic model

Att.	Formal Specification	
	Without Attacker	With Attacker
PCT	(1) <code>authentic_on(Ch_U2B, User);</code> (2) <code>userOwnComputer;</code> — —	(3) <code>weakly_authentic(Ch_U2B);</code> (4) Use same channel <code>ch_U2B</code> in sessions
	(5) <code>authentic_on(Ch_U2EICApp, User);</code> (6) <code>userOwnSmartphone;</code> — —	(7) <code>weakly_authentic(Ch_U2EICApp);</code> (8) Use same channel <code>ch_U2EICApp</code> in sessions
CT	(9) <code>authentic_on(Ch_U2EIC, User);</code> (10) <code>userOwnEIC;</code> — —	(11) <code>weakly_authentic(Ch_U2EIC);</code> (12) Use same channel <code>ch_U2EIC</code> in sessions
	D	(13) <code>iknows(PIN);</code>
ES, SS	(14) <code>confidential_to(Ch_U2EICApp, EICApp);</code> (15) <code>confidential_to(Ch_U2B, Browser);</code> (16) <code>confidential_to(Ch_EICApp2U, User);</code>	— — —
	SE	(17) <code>confidential_to(Ch_U2B, Browser);</code> (18) <code>authentic_on(Ch_B2U, Browser);</code> — —
MB		(21) Replace <code>browser</code> with <code>i</code> in one session
MM	(22) <code>authentic_on(Ch_EICApp2U, EICApp);</code> (23) <code>authentic_on(Ch_EICApp2EIC, EICApp);</code> —	— — (24) Replace <code>eicapp</code> with <code>i</code> in one session

needed because users are displayed an OTP on the mobile application and have to insert this OTP on the personal computer's browser, thus the corresponding channels need to be confidential when the attacker is not considered.

All the properties above are no longer true when the attackers become effective, since they manage to intercept all these values.

**SE** Without considering this attacker, the communications between the user and the browser are protected by the following properties:

- what the user types in the browser can be known only by the browser, thus the channel is *confidential* (17);
- the user is sure that whatever the browser shows to her really comes from the original browser, thus the channel is *authentic* (18).

Properties (17) and (18) are needed because users are displayed a QR code on the personal computer's browser and at the end of the operation they have to insert an OTP on the personal computer's browser, thus the corresponding channels need to be confidential and authentic, respectively, when the attacker is not considered.

When SE becomes effective, instead, these properties are no longer valid: the attacker can both deceive users into revealing what they typed in the browser, and provide them with some malicious values by pretending to be the browser (e.g., the QR code containing the challenge). Moreover, the attacker manages to make the user reveal the PIN of the eID card (19) and the

OTP generated by the eIDApp (20), thus compromising these values ( $SE \xrightarrow{\text{eIDApp}} \text{[OTP]}$ ).

**MB** This attacker can take full control of the user's browser and perform any operation he wishes, thus we model MB by making the attacker impersonate the browser (21). However, it does not violate any authentication factor in our protocol ( $MB \xrightarrow{\text{browser}} \emptyset$ ).

**MM** This attacker can take full control of the user's mobile device and perform any operation he wishes. Therefore, we model MM by explicitly making the attacker impersonate the eIDApp (24). However, we need to restrict other attackers' capabilities when the MM is *not* to be considered, otherwise they could be too powerful than how we have really modelled them. To this end, when MM is not considered:

- the user is sure that whatever the eIDApp shows to her really comes from the original eIDApp, thus the channel is *authentic* (22);
- the eID card is sure that whatever the eIDApp sends to her really comes from the original eIDApp, thus the channel is *authentic* (23).

When considering MM, there is no need to remove instructions (22) and (23), as the channels do remain either authentic or confidential. However, by impersonating the eIDApp due to (24), MM can deceive the user into interacting with her eID card ( $MM \xrightarrow{\text{eIDApp}} \text{[eIDCard]}$ ), as well as know the PIN when the user types it ( $MM \xrightarrow{\text{PIN}} \text{[PIN]}$ ).

#### 4.3.3 Security goal ( $G_s$ )

In ASLan++, a channel goal has the following form:

name: ( ) Sender Channel Receiver

We can rely on this syntax to model the security goal (identified in Section 2) as follows:

User\_authn\_to\_SP: ( ) User \*->> SPServer;

This represents a goal called *User\_authn\_to\_SP* that must be satisfied in the run(s) of the protocol. Specifically, the goal requires that a communication channel between the user (sender) and the SPServer (receiver) gets established at the end of the protocol; on this channel, the following properties (represented by \*->>) must hold:

- *authenticity*: guarantees that any incoming message on this channel indeed comes from the user;
- *directedness*: guarantees that any incoming message on this channel was indeed intended for the SPServer;
- *freshness* (or *replay protection*): guarantees that any message sent on this channel can be received only once.

#### 4.3.4 Number of sessions

For each analysis, we run two parallel sessions of the protocol. This way, we can evaluate an attacker leveraging a parallel session launched by the users themselves to finalise the attack, which represents how implicit attacks are usually performed. The two sessions share the same inputs, though the communication channels used are different (except when modelling some attackers, as detailed above).

#### 4.3.5 Results

The symbolic analysis tested the following attackers: D, ES, SS, SE, MB, D+ES, D+SS, ES+SS, D+ES+SS (see Section 4.5 for more details).and detected two attackers:

$$\mathcal{A}_s = \{\{SE\}, \{MB\}\}$$

For both of them, the analysis reported the attack trace,<sup>3</sup> which is a graphical representation of the messages exchanged between the entities taking part in the protocol, by using arrows labelled with the content of the message. Messages consisting in the concatenation of more values are joined by a dot, while fresh values  $f$  are represented by the expression  $n(f)$ . In addition to the entities taking part in the protocol, attack traces usually display an additional entity  $i$  representing the intruder (i.e., the attacker). In case the attacker impersonates another entity  $e$ , this fact is represented by the expression  $i(e)$  placed inside a box.

Considering MB, the attack trace in Fig. 3 shows that when the victim tries to authenticate on an SP (through `request1`) and inserts her `userId`, the attacker can initiate an authentication process on the same SP by using the same `userId` and obtain a challenge in the form of a QR code. Following the authentication process, the victim should be displayed a QR code as well, but – since the attacker has full control of the victim’s browser – he can tamper with the victim’s authentication page and replace the original QR code with that obtained in his parallel authentication session. The user will be deceived into scanning the malicious QR code with the eIDApp, inserting the PIN and reading the eID card through NFC. At the end, she will be displayed

3. To enhance readability, the attack traces displayed in this paper are a simplified version of those generated by SATMC.

an OTP on the eIDApp to insert on her personal computer’s browser; however, having control of the browser, MB can intercept the OTP and insert it in his own authentication page. As a result, the attacker will be authenticated on his own personal computer with the victim’s credentials, as the QR code scanned by the user had been originally issued in the context of the attacker’s authentication session.

The second implicit attack, performed by SE, is similar to the previous one. However, SE does not need to alter the victim’s browser, since he can provide her with the QR code through other means (e.g., via email or a social media) and deceive her into scanning it through social engineering techniques (e.g., «Scan the QR code and use your eID card to win a wonderful cruise!»). Then, the attacker can – again – deceive the victim into revealing the OTP, so that he can finalise the authentication by impersonating the user.

#### 4.4 Risk Analysis

Table 9 shows the results of the risk analysis applied to our use case scenario, where we can identify three attackers associated with a *low* risk (MM, CT+ES and CT+SE), three with a *medium* risk (CT+D, CT+SS and SE) and one with a *high* risk (MB). MB is a powerful attacker that requires a specific technical preparation to infect the browser, can be performed fully remotely, does not need any interaction with the user, is difficult to detect, and can perform large-scale attacks.

#### 4.5 Computational Considerations

In Section 2.5 we have proposed some considerations to reduce the computational complexity of the symbolic analysis. Table 10 displays the number of attackers that our considerations exempted us from testing in the use case. When applied in sequence, each consideration further improves the set of attackers excluded by the previous ones. As a result, the symbolic analysis needed to test only 9 attackers in our use case scenario (1.8%), thus considerably optimising the analysis flow.

### 5 SECURITY MITIGATIONS

The role of security mitigations is extremely important to reduce risks: they play a fundamental role in shortening the list of successful attackers or reducing the likelihood and/or impact of certain attackers. As a consequence, the selection of which mitigations are worth implementing is a crucial phase during protocol design. However, companies or governmental agencies implementing authentication procedures could have custom requirements; that imposes trade-offs between usability and security when choosing mitigations. Table 11 displays the mitigations that are implemented in our use case scenario. In addition to security, we focus on usability that is one of the most important dimensions to consider when selecting mitigations, as users are more willing to accept simple protocols rather than cumbersome ones. We now discuss each mitigation along with their effects.

**M1** Since rooted devices are known to be extremely vulnerable to common attacks, preventing the use of the eIDApp on such devices brings many advantages

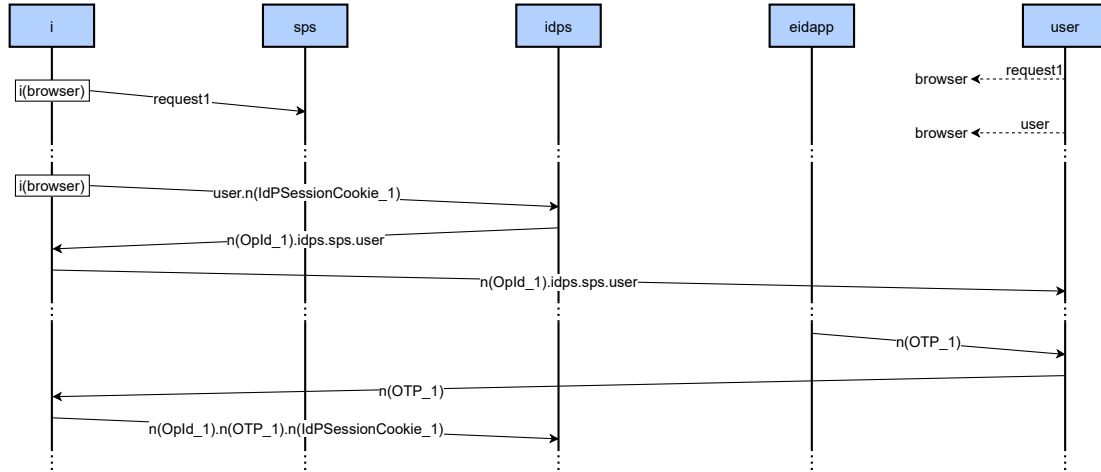


Fig. 3. Trace of the implicit attack performed by MB

TABLE 9  
Results of the risk analysis

Attackers	Likelihood							Impact					Risk	
	TD	O	AV	UI	SA	Overall	LSP	AS	AD	AP	Overall			
MM	2	2	7	1	2	2.80	Low	9	8	3	2	5.50	Medium	Low
CT+D	8	1	1	7	4	4.20	Medium	9	2	3	8	5.50	Medium	Medium
CT+ES	5	0	1	4	2	2.40	Low	9	2	3	8	5.50	Medium	Low
CT+SS	8	4	1	2	5	4.00	Medium	9	2	3	8	5.50	Medium	Medium
CT+SE	4	2	1	4	3	2.80	Low	9	2	3	8	5.50	Medium	Low
SE	4	9	7	1	4	5.00	Medium	9	5	7	2	5.75	Medium	Medium
MB	3	5	7	1	4	4.00	Medium	9	8	7	2	6.50	High	High

TABLE 10  
Computational effects of our considerations during our analyses

Considerations	Attackers not to be tested in the symbolic analysis (out of $2^9 - 1 = 511$ )
C1	376 (73.6%)
C2	104 (20.3%)
C3	22 (4.3%)
<b>Total</b>	<b>502 (98.2%)</b>

in terms of security. On the other hand, it may result in usability issues since people who rooted their devices on purpose would not manage to use the application.

**M2** Restricts the attack surface: attackers cannot just send malicious QR codes to random people, but they need to choose a precise victim as the `userId` is part of the challenge. Although we do not require the `userId` to be a secret value (like a password), it should not be commonly known (like the name of the user) to reduce the possibility of a general attack. Given the additional value users have to insert, this mitigation slightly reduces usability.

**M3** Restricts the ability of attackers to deceive users by sending improper QR codes, since the operation must be completed within a certain time interval. This mitigation partially affects usability as well, consider-

ing that expired authentication attempts have to be launched again.

**M4** By restricting the possible attempts, prevents guessing and brute-force attacks on secret values. The number of available attempt should be carefully set in order to find a trade-off between security and usability (i.e., neither too tight, nor too loose).

**M5** Restricts the possibility of an attack, since the attacker would also need to obtain the OTP associated with that specific authentication attempt. Specifically, we have decided to display the OTP on the mobile application in order to reduce phishing attacks carried out via email: in case users do not have an authentication attempt currently ongoing on the personal computer's browser, they would not know where to insert the OTP. Given the additional value users have to insert, this mitigation slightly reduces usability.

**M6** Reminds users of verifying that QR codes are displayed on an official website, thus reducing the likelihood of attackers sending improper QR codes on phishing websites (such as SE and MB). However, since the warning is always displayed within the mobile application, users could ignore or get used to it. This mitigation has no impact on usability, as it does not restrict users' attempt nor it requires users to perform additional operations.

**M7** Helps the user distinguish between legitimate and malicious authentication attempts. This mitigation is

TABLE 11  
List of possible mitigations

#	Mitigations	Security	Usability
M1	Implement root detection mechanisms on the eIDApp in order to prevent its use on rooted devices.	●●●○	●○○○
M2	Require the input of a uniquely identifying information (namely, the <code>userId</code> ) during authentication.	●●●○	●●○○
M3	Restrict the validity of authentication attempts only to a certain interval of time, thus rejecting those completed after the fixed threshold.	●●○○	●●●○
M4	Restrict the number of possible attempts to provide the correct secret values (i.e., eID cards's PINs and OTPs).	●●○○	●●●○
M5	At the end of the authentication procedure, display an OTP on the mobile device and require users to insert it in the personal computer's browser.	●●○○	●●○○
M6	During potentially dangerous operations (e.g., reading the QR code through the mobile application), advise users to verify the trustworthiness of the source, for instance by checking that the connection is protected through TLS or by verifying that the URL really belongs to the <code>IdPServer</code> .	●○○○	●●●●
M7	During the authentication process, always inform users of the ongoing operation.	●●○○	●●●●

○○○○ = minimum    ●●●● = maximum

extremely effective in authorization contexts, as precise details about the ongoing operation to authorize are displayed to the user. When dealing with authentication, instead, it is difficult to find suitable details to uniquely identify the ongoing attempt, thus the security benefits are slightly lower. In general, the information displayed to the user should be relevant and uniquely identify the operation, otherwise some attacks could anyway be performed. The information can be shown either without affecting the procedure (e.g., in the same window where the user inserts the PIN) or by introducing an *ad hoc* activity, which would slightly affect usability. When rating usability in Table 11, we consider the former case.

### What if...?

In Section 4, we have analysed the use case scenario that implements all the mitigations listed in Table 11. However, it may be interesting to understand the effects of removing some mitigations, e.g., to improve the usability level of the designed protocol. To this end, let us now consider the protocol in Fig. 2 only implementing mitigation M4, which is already enforced by eID cards themselves (as far as their PIN is concerned). Table 12 displays the results of the new risk analysis, highlighting a significant worsening of the situation: SE becomes a *critical* attacker, MM is now associated with a *high* risk, and the combination CT+SE increases to a *medium* risk. These results show that carefully selecting the mitigations to implement during the design phase brings many improvements to the security level of the protocol.

In general, SE, MB and MM are clearly the most powerful attackers, as they can deceive users through social engineering techniques or compromise users' devices. Therefore, most of the mitigations aim at targetting these attackers. As a result, the risk of these powerful attackers can be significantly reduced.

## 6 RELATED WORK

The scientific literature contains many approaches to the security analysis of authentication protocols. For the sake

of brevity, we discuss only the approaches that are more relevant for our work.

A first approach consists in proposing a new authentication scheme and performing a formal analysis to demonstrate its compliance with a given set of requirements and security goals. Many scientific works follow this approach in different contexts: healthcare [23], [24], generic [25] and industrial [26], [27] IoT environments, smart homes [28], [29], wireless sensor networks [30], [31], and many more. However, several security analyses have been found flawed [32], thus leading to potentially incorrect results.

Another approach aims at analysing the security of existing authentication protocols or standards, such as FIDO [33], [34], OAuth2.0 [35], 5G EAP-TLS [36] and Single Sign-On [22]. In this case, the analysis aims at validating a third-party authentication scheme that is widely used.

In general, most of the analyses rely on formal frameworks, which requires security experts to model the considered scenario; then, they use formal provers or model checkers (such as ProVerif [12] or Tamarin [13]) to assess the security of the protocol, finally obtaining a list of the attackers that can violate the security goals. These techniques usually analyse all the attackers contained in the threat model, thus resulting in a high execution time associated with the computational complexity of the process.

Moreover, the mere list of successful attackers may not be enough: in a corporate scenario, for instance, security designers could need to have clear indications on the risks associated with the successful attackers, in order to prioritise them and understand which ones need to be mitigated more urgently and which of them can be ignored.

By combining different level of analysis, our methodology provides several benefits: the *combinatorial analysis* can be performed even by less-expert users, and provides a list of explicit attacks; the *symbolic analysis* still needs to be set up by security experts, but is performed on a smaller set of attackers as it only searches for implicit attacks, thus reducing the computational complexity of the analysis; the *risk analysis* complements the list of attackers by associating the related risks. Security designers are thus provided with a methodology that they can customise according to their needs (e.g., they can skip the symbolic analysis if they need



TABLE 12  
Results of the risk analysis with only M4 applied

Attackers	Likelihood							Impact					Risk	
	TD	O	AV	UI	SA	Overall	LSP	AS	AD	AP	Overall			
MM	2	2	7	4	2	3.40	Medium	9	8	7	2	6.50	High	High
CT+D	8	1	1	7	4	4.20	Medium	9	2	3	8	5.50	Medium	Medium
CT+ES	5	0	1	4	2	2.40	Low	9	2	3	8	5.50	Medium	Low
CT+SS	8	4	1	2	5	4.00	Medium	9	2	3	8	5.50	Medium	Medium
CT+SE	6	2	1	4	3	3.20	Medium	9	2	3	8	5.50	Medium	Medium
SE	6	9	7	4	4	6.00	High	9	8	7	2	6.50	High	Critical
MB	5	5	7	4	4	5.00	Medium	9	8	7	2	6.50	High	High

for quick, yet possibly incomplete, results).

## 7 CONCLUSIONS

In this paper, we have presented a multi-layered security methodology to analyse multi-factor authentication protocols. In addition to identifying the list of attackers that are able to compromise the protocol, our methodology provides information about the associated risks. For concreteness, we have showed how we applied the methodology to a real use case scenario: an authentication procedure based on QR codes and electronic documents that currently represents one of the main authentication procedures to access Italian Public Administration's online services. This activity, performed in the context of a long-standing collaboration with *Poligrafico e Zecca dello Stato Italiano* (the Italian Government Printing Office and Mint), supported the design of the authentication protocol by highlighting which mitigations reached the best trade-off between security and usability.

## Future Work

Given the rising importance of eID cards, we plan to elaborate on how they can be involved in other authentication contexts such as those dealing with OpenID Connect or FIDO2, in order to understand the advantages they could bring in terms of security. We are also going to refine the formal models of our use case scenario, possibly moving to more supported model checkers such as Tamarin [13], which would allow us to benefit from the active community of users. Finally, we would like to improve our risk analysis procedure by enhancing its flexibility, for instance by adapting the risk factors' values to the context in order to allow for more granular what-if analyses.

## ACKNOWLEDGMENTS

This work has been partially supported by *Futuro & Conoscenza Srl*, jointly created by FBK and IPZS. We would like to thank the anonymous reviewers for their valuable comments that helped us improve the quality of the paper.

## REFERENCES

- [1] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "Digital Identity Guidelines," ser. NIST Special Publication 800-63-3. NIST, June 2017.
- [2] Auth0, *The State of Secure Identity 2022*, 2022. [Online]. Available: <https://auth0.com/resources/whitepapers/2022-state-of-secure-identity-report>
- [3] Vanson Bourne, *The State of Authentication in the Finance Industry 2022*, 7 2022. [Online]. Available: <https://get.hypr.com/state-of-authentication-in-the-finance-industry-2022>
- [4] S. An, T. Eom, J. S. Park, J. B. Hong, A. Nhlabatsi, N. Fetais, K. M. Khan, and D. S. Kim, "CloudSafe: A Tool for an Automated Security Analysis for Cloud Computing," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2019, pp. 602–609.
- [5] G. Agosta, A. Barengi, A. Parata, and G. Pelosi, "Automated Security Analysis of Dynamic Web Applications through Symbolic Code Execution," in *2012 Ninth International Conference on Information Technology - New Generations*, 2012, pp. 189–194.
- [6] A. Armando, W. Arzac, T. Avanesov, M. Barletta, A. Calvi, A. Cappai, R. Carbone, Y. Chevalier, L. Compagna, J. Cuéllar, G. Erzse, S. Frau, M. Minea, S. Mödersheim, D. von Oheimb, G. Pellegrino, S. E. Ponta, M. Rocchetto, M. Rusinowitch, M. Torabi Dashti, M. Turuani, and L. Viganò, "The AVANTSSAR Platform for the Automated Validation of Trust and Security of Service-Oriented Architectures," in *Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2012, pp. 267–282.
- [7] A. Armando, R. Carbone, L. Compagna, J. Cuellar, and L. Tobarra, "Formal Analysis of SAML 2.0 Web Browser Single Sign-on: Breaking the SAML-Based Single Sign-on for Google Apps," in *Proceedings of the 6th ACM Workshop on Formal Methods in Security Engineering*, ser. FMSE '08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 1–10.
- [8] P. A. Grassi, E. M. Newton, R. A. Perlner, A. R. Regenscheid, J. L. Fenton, W. E. Burr, J. P. Richer, N. B. Lefkovitz, J. M. Danker, Y.-Y. Choong, K. K. Greene, and M. F. Theofanos, "Digital Identity Guidelines: Authentication and Lifecycle Management," ser. NIST Special Publication 800-63B. NIST, June 2017.
- [9] A. Armando, R. Carbone, and L. Compagna, "SATMC: a SAT-based model checker for security protocols, business processes, and security APIs," *International Journal on Software Tools for Technology Transfer*, vol. 18, no. 2, pp. 187–204, April 2016.
- [10] M. Pernpruner, R. Carbone, S. Ranise, and G. Sciarretta, "The Good, the Bad and the (Not So) Ugly of Out-of-Band Authentication with eID Cards and Push Notifications: Design, Formal and Risk Analysis," in *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, ser. CODASPY '20. Association for Computing Machinery, 2020, p. 223–234.
- [11] AVANTSSAR, *ASLan++ specification and tutorial. Deliverable D2.3*, March 2011. [Online]. Available: <https://st.fbk.eu/complementary/TDSC2022>
- [12] B. Blanchet, B. Smyth, V. Cheval, and M. Sylvestre, *ProVerif 2.00: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial*, May 2018. [Online]. Available: <https://prosecco.gforge.inria.fr/personal/bblanche/proverif/manual.pdf>
- [13] S. Meier, B. Schmidt, C. Cremers, and D. A. Basin, "The TAMARIN Prover for the Symbolic Analysis of Security Protocols," in *Computer Aided Verification*. Springer, 2013, pp. 696–701.
- [14] D. Dolev and A. C. Yao, "On the security of public key protocols," *IEEE Trans. Information Theory*, vol. 29, no. 2, pp. 198–207, 1983.
- [15] OWASP. (2018, August) OWASP Risk Rating Methodology. [Online]. Available: [https://www.owasp.org/index.php/OWASP\\_Risk\\_Rating\\_Methodology](https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology)
- [16] European Union, "Directive (EU) 2015/2366," in *Official Journal of the European Union*, vol. OJ L 337/35, 5 2015. [Online]. Available: <http://data.europa.eu/eli/dir/2015/2366/oj>

[17] Ministero dell'Interno. CIE Features. [Online]. Available: <https://www.cartaidentita.interno.gov.it/en/cie/cie-features/>

[18] ITU, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks*, October 2006. [Online]. Available: <http://handle.itu.int/11.1002/1000/13031>

[19] GIXEL, *European Card for e-Services and National e-ID Applications*, February 2009.

[20] P. A. Grassi, J. P. Richer, S. K. Squire, J. L. Fenton, E. M. Nadeau, N. B. Lefkowitz, J. M. Danker, Y.-Y. Choong, K. K. Greene, and M. F. Theofanos, "Digital Identity Guidelines: Federation and Assertions," ser. NIST Special Publication 800-63C. NIST, June 2017.

[21] M. Pernpruner, R. Carbone, G. Sciarretta, and S. Ranise. Complementary Website. [Online]. Available: <https://st.fbk.eu/complementary/TDSC2022>

[22] G. Sciarretta, R. Carbone, S. Ranise, and L. Viganò, "Formal Analysis of Mobile Multi-Factor Authentication with Single Sign-On Login," *ACM Trans. Priv. Secur.*, vol. 23, no. 3, 6 2020.

[23] R. Hajian, S. ZakeriKia, S. Erfani, and M. Mirabi, "SHAPARAK: Scalable healthcare authentication protocol with attack-resilience and anonymous key-agreement," *Computer Networks*, vol. 183, 2020.

[24] S. S. Sahoo, S. Mohanty, and B. Majhi, "A secure three factor based authentication scheme for health care systems using IoT enabled devices," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 1419–1434, 2021.

[25] H. Lee, D. Kang, J. Ryu, D. Won, H. Kim, and Y. Lee, "A three-factor anonymous user authentication scheme for Internet of Things environments," *Journal of Information Security and Applications*, vol. 52, 2020.

[26] X. Li, J. Niu, M. Z. A. Bhuiyan, F. Wu, M. Karuppiah, and S. Kumari, "A Robust ECC-Based Provable Secure Authentication Protocol With Privacy Preserving for Industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3599–3609, 2018.

[27] R. Vinoth, L. J. Deborah, P. Vijayakumar, and N. Kumar, "Secure Multifactor Authenticated Key Agreement Scheme for Industrial IoT," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3801–3811, 2021.

[28] Y. Guo, Z. Zhang, and Y. Guo, "SecFHome: Secure remote authentication in fog-enabled smart home environment," *Computer Networks*, vol. 207, 2022.

[29] Y. Xia, R. Qi, S. Ji, J. Shen, T. Miao, and H. Wang, "PUF-Assisted Lightweight Group Authentication and Key Agreement Protocol in Smart Home," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[30] Y. Lu, G. Xu, L. Li, and Y. Yang, "Anonymous three-factor authenticated key agreement for wireless sensor networks," *Wireless Netw.*, vol. 25, pp. 1461–1475, 2019.

[31] T.-Y. Wu, L. Yang, Z. Lee, S.-C. Chu, S. Kumari, and S. Kumar, "A Provably Secure Three-Factor Authentication Protocol for Wireless Sensor Networks," *Wireless Communications and Mobile Computing*, vol. 2021, 2021.

[32] Q. Wang and D. Wang, "Understanding Failures in Security Proofs of Multi-Factor Authentication for Mobile Devices," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 597–612, 2023.

[33] H. Feng, H. Li, X. Pan, and Z. Zhao, "A Formal Analysis of the FIDO UAF Protocol," in *Network and Distributed System Security Symposium*, ser. NDSS 2021, 2021, <https://www.ndss-symposium.org/ndss-paper/a-formal-analysis-of-the-fido-uaf-protocol/>.

[34] J. Guan, H. Li, H. Ye, and Z. Zhao, "A Formal Analysis of the FIDO2 Protocols," in *Computer Security – ESORICS 2022*. Springer Nature Switzerland, 2022, pp. 3–21.

[35] D. Fett, R. Küsters, and G. Schmitz, "A Comprehensive Formal Security Analysis of OAuth 2.0," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16, 2016, <https://doi.org/10.1145/2976749.2978385>.

[36] J. Zhang, L. Yang, W. Cao, and Q. Wang, "Formal Analysis of 5G EAP-TLS Authentication Protocol Using Proverif," *IEEE Access*, vol. 8, pp. 23 674–23 688, 2020.



**Marco Pernpruner** is a student of the PhD Program in Security, Risk and Vulnerability, jointly offered by the University of Genoa and Fondazione Bruno Kessler (Italy). He received the BSc degree in Information and Business Organisation Engineering from the University of Trento in 2016, and the MSc degree in Computer Science and Engineering from the University of Verona in 2019. He has also been a visiting PhD student at King's College London in 2022. His research focuses on digital identity, with a specialization in the design, security and risk assessment of multi-factor authentication and fully-remote enrollment procedures.



**Roberto Carbone** is the head of the Security & Trust Research Unit of the Center for Cybersecurity at Fondazione Bruno Kessler. He received his Ph.D. in Electronic and Computer Engineering and Telecommunications from the University of Genova (Italy) in 2009. His previous appointments include a period as visiting scholar in the Department of Computer Science at the University of Pittsburgh (Pennsylvania, US). He has been involved in several international and national research projects and industrial collaborations. His research focuses on digital identity management and the (formal) analysis of security protocols and services.



**Giada Sciarretta** is a researcher of the Security & Trust research unit of Fondazione Bruno Kessler. She obtained her MSc in mathematics and received her PhD in computer science at the University of Trento in 2012 and 2018, respectively. Her research focuses on digital identity with a specialization in the design, security and risk assessment of access delegation and single sign-on protocols (e.g., OAuth 2.0 and OpenID Connect), multi-factor authentication (e.g., based on biometric or eID cards) and fully-remote enrollment procedures.



**Silvio Ranise** is the director of the Center for Cybersecurity at Fondazione Bruno Kessler and a full professor of Computer Science at the University of Trento Department of Mathematics. He holds a PhD in Computer Engineering from the University of Genoa (Italy) and the Henri Poincaré University (Nancy, France). He has been a researcher at the INRIA-National Institute for Research in Digital Science and Technology, visiting professor at the Computer Science Department of the University of Milan and senior researcher at Fondazione Bruno Kessler.