Formal Security Analysis of Traditional and Electronic Exams*

Jannik Dreier[1], Rosario Giustolisi[2], Ali Kassem[3],
Pascal Lafourcade[4], Gabriele Lenzini[2], and Peter Y. A. Ryan[2]

[1] Institute of Information Security, ETH Zurich, Switzerland
`jannik.dreier@inf.ethz.ch`
[2] SnT/University of Luxembourg, Luxembourg
`firstname.lastname@uni.lu`
[3] Université Grenoble Alpes, CNRS, VERIMAG, Grenoble, France
`ali.kassem@imag.fr`
[4] University d'Auvergne, LIMOS, France
`pascal.lafourcade@udamail.fr`

Abstract. Nowadays, students can be assessed not only by means of pencil-and-paper tests, but also by electronic exams which they take in examination centers or even from home. Electronic exams are appealing as they can reach larger audiences, but they are exposed to new threats that can potentially ruin the whole exam business. These threats are amplified by two issues: the lack of understanding of what security means for electronic exams (except the old concern about students cheating), and the absence of tools to verify whether an exam process is secure. This paper addresses both issues by introducing a formal description of several fundamental authentication and privacy properties, and by establishing the first theoretical framework for an automatic analysis of exam security. It uses the applied π-calculus as a framework and ProVerif as a tool. Three exam protocols are checked in depth: two Internet exam protocols of recent design, and the pencil-and-paper exam used by the University of Grenoble. The analysis highlights several weaknesses. Some invalidate authentication and privacy even when all parties are honest; others show that security depends on the honesty of parties, an often unjustified assumption in modern exams.

1 Introduction

Exams are used to assess the skills, the capabilities, and the knowledge of students and professionals. Traditional exams are taken pencil-and-paper at hand. In contrast, modern exams make a partial or complete use of information and communication technology. They are called *electronic exams*, in short e-exams.

Running e-exams promises to be cheaper than staging traditional tests. E-exams are deployed easily, and they are flexible in where and when exams can be set [1]; their test sessions are open to a very large public of candidates and, if the implementation allows automatic marking, their results are immediately available. We are neutral with respect

* This research was conducted with the support of the Digital trust Chair from the University of Auvergne Foundation.

whether e-exams will improve the way people are assessed; undoubtedly though, their use has raised considerably. Several universities, such as MIT, Stanford, and Berkeley, just to cite a very few of them, have began to offer university courses using the Massive Open Online Course (MOOC) platforms which offer taking e-exams. Other institutions, such as CISCO and Microsoft, ETS[5] and ECDL[6], have already since long time adopted proprietary platforms to run electronically the tests required by their individual certification programs.

This migration towards information technology is changing considerably how taking exams looks like, but the approach in coping with their security has remained the same: it is mainly about preventing students from cheating [2]. To discourage such practices, institutions prefer invigilated tests. Where it is not possible to have human invigilators, a software running on the student computer is used, for instance ProctorU[7]. Such measures are limited and insufficient. The security as well as the trustworthiness and the reliability of exams are today threatened not only by students. Obviously there are threats coming from the use of information technology; besides, recent scandals have shown that bribed examiners and dishonest exam authorities are as willing to tamper with exams as students (*e.g.,* see [3, 4]). When this happens, the consequences are generally worse than those due to cheating. This is problematic since the growth in use of exam protocols has not been followed, nor preceded, by a rigorous understanding and analysis of their security. Looking at the security claims of the few available exam specifications, we realized that such claims are argued high level; moreover, they rely on implicit trust assumptions that, similarly to assuming that authorities are honest, are not justifiable anymore. Thus, there is the need for a formal framework to define and analyze the security of traditional and e-exam protocols. This paper fulfils this need.

Contributions: We present the first formalization for exams, and we model both traditional and electronic protocols. We define several fundamental security properties in this domain: (a) **authentication properties**, namely *Answer Origin Authentication*, *Form Authorship*, *Form Authenticity*, and *Mark Authenticity*, and (b) **privacy properties**, containing *Question Indistinguishability*, *Anonymous Marking*, *Anonymous Examiner*, *Mark Privacy*, and *Mark Anonymity*. All properties are defined in the applied π-calculus [5]. ProVerif [6, 7] is used to verify them on a given model of an exam protocol. All our verification code is available on line.[8]

We validate our approach on three existing exam systems. We check them against a Dolev-Yao attacker, and also consider cases with corrupted participants that can communicate with the attacker and so indirectly collude. The first and the second system that we validate are electronic exams. They have been recently proposed respectively by Huszti *et al.* [8], and by Giustolisi *et al.* [9]. The last one is a pencil-and-paper exam system currently used at the University of Grenoble. It resembles other European University exams where students write their personal data and a pseudonym on a piece of paper with a corner that can be folded and sealed to only leave visible the pseudonym.

[5] `www.ets.org`
[6] `www.ecdl.org`
[7] `www.proctoru.com`
[8] `apsia.uni.lu/stast/codes/exams/proverif_secrypt_journal.tar.gz`

Our security analysis (see Section 4.2, 5.2, and 6.2) gives insights on the security of the protocols. In some cases our properties hold even in presence of corrupted parties, in others they are violated already without assuming corrupted parties and just because of the attacker; in still others, we have attacks only when a subset of parties is corrupted; in a few cases the properties hold under condition that all parties are honest. Where possible and meaningful we discuss how to fix the weaknesses. In summary, our analysis reveals exactly how strong or how weak those systems are in realistic and modern exam contexts.

This work is an extended version of [10]. It discusses the framework in more detail, suggests fixes for a flawed exam protocol, and studies one new use case, the Grenoble exam. This proves that the framework can be applied to analyze the security not only of electronic exam protocols (as done in [10]), but also of pencil-and-paper exam procedures. By doing so, this work presents itself as the first theoretical framework for the security analysis of all kinds of exam protocols, traditional, computer-based, computer-assisted, and internet-based.

Related Work: Only a few papers propose e-exam protocols that guarantee some security, mainly under the assumption that some authority is trusted [8, 11–13]. Few other works [2, 14, 15] list some relevant properties for e-exams, yet only informally.

To the best of our knowledge, no formal definitions have been given for the security properties of traditional and e-exam systems. There are instead papers presenting the formalization and verification of properties in domains that seem related to e-exams, namely e-voting [16–21], e-auction systems [22–24], or cloud-based conference management systems [25]. Some of the security properties therein studied remind those we are presenting for exams. For instance, *Answer Origin Authentication* is analogous to voter and bidder authentication. *Mark Privacy* reminds ballot privacy and losing bids privacy. Yet, there are fundamental differences. In exams, *Answer Authorship* should be preserved even in the presence of colluding candidates. Conversely, vote (bid) authorship is not a problem for e-voting (e-auction), in fact unlinkability between a voter (bidder) and her vote (bid) is a desired property. An other important property for exams is to keep exam questions secret until the exam ends. We do not find such a property in e-voting where the candidates are previously known to the voters, and in e-auction where the goods to bid for are previously known to the bidders. Moreover, properties such as *Anonymous Marking*, meaning that the examiners do not know whose copy they are grading, evaluates to a sort of fixed-term anonymity. This property is meant to hold during the marking, but is trivially falsified when the marks are assigned to the candidates. In the cloud-based protocol for conference management that supports applications, evaluations, and decisions by Arapinis et al. [25], the authors identify and analyze a few privacy properties (secrecy and unlinkability) that should hold despite a *malicious-but-cautious* cloud. In e-exams we have to consider a different attacker model: corruption is not limited to the server, and the attacker is not necessarily cautious.

Finally, there are a few works studying the formal security analysis of physical protocols [26–29], but none of them considers traditional pencil-and-paper exams.

Outline: In Section 2, we model exam protocols in the applied π-calculus. Then, we specify security properties in Section 3. We validate our framework by analysing the security of two e-exam protocols [8] and [9] in Section 4 and 5, and one traditional exam in Section 6. Finally, in Section 7, we discuss our results and outline future work.

2 Modelling

We model exam protocols in the applied π-calculus [5], a process calculus designed for the verification of cryptographic protocols. To perform the automatic protocol verification we use Proverif [6, 7]. This tool uses a process description based on the applied π-calculus, but has syntactical extensions, for example its language is enriched by *events* to check reachability and correspondence properties; besides it can check *observational equivalence* properties. We use events to define various authentication properties, and we model privacy properties as equivalence properties.

Honest Party Model. Exam parties are modelled as processes in the applied π-calculus. These processes communicate via public or private channels, can create keys or fresh random values, and perform tests and cryptographic operations, which are functions on terms with respect to an equational theory describing their properties. A party is honest when it follows its specification and does not leak information to the attacker. *Threat Model.* Threats come from a Dolev-Yao attacker [30] who has full control of the network except private channels. He can eavesdrop, remove, substitute, duplicate and delay messages that the parties are sending one another, and insert messages of his choice on the public channels, but also play the protocol. Threats come also from corrupted parties, who communicate with the attacker, share personal data (*e.g.,* secret keys) with him, or receive orders (*e.g.,* how to answer a question) from him. We model corrupted parties as in Definition 8 from [31]: if process P is honest, then P^{c_1,c_2} is its corrupted version. This variant is exactly as P, but uses channels c_1 and c_2 to communicate with the attacker. Through c_1, P^{c_1,c_2} sends all its inputs and freshly generated names (but not other channel names). From c_2, P^{c_1,c_2} receives messages that can influence its behaviour.

Exam Model. The exam is the parallel composition of the processes modelling the exam parties. We have at least the following parties: *candidates* who sit for the exam; the *examiners* who mark the answers submitted by the candidates; the *question committee*, which prepares the exam questions; the *exam authorities*, which conduct the exam, and include registrars, invigilators, exam collectors, and a notification committee. In some protocols, an authority can be responsible of two or more roles.

Definition 1 (Exam protocol). *An exam protocol is a tuple $(C, E, Q, A_1, \ldots, A_l, \tilde{n}_p)$, where C is the process executed by the candidates, E is the process executed by the examiners, Q is the process executed by the question committee, A_i's are the processes executed by the authorities, and $\tilde{n}_p$ is the set of private channel names.*

All candidates execute the same process C, and all examiners the same process E. However, different candidates and examiners are instantiated with different variable values, *e.g.,* keys, identities, and answers.

Definition 2 (Exam instance). *An* exam instance *of an exam protocol given by the tuple $(C, E, A_1, \ldots, A_l, \tilde{n})$ is a closed process $EP = \nu\tilde{n}.(C\sigma_{id_1}\sigma_{a_1}| \ldots |C\sigma_{id_j}\sigma_{a_j}| E\sigma_{id'_1}\sigma_{m_1}| \ldots |E\sigma_{id'_k}\sigma_{m_k}|Q\sigma_q|A_1\sigma_{dist}| \ldots |A_l)$, where $\tilde{n}$ is the set of all restricted names, which includes the private channels; $C\sigma_{id_i}\sigma_{a_i}$'s are the processes run by the candidates, the substitutions σ_{id_i} and σ_{a_i} specify the identity and the answers of the i^{th} candidate respectively; $E\sigma_{id'_i}\sigma_{m_i}$'s are the processes run by the examiners, the substitution $\sigma_{id'_i}$ specifies the i^{th} examiner's identity, and σ_{m_i} specifies for each possible question/answer pair the corresponding mark; Q is the process run by the question committee, the substitution σ_q specifies the exam questions; the A_i's are the processes run by the exam authorities, the substitution σ_{dist} determines which answers will be submitted to which examiners for grading. Without loss of generality, we assume that A_1 distributes the copies to the examiners.*

Definition 2 handles equally examiners that are machines and examiners that are humans: they are both entities that mark answers. Q and A_1 can coincide when there is only one authority A: in that case, $Q\sigma_q|A_1\sigma_{dist}$ is simplified as $A\sigma_q\sigma_{dist}$.

We organize the exam's steps in four phases. (1) *Registration:* the exam authority (the registrar) creates a new examination and checks the eligibility of candidates who attempts to register for it; (2) *Examination:* the exam authority authenticates the candidates, and sends to each of them an *exam form* that contains the exam questions. Each candidate fills the form with his answer, and submits it to the exam collector; (3) *Marking:* the authority distributes the forms submitted by the candidates to the examiners, who in their turn evaluate and mark them; (4) *Notification:* once the forms have been evaluated, the marks are notified to the candidates.

3 Security Properties

We formalize *authentication* and *privacy* properties. They best represent exam security requirements as corroborated by other works [2, 14, 15]. We introduce four authentication properties meant to ensure the associations between the candidate's identity, the answer, and the mark being preserved through all phases. When authentication holds there is no loss, no injection, and in general no manipulation of the exam forms from examination to notification. We also introduce five privacy properties that ensure the anonymity of critical parties.

It is worth to report that in the context of exams other classes of properties might be of interest, for instance verifiability, reliability, or accountability, but we do not study them here. This task is ongoing work.

3.1 Authentication properties

We model our authentication properties as correspondence properties, a well-known approach [32, 33]. Specific events, whose parameters refer to the pieces of information in the exam form, flag important steps in the execution of the exam. Events are annotations that do not change a process behavior, but are inserted at precise locations to allow reasoning about the exam's execution. In the following id_c is the candidate identity,

$ques$ the question(s), ans the answer(s), $mark$ the mark(s), id_form is an identifier of the exam form used during marking, and id_e is the examiner's identity. The id_form is only used to identify an exam form during marking. This could be a pseudonym to allow anonymous marking, or simply the candidate identity if the marking is not anonymous. In our model, we use the following events.

• $reg(id\_c)$: is the event inserted into the registrar process at the location where candidate id_c has successfully registered for the exam.

• $submitted(id\_c, ques, ans)$: is the event inserted into the process of candidate id_c in the examination phase, injected on the candidate process at the location where he sends his answer ans corresponding to the question $ques$.

• $collected(id\_c, ques, ans)$: is the event inserted into the exam collector's process in the examination phase, just after it received and accepted the exam form (id_c, $ques$, ans) from candidate id_c.

• $distrib(id\_c, ques, ans, id\_form, id\_e)$: is the event inserted into the authority process in the marking phase, when it assigns the exam form (id_c, $ques$, ans) from candidate id_c to the examiner id_e using the identifier id_form.

In exam with only one examiner the event $distrib$ seems not necessary, but it is: it links exam forms to id_form and, for instance, helps revealing when identical answers are graded multiple times (and not necessarily with the same mark).

• $marked(ques, ans, mark, id\_form, id\_e)$: is the event inserted into the examiner id_e's process in the marking phase, at the location where he marked the question/answer pair ($ques$, ans) identified by id_form with the mark $mark$.

• $notified(id\_c, mark)$: is the event inserted into the process of candidate id_c in the notification phase, just after he received and accepted the mark $mark$ from the responsible authority.

Authentication properties are correspondence properties among these events. All properties have the following structure: "on every trace the event e_1 is preceded by the event e_2".

The first authentication property is *Answer Origin Authentication*. When satisfied, it ensures that only one exam form from each candidate and only the forms submitted by eligible candidates (registered) are actually collected.

Definition 3 (Answer Origin Authentication) *An exam protocol ensures* Answer Origin Authentication *if each occurrence of the event $collected(id\_c, ques, ans)$, for every exam process EP, is preceded by a distinct occurrence of the event $reg(id\_c)$ on every execution trace.*

At examination phase, each candidate submits his exam form with an answer, and the collector collects the forms. *Form Authorship* ensures that the contents of each collected exam form (id_c, $ques$, and ans) are not modified after submission.

Definition 4 (Form Authorship) *An exam protocol ensures* Form Authorship *if, for every exam process EP, each occurrence of the event $collected(id\_c, ques, ans)$ is preceded by a distinct occurrence of the event $submitted(id\_c, ques, ans)$ on every execution trace.*

Similarly, *Form Authenticity* ensures that the content of each exam form is not modified after the collection and until after the form is marked by an examiner.

Definition 5 (Form Authenticity) *An exam protocol ensures* Form Authenticity *if each occurrence of the event $marked(ques, ans, mark, id\_form, id\_e)$, for every exam process EP, is preceded on every execution trace by a distinct occurrence of the events $collected(id\_c, ques, ans)$ and $distrib(id\_c, ques, ans, id\_form, id\_e)$.*

At notification phase, the candidate should receive the mark which was assigned by the examiner to his answer. We call this property *Mark Authenticity*.

Definition 6 (Mark Authenticity) *An exam protocol ensures* Mark Authenticity *if, for every exam process EP, each occurrence of the event $notified(id\_c, mark)$ is preceded by a distinct occurrence of the events $marked(ques, ans, mark, id\_form, id\_e)$ and $distrib(id\_c, ques, ans, id\_form, id\_e)$ on every execution trace.*

Note that *Mark Authenticity* ensures that the candidate is notified with the mark delivered by the examiner on the answer assigned to him by the authority. This answer may be different from that submitted by the candidate. Only if also *Form Authorship* and *Form Authenticity* hold then the candidate can be sure that the assigned and submitted answers are identical. Moreover, *Mark Authenticity* does not guarantee that the mark is computed correctly.

3.2 Privacy properties

We model our privacy properties as observational equivalence, a standard choice for such kind of properties [33, 34]. We use the *labeled bisimilarity* ($\approx_l$) to express the equivalence between two processes [5]. Informally, two processes are equivalent if an observer has no way to tell them apart.

Notation. In the following, we use two simplifying notations: (a) "$EP_I[\_]$", called context, is the process EP without the identities in the set I. We use it, for instance, to specify exactly the processes for candidates id_1 and id_2 without repeating the entire exam instance; in that case we rewrite EP as $EP_{\{id_1,id_2\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}]$; (b) "$EP|_e$" stands for process EP without the code that follows the event e.

The first privacy property says that questions are kept secret until the exam starts.

Definition 7 (Question Indistinguishability) *An exam protocol ensures* Question Indistinguishability *if for any exam process EP that ends with the registration phase, any questions q_1 and q_2, we have that: $EP_{\{id_Q\}}[Q\sigma_{q_1}]|_{reg} \approx_l EP_{\{id_Q\}}[Q\sigma_{q_2}]|_{reg}$.*

Question Indistinguishability states that two processes with different questions have to be observationally equivalent until the end of the registration phase. This prevents the attacker from obtaining information about the exam questions before the examination phase starts. This property requires the question committee to be honest; otherwise the property is trivially violated since the committee reveals the questions to the attacker. However, it is particularly interesting to consider corrupted participants, for example candidates might be interested in obtaining the questions in advance. We can do this by replacing honest candidates with corrupted ones. For example, if we assume that candidate id_1 is corrupted, we obtain

$$EP_{\{id_1,id_Q\}}[(C\sigma_{id_1}\sigma_{a_1})^{c_1,c_2}|Q\sigma_{q_1}]|_{reg}\approx_l EP_{\{id_1,id_Q\}}[(C\sigma_{id_1}\sigma_{a_1})^{c_1,c_2}|Q\sigma_{q_2}]|_{reg}$$

The next property ensures that the marking process is done anonymously, *i.e.,* that two instances where candidates swap their answers cannot be distinguished until after the end of the marking phase.

Definition 8 (Anonymous Marking) *An exam protocol ensures* Anonymous Marking *if for any exam process EP that ends with the marking phase, any two candidates id_1 and id_2, and any two answers a_1 and a_2, we have that:*
$EP_{\{id_1,id_2\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}]|_{mark}\approx_l EP_{\{id_1,id_2\}}[C\sigma_{id_1}\sigma_{a_2}|C\sigma_{id_2}\sigma_{a_1}]|_{mark}$.

Anonymous Marking ensures that the process where id_1 answers a_1 and id_2 answers a_2 is equivalent to the process where id_1 answers a_2 and id_2 answers a_1. This prevents the attacker to obtain the identity of the candidate who submitted a certain answer before the marking phase ends. For this property, it is interesting to consider corrupted examiners. It can be done using the same technique employed for corrupted candidates outlined above. We can also have some corrupted candidates, however the candidates id_1 and id_2 who are assigned the two different answers have to be honest – otherwise the property can be trivially violated by one of them revealing his answer to the attacker.

To prevent bribing or coercion of the examiners, it might be interesting to ensure their anonymity, so that no candidate knows which examiner marked his copy.

Definition 9 (Anonymous Examiner) *An exam protocol ensures* Anonymous Examiner *if for any exam process EP, any two candidates id_1 and id_2, any two examiners id'_1 and id'_2, and any two marks m_1 and m_2, we have that: $EP_{\{id_1,id_2,id'_1,id'_2,id_{A_1}\}}$ $[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}|E\sigma_{id'_1}\sigma_{m_1}|E\sigma_{id'_2}\sigma_{m_2}|A_1\sigma_{dist_1}]\ \approx_l\ EP_{\{id_1,id_2,id'_1,id'_2,id_{A_1}\}}$ $[C$ $\sigma_{id_1}\ \sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}|E\sigma_{id'_1}\sigma_{m_2}|E\sigma_{id'_2}\sigma_{m_1}|A_1\sigma_{dist_2}]$ where σ_{dist_1} attributes the exam form of candidate id_1 to examiner id'_1 and the exam form of candidate id_2 to examiner id'_2, and σ_{dist_2} attributes the exam form of candidate id_1 to examiner id'_2 and the exam form of candidate id_2 to examiner id'_1.*

Anonymous Examiner ensures that a process in which examiner id'_1 grades the exam form of candidate id_1 and examiner id'_2 grades that of candidate id_2 cannot be distinguished from a process in which id'_1 grades the exam form of id_2 and id'_2 grades that of id_1. Note that to ensure that in both cases the candidates receive the same mark, we also have to swap σ_{m_1} and σ_{m_2} between the examiners. Similar to *Anonymous Marking*, this property prevents the attacker to obtain or guess the identity of the examiner who marked a certain answer. *Anonymous Examiner* requires that the examiners id'_1 and id'_2 are honest, otherwise it will trivially violated by one of them revealing the mark he gave. We can again include corrupted candidates as they might be interested in finding out which examiner marked their copies.

In some exams settings the marks have to remain private. This is formalized in the next property.

Definition 10 (Mark Privacy) *An exam protocol ensures* Mark Privacy *if for any exam process EP, any marks m_1, m_2, we have that: $EP_{\{id'\}}[E\sigma_{id'}\sigma_{m_1}]\ \approx_l\ EP_{\{id'\}}[E$ $\sigma_{id'}\sigma_{m_2}]$.*

Mark Privacy guarantees that two processes where the examiner id'_1 assigns for the same answer, entailed by the same context EP, two different marks m_1, m_2, cannot be distinguished from each other. Depending on the exam policy this can be an optional property since some exams system may publicly disclose the marks of the candidates. However, the intuition here is that candidate's performance should not be known to any other candidate. Again, we can assume that some candidates are corrupted and try to find out the marks of their colleagues, or that an examiner tries to find out the mark achieved by a candidate. The candidate who is assigned the two different marks has to be honest – otherwise the property is violated by him revealing his mark to the attacker. Similarly the examiner assigning the marks has to be honest, otherwise he can reveal the mark himself.

The previous definition of *Mark Privacy* ensures that the attacker cannot know the mark of a candidate. A weaker variant of *Mark Privacy* is *Mark Anonymity*, *i.e.,* the attacker might know the list of all marks, but is unable to associate a mark to its corresponding candidate. This is often the case in practice, where a list of pseudonyms (*e.g.,* student numbers) and marks is published.

Definition 11 (Mark Anonymity) *An exam protocol ensures* Mark Anonymity *if for any exam process EP, any candidates id_1, id_2, any examiner id'_1, any answers a_1, a_2 and a distribution σ_{dist} that assigns the answers of both candidates to the examiner, and two substitutions σ_{m_a} and σ_{m_b} which are identical, except that σ_{m_a} attributes the mark m_1 to the answer a_1 and m_2 to a_2, whereas σ_{m_b} attributes m_2 to the answer a_1 and m_1 to a_2, we have that: $EP_{\{id_1,id_2,id'_1,id_{A_1}\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}|E\sigma_{id'_1}\sigma_{m_a}$ $|A_1\sigma_{dist}] \approx_l EP_{\{id_1,id_2,id'_1,id_{A_1}\}}[C\sigma_{id_1}\sigma_{a_1}|C\sigma_{id_2}\sigma_{a_2}|E\sigma_{id'_1}\sigma_{m_b}|A_1\sigma_{dist}]$*

The definition states that if an examiner id'_1, who is assigned the same answers a_1 and a_2 as σ_{dist} is unchanged, swaps the marks between these answers, the two situations cannot be distinguished by the attacker. This means that a list of marks can be public, but the attacker must be unable to link the marks to the candidates. Again, we can consider corrupted parties, but this definition requires the two concerned candidates and the two concerned examiners to be honest. Otherwise they can simply reveal the answer and the associated mark, which allows to distinguish both cases.

It is also easy to see that a protocol ensuring *Mark Privacy* also ensures *Mark Anonymity*. In fact, σ_{m_a} and σ_{m_b} are special cases of σ_{m_1} and σ_{m_2}.

4 Huszti & Pethő Protocol

We first analyze the protocol by Huszti & Pethő [8] (in short, H&P protocol). This protocol aims to ensure authentication and privacy for e-exams in presence of corrupted candidates, examiners, and exam authorities; the guarantees are argued only informally in [8]. Notably from a point of view of this paper, all arguments supporting privacy rely on the reliability of a single component, the *reusable anonymous return channel*, or RARC [35]. A RARC implements anonymous two-way conversations. A sender posts a message to a recipient and the RARC ensures its anonymity; in its turn, the recipient can reply to that message without knowing nor learning the sender's identity, sure that the RARC will dispatch it to actual sender. RARC ensures the anonymity of the messages, and the entire conversation remains untraceable to an external attacker, but it does not guarantee the secrecy of the messages [35].

A RARC is implemented by a re-encryption mixnet. The mix servers jointly generate and share an ElGamal key pair (PK_{MIX}, SK_{MIX}) and a pair of public/private signing keys (SPK_{MIX}, SSK_{MIX}). The sender A and the receiver B also hold ElGamal public/private key pairs, (PK_A, SK_A) and (PK_B, SK_B) respectively. A and B are represented by ID_A and ID_B, identity tags which can be for example A's and B's email addresses. To send the message m to B, sender A submits to the mixnet the tuple $Mix(m, A, B)$ that denotes $(\{ID_A, PK_A\}_{PK_{MIX}}, \{m\}_{PK_{MIX}}, \{ID_B, PK_B\}_{PK_{MIX}})$ and proves knowledge of $\{ID_A, PK_A\}$ and of $\{ID_B, PK_B\}$. The proofs of knowledge are claimed to impede the attacker's decrypting the triplets by using the mixnet as an oracle (in Section 4.2 we falsify this claim). The mixnet waits to collect more triplets and shuffles them. Then, it adds a checksum to the triplets, which is supposed to vouch for integrity (again, we disprove this claim in Section 4.2).

The message m is then re-encrypted with the public key of B using a switching encryption keys technique. The mixnet signs the encrypted public key of A. Thus B receives the pair $(sign(\{ID_A, PK_A\}_{PK_{MIX}}, SK_{MIX}), \{m\}_{PK_B})$ where $sign(x, sk)$ is message x plus the signature with the secret key sk. Then B replies to A with a new message m' by sending to the mixnet $(Mix(m', B, A), sign(\{ID_A, PK_A\}_{PK_{MIX}}, SK_{MIX}))$ and proving only knowledge of $\{ID_B, PK_B\}$. The mixnet checks the proof and the signature, and then processes the tuples like a normal message.

4.1 Protocol Description

We briefly review the H&P protocol here, for the full details see the original paper [8]. The protocol relies upon different cryptographic building blocks. The ElGamal cryptosystem [36] is used to provide parties with public/private key pairs. A RARC implements anonymous two-way communication. A network of servers (NET) provides a timed-release service. It will create and revoke a candidate's pseudonym. More precisely, the NET's contribution to the pseudonym is shared among the servers using the threshold Shamir secret sharing system [37]. At notification, a subset of the NET servers use their shares to recover the secret and de-anonymize candidate: the exam authority can so associate the answer with the corresponding candidate. To avoid plagiarism, the protocol assumes that no candidate reveals his private key to another candidate, and that invigilators supervise candidates during the examination.

The original protocol has five phases: examiner and the candidate registration, examination, marking and notification. To match this structure with our exam model, which has four phases, we merge the candidate and examiner registrations into a single registration phase.

Examiner Registration: The exam authority publishes the public parameters to identify a new examination. The question committee then signs and sends the questions and the starting time of the phases encrypted with the public key of the RARC mixnet. The mixnet forwards the message only when the examination begins. The examiner is then provided with a pseudonym, which is jointly generated by the exam authority and the examiner. The examiner verifies the correctness of the pseudonym by using a zero-knowledge proof (ZKP). Then, the examiner sends his pseudonym to the exam authority, and proves the knowledge of his secret key.

Candidate Registration: The registration of a candidate slightly differs from the registration of an examiner. The candidate pseudonym is jointly calculated by the exam authority, the candidate, and also the NET to provide anonymity for the candidates. The NET stores the secret values used for the pseudonym generation, which can be used to de-anonymize the candidate after the examination has finished. Again, the candidate finally verifies the correctness of his pseudonym using a ZKP.

Examination: The candidate sends his pseudonym via the RARC to the exam authority and proves the knowledge of his private key. Then, the exam authority checks whether the candidate is registered for the examination, and sends him the questions signed by the question committee. The candidate sends his answer, again via the RARC. The exam authority replies with a receipt which consists of the hash of all parameters seen by the exam authority during the examination, the transcription of the ZKPs, and the time when the answer was submitted.

Marking: The exam authority chooses an examiner who is eligible for the examination, and forwards him the answer via the RARC. Then the examiner assigns a mark to the answer, and authenticates them using a ZKP.

Notification: When all the answers are marked, the NET de-anonymizes the pseudonyms linked to the answers. The exam authority stores the marks.

4.2 Formal Analysis

The equational theory depicted in Figure 1 models the cryptographic primitives used within the H&P protocol. It includes the well-known model for digital signatures. It also describe zero-knowledge proofs (ZKP). The theory we use is inspired by Backes *et al.* [38] which models a ZKP of a secret exponent as two functions, zkp_proof for proof, and zkp_sec for verification. The function $zkp\_proof(public, secret)$ takes as arguments a secret and public parameters (i.e. the exponent and the generator to the power of the exponent). It can be constructed only by the prover who knows the secret parameter. The verification function $zkpsec(zkp\_proof(public, secret), verinfo)$ takes as arguments the proof function and the verification parameter $verinfo$. The verifier only accepts the proof if the relation between $verinfo$ and $secret$ is satisfied. We support the model for the ZKP of the equality of discrete logarithms $check\_proof$ with tables in ProVerif. This is due to the difficulties of ProVerif when dealing with associativity of multiple exponents, which is used in the H&P protocol. In particular, this approach is needed to let ProVerif terminate for Mark Privacy and Mark Anonymity. It is sound because it limits the attacker capability to generate fake ZKPs, as he cannot write and read ProVerif's table. Nevertheless, ProVerif still finds counterexamples that falsify the property, as shall we see later.

We also assume the same generator is used for generating the pseudonyms of candidates and examiners. This is sound because we distinguish the roles, and each principal is identified by its public key. We replace the candidate identity with his corresponding pseudonym inside the events to check authentication properties. We note that the replacement is also sound because the equational theory preserves the bijective mapping between the keys that identify the candidate and his pseudonym.

First we analyze the RARC alone and show that there are attacks on anonymity and privacy (see next paragraph).

$$\begin{aligned}
getmess(sign(m,k)) &= m\\
checksign(sign(m,k), pk(k)) &= m\\
exp(exp(exp(g,x),y),z) &= exp(exp(exp(g,y),z),x)\\
checkproof(xproof(p,p',t,exp(t,e),e),p,p',t,exp(t,e)) &= true\\
zkpsec(zkp\_proof(exp(exp(g,e1),e2),e2),exp(exp(g,e1),e2)) &= true
\end{aligned}$$

Fig. 1: Equational theory to model H&P protocol

Attack on RARC: ProVerif shows that the RARC fails to guarantee both secrecy of messages and anonymity of sender and receiver identities, which is its main purpose inside the H&P protocol. We refer the triplet $\langle c_1, c_2, c_3 \rangle$ as the encrypted messages that A submits to the mixnet when she wants to send a message to B. From the description of RARC given at the beginning of this section, we recall that c_1 encrypts the A's public key, c_2 encrypts the message to B, and c_3 encrypts the B's public key. All cipher-texts are encrypted with the mixnet's public key.

Property	Result	Time
Answer Origin Authentication	$\times$	26 s
Answer Origin Authentication*	✓ (E,EA,C,NET)	3 s
Form Authorship	$\times$	3 s
Form Authorship*	$\times$	2 s
Form Authenticity	$\times$	33 s
Form Authenticity*	✓ (E,EA,C,NET)	3 s
Mark Authenticity	$\times$	52 s
Mark Authenticity*	✓ (E,EA,C,NET)	4 s
Question Indistinguishability	$\times$	< 1 s
Anonymous Marking	$\times$	1h 58 m 33 s
Anonymous Examiner	$\times$	6h 37 m 33 s
Mark Privacy	$\times$	23 m 59 s
Mark Anonymity	$\times$	49 m 5 s

Fig. 2: Summary of our analysis on the formal model of the H&P protocol; $\times$ indicates that the property does not hold despite all parties being honest. $(*)$ are the results after applying our fixes.

The attacker uses the RARC as a decryption oracle, letting the RARC reveal any of the plaintexts. The attack works as follows. The attacker chooses one of the three ciphertexts (depending on whether he wants to target the contents of the message, or the identities of the sender and receiver) and submits this as a new message. For example, if the attacker targets $c_1 = \{ID_A, PK_A\}_{PK_{MIX}}$, he resubmits c_1 as a new encrypted message, which means that $c_2' = c_1$ in the new triplet. He can leave the encryption of the senders key and the proof concerning the key unchanged, but replaces the encryption of the receiver's key with a public key PK_I for which he knows the corresponding secret key SK_I. In our example this means $c_3' = \{ID_I, PK_I\}_{PK_{MIX}}$. The attacker can also provide the necessary proof of knowledge of plaintext, since he knows this plaintext.

The RARC then mixes the input messages, and sends the encryption of the message under the receiver's public key to the receiver. In our example the attacker receives $\{ID_A, PK_A\}_{PK_I}$. Since the attacker knows the secret key SK_I he can obtain the original message. In our example he gets ID_A, the identity of the sender which should have remained anonymous. Since the attacker can substitute any of the items in the triplet as the new message, the RARC does neither ensure secrecy of the messages nor the anonymity of the sender or the receiver. Note that the checksum meant to guarantee the integrity of the triplet is only added after the submission of the message and is only used inside the mixnet. Hence, the checksum does not prevent the attacker from submitting a modified triplet. Even if it were added before, it would not prevent the attack as the knowledge of the ciphertexts is sufficient to compute the checksum.

Note that the RARC was originally designed to withstand a passive attacker that however can statically corrupts parties [35]. This is not realistic in the e-exam setting where corrupted parties could actively try to cheat. Even an attacker that statically corrupts parties can attack the RARC as described above. A corrupted party instructed statically can send and receive messages via the RARC on behalf of the attacker. The attacker still has to intercept those messages before they enter the RARC, but this is possible with insecure networks such as the Internet.

All properties fail with such a RARC. But even if we replace this RARC with an ideal implementation – which, according to the RARC original requirements [35], ensures anonymity of senders and receivers but not message secrecy, implemented as an anonymous channel in ProVerif – the H&P protocol does not satisfy any of our properties. The next paragraphs details the findings, and Figure 2 summarizes the results found assuming all parties being honest.

Authentication properties: We verified the authentication properties without and with an ideal RARC. All the following counterexamples remain valid in both cases.

ProVerif finds a counterexample for *Answer Origin Authentication* where the attacker can create a fake pseudonym that allows him to take part in an exam for which he did not register. This is possible because the exam authority does not check whether the pseudonym has been actually created using the partial information provided by the timed-release service. The attacker generates his own secret key SK_A, and calculates an associate pseudonym, which sends to the exam authority. The exam authority successfully verifies the received data and that the attacker knows SK_A, thus the exam authority accepts the answer. Regarding *Form Authorship*, ProVerif shows the same attack trace that falsifies Answer Origin Authentication. In fact, the exam authority may collect an exam form where the pseudonym is exchanged with one chosen by the attacker.

ProVerif also shows that the H&P protocol does not ensure *Form Authenticity*, because there is no mechanism that allows the examiner to check whether the answers have been forwarded by the exam authority. Even if the original RARC is used and the answer is encrypted with the public key of the mixnet, this does not guarantee that the exam authority actually sent the message.

Regarding *Mark Authenticity*, ProVerif provides a counterexample in which the attacker can forward any answer to any examiner, even if the answer was not collected by the exam authority. Moreover, the attacker can notify the candidate by himself with a mark of his choice.

Privacy properties: ProVerif finds an attack trace on *Question Indistinguishability*. This is because the attack on the RARC exposes the message and the identities of the sender and receiver. As the questions are sent through the RARC, the attacker can obtain them. Since the candidate's answer is also sent through the RARC, *Anonymous Marking* does not hold: the answer can be linked to its corresponding sender. The protocol ensures neither *Mark Privacy* nor *Anonymous Examiner*, as the marks are also sent through the RARC. Hence, they can be decrypted and the examiner can be identified.

We checked the H&P protocol in ProVerif assuming ideal RARC case ProVerif shows an attack for each property. *Anonymous Examiner* can be violated because the attacker can track which examiner accepts the ZKP when receiving the partial pseudonym, and then associate to the examiner the answer that the latter grades. Moreover, a similar attack on *Anonymous Marking* remains: the attacker can check whether a candidate accepts the ZKP to associate him with a pseudonym, and then identify his answer. *Mark Privacy* fails because the examiner sends the mark to the exam authority via the RARC, which does not ensure secrecy. Finally, ProVerif shows that the H&P protocol does not satisfy *Mark Anonymity*: the attacker can track which pseudonym is assigned to a candidate and the mark is not secret, and link a candidate to the assigned mark.

Fixing Authentication We propose four simple modifications to the H&P protocol in order to achieve a set of authentication properties. In particular, we prove in ProVerif that the so modified protocol achieves Answer Origin Authentication, Form Authenticity, and Mark Authenticity. We found no easy solution for Form Authorship as the protocol sees no signatures for candidates, and RARC does not guarantee authentication.

Concerning *Candidate registration*, we observe that EA and NET do not need to communicate anonymously via RARC, as the original protocol prescribes. Conversely, they both need to authenticate each other messages to avoid considering attacker message injections. Thus, the first modification consists on the NET receiving the partial pseudonyms generated by EA via a secure channel instead via a RARC. In doing so, the attacker cannot use the NET to generate fake pseudonyms.

As second modification we let NET send via secure channel the eligible pseudonyms to the EA, who, in doing so, can generate ZK proofs of equality of discrete logarithm to eligible pseudonyms only. The EA can also store the eligible pseudonyms, which can be checked at examination before accepting a test from a candidate.

Concerning *Marking*, we note that the examiner cannot verify whether a test has been sent by the EA. Since the anonymity requirement is on the examiner but not on the EA, the latter can sign the test. Thus, the third modification consists on EA signing the test prior to forward it to the chosen examiner, who authenticates the signature.

The last issue concerns the form identifier the EA affixes to the test before forwarding it to the examiner. Since the candidate is unaware of such identifier, the attacker can notify him any other examiner's mark. The forth modification sees the EA adding the form identifier to the candidate's submission receipt, which is also signed by the EA. Also the examiner adds the form identifier to the marking receipt, so the candidate can verify whether he has been notified with the correct mark.

5 Remark! Protocol

We first describe the protocol presented in [9] and then the results of our analysis.

5.1 Protocol Description

The Remark! protocol has the same set of parties of the H&P protocol, but relies on a different approach. The NET is indeed several servers that implement an *exponentiation mixnet* [39]. The speciality of exponentiation mixnets is that each server blinds its entries by a common exponent value. On entry X, the mixnet outputs X^r where r is the product of the secret exponent values of the servers. At registration, the NET creates the pseudonyms for the candidates and examiners without involving any of them. The pseudonyms are eventually used as public-key encryption and signature verification keys in such a way to allow parties to communicate anonymously. A bulletin board[9] is used to publish the pseudonyms, the test questions and the receipts of test submissions. The combination of the exponentiation mixnet and a bulletin board is meant to ensure anonymity and verifiability.

Remark! only assumes that each party is given a pair of public/private key with a common generator g, i.e. the private key x and the public key $y = g^x$. Below, we present the protocol within the four exam phases.

Registration: The list of eligible candidates' and examiners' public keys is sent as a batch to the NET. The NET calculates the pseudonyms by raising the initial public keys to a common value $r = \prod_i r_i$. More specifically, each mix server raises the input message to a secret value r_i, and forwards it to another mix server. At the same time the NET blindly permutes the batch of public keys. The so obtained keys eventually become the pseudonyms for candidates and examiners. Along with the pseudonyms $y' = y^r = (g^x)^r$, the NET publishes a new generator h, which is the output of g raised to the product of each mix server secret value, i.e. $h = g^r$. Both the candidates and the examiners can identify their own pseudonyms by raising h to their secret key x, i.e. $h^x = (g^r)^x$. The pseudonyms serve as public encryption and signature verification keys from now on. Two different batches are used for candidates and examiners because only the identities of candidates are revealed at notification.

Examination: The exam authority signs and encrypts the test questions with the candidate's pseudonym and publishes them on the bulletin board. Each candidate submits his answer, which is signed with the candidate's private key (but using the generator h instead of g) and encrypted with the public key of the exam authority. The exam authority collects the test answer, checks its signature using the candidate's pseudonym, re-signs it, and publishes its encryption with the corresponding candidate's pseudonym as receipt.

Marking: The exam authority encrypts the signed test answer with an eligible examiner pseudonym and publishes the encryption on the bulletin board. The corresponding examiner marks the test answer, and signs it with his private key (again using the generator h instead of g). The examiner then encrypts it with the exam authority public key, and submits its marks to the exam authority.

Notification: When the exam authority receives all the candidate evaluations, it publishes the signed marks, each encrypted with the corresponding candidate's pseudonym. Then, the NET servers de-anonymize the candidate's pseudonyms by revealing their secret exponents. Hence the candidate anonymity is revoked, and the mark can finally

[9] A public append-only memory.

be registered. Note that the examiner's secret exponent is not revealed to ensure his anonymity even after the exam concludes.

5.2 Formal Analysis

We analyze Remark! with ProVerif, following similar techniques as the one used in the analysis of the H&P protocol. Figure 4 sums up the results together with the time required for ProVerif to conclude on the same PC used for H&P. We model the bulletin board as a public channel, and use the equational theory depicted in Figure 3. The equations for encryption and signatures are standard, but we also added the possibility of using the pseudonym keys to encrypt or sign. The public pseudonym, which also serves as exam form identifier, is obtained using the function $pseudo\_pub$ on the public key and the random exponent. The function $pseudo\_priv$ can be used to decrypt or sign messages, using the private key and the new generator g^r (modelled using the function exp) as parameters. The function $checkpseudo$ allows us to check if a pseudonym corresponds to a given secret key (or its pseudonym variant).

$$\begin{aligned}
checkpseudo(pseudo\_pub(pk(k), rce), pseudo\_priv(k, exp(rce))) &= true\\
decrypt(encrypt(m, pk(k), r), k) &= m\\
decrypt(encrypt(m, pseudo\_pub(pk(k), rce), r), pseudo\_priv(k, exp(rce))) &= m\\
getmess(sign(m, k)) &= m\\
checksign(sign(m, k), pk(k)) &= m\\
checksign(sign(m, pseudo\_priv(k, exp(rce))), pseudo\_pub(pk(k), rce)) &= m
\end{aligned}$$

Fig. 3: Equational theory to model Remark! protocol

Authentication properties: Assuming an attacker in control of the network and all parties to be honest, we can successfully verify all authentication properties in ProVerif. To model properly authentication in ProVerif, where events need to refer to candidates along the whole code, it was necessary to replace the candidate key (used to identify the candidate) with the candidate's pseudonym inside the events. This is sound because there is a bijective mapping between keys and pseudonyms, and pseudonyms are always available.

We also verified the authentication properties considering corrupted parties. In this case, all properties are guaranteed except *Form Authenticity*. The attack trace shows that a corrupted candidate can pick the examiner of his choice by re-encrypting the signed receipt received from the exam authority. It means that the candidate can influence the choice of the examiner who will correct his exam. As the protocol description envisages an access control for publishing into the bulletin board, a feature that we could not code in ProVerif, we cannot claim this to be an attack as the candidate may not be allowed to post on the bulletin board. However, we demonstrate that with a simple fix there is no need of access control policies for publishing into the bulletin board. The fix consists in making the intended pseudonym of an examiner explicit within the signature that designates the examiner as evaluator of an exam. In doing so, the exam authority's signature within the receipt cannot be used by a candidate to designate any examiner because the receipt includes no examiner's pseudonym. The exam authority will only accept exam evaluations that contain its signature on examiner's pseudonym. Considering the fix, ProVerif confirms that Remark! guarantees all the security properties including *Form Authenticity*, even in presence of corrupted candidates.

Property	Result	Time
Answer Origin Authentication	✓ (NET)	< 1 s
Form Authorship	✓ (C, EA, NET)	< 1 s
Form Authenticity	✓ (C, E, EA, NET)	< 1 s
Form Authenticity*	✓ (E, EA, NET)	< 1 s
Mark Authenticity	✓ (E, EA, NET)	< 1 s
Question Indistinguishability	✓ (E, EA, NET)	< 1 s
Anonymous Marking	✓ (C, NET)	1 s
Anonymous Examiner	✓ (E, NET)	< 1 s
Mark Privacy	✓ (EA, NET)	3 m 39 s

Fig. 4: Summary of our analysis on the formal model of the Remark! protocol. The parties which are assumed to be honest for the result to hold are in brackets. NET is the process that models the mixnet. (*) after applying our fix.

Privacy properties: All the privacy properties are satisfied. For *Question Indistinguishability*, we only assume the exam authority to be honest, and then conclude that the property holds. For *Mark Privacy*, we assume only the concerned candidate and examiner, as well as the exam authority, to be honest. All other candidates and examiners are corrupted, and ProVerif still concludes successfully. Note that this subsumes a case with multiple honest candidates and examiners, since a dishonest party can behave like a honest party. This also implies that the protocol ensures *Mark Anonymity* as noted above. For *Anonymous Examiner*, we assume only the examiners and the NET to be honest. If the NET publishes the pseudonyms in random order, ProVerif concludes successfully. Similarly for *Anonymous Marking*, we assume only the candidates and the NET to be honest. Again, if the NET publishes the pseudonyms in random order, ProVerif concludes successfully.

6 Grenoble Protocol

The third case study we analyze is Grenoble exam, which is paper-and-pencil procedure used to evaluate undergraduate students at the University of Grenoble.

6.1 Protocol Description

The protocol involves candidates (C), an examiner (E), a question committee (QC), and an exam authority (EA). It has four phases:

Registration: In Grenoble exam each student has an identity (student name + her birthday), and a pseudonym (student number) which is assigned to her by the exam authority when she registered to the course. All the students of the course are automatically enrolled as eligible candidates for the exam; they are informed about the exam's date, time and location. The QC, the course's lecturer(s), prepares the questions and hands them securely to EA.

Examination: After EA authenticates all Cs, EA lets them take a seat. There, each C finds a special exam paper: the top-right corner is glued and can be folded. Each C signs it, and writes down her name in such a way that the corner, when folded, hides both the signature and the name. Each C also writes down visibly her pseudonym. EA checks that each C writes down her correct name and pseudonym, then the glued part can be folded and sealed. After that, EA distributes the questions and the exam begins. At the end, EA collects the exam-tests, checks that all copies have been returned, that all corners are correctly glued, and gives the exam-tests to E.

Marking: E evaluates the exam-tests: each pseudonym is given a mark. E returns them, along with the marks, to EA.

Notification: For each exam-test, EA checks that the corner is still glued and maps the pseudonym to the real identity without opening the glued part. Then, EA stores the pairs identities/marks, and publishes the pairs pseudonyms/marks. After that, C can review her exam-test in presence of E to check the integrity of her exam-test and verify the mark. If, for instance, C denies that the exam-test containing her pseudonym belongs to her, the glued part is opened.

6.2 Formal Analysis

We analyze Grenoble exam with ProVerif, using the equational theory depicted in Figure 5. The obtained results together with the time required for ProVerif to conclude, on the same PC used for the previous case studies, are resumed in Figure 6.

$$\begin{aligned}
checksign(sign(m,k),pk(k)) &= m\\
getmess(sign(m,k)) &= m\\
unfold(fold(m,k),k) &= m\\
authcheck(auth(m,s),genPublic(s)) &= m\\
openauth(auth(m,s)) &= m\\
seen(unseen(m,pk(k),r),k) &= m
\end{aligned}$$

Fig. 5: Equational theory to model Grenoble exam

We use the standard equational theory of digital signature (functions: $sign$, $checksign$ and $getmess$) to model candidate's signature. The function $fold$, similar to symmetric encryption, is used to hide candidate's identity and signature. The key necessary to reveal the hidden data using the function $unfold$ is included inside the message, so that anyone can unfold it. The function $auth$, similar to a signature, is used to model that everybody can see the other participants and thus authenticate them using a secret (corresponds to the physical identity). The attacker can get the content of the authenticated message using the function $openauth$. The authenticated message can be verified by applying the function $authcheck$, using a public value generated by the function $genPublic$. The function $unseen$, similar to asymmetric encryption, is used to model that the attacker cannot see the content of the exchanged messages. For instance when a candidate submits an answer, the others can see that she is submitting an answer but cannot look into its content (this is prevented by the authority which is controlling the exam room). The function $seen$ is the inverse of $unseen$. Note that, all the functions of Figure 5 are public functions, which can be applied by the attacker.

We use private channel for the transmission of the questions from QC to EA, as in reality this happen in a secure way (so nobody can see this transmission). Similarly, the authority provides a pseudonym (student number) to the candidate securely. We use a table to model this; EA inserts the identity of the candidate together with her pseudonym in the table, then the candidate gets it. Note that, in ProVerif, tables cannot be accessed by the attacker.

Also we assume that, an examiner cannot register as a candidate. This is normal since a candidate cannot be an examiner at the same time.

Authentication properties: ProVerif verifies that all the authentication properties are satisfied, if the parties that emits events (necessary for the considered property) are honest. This is necessary for authentication properties, since otherwise the processes may not emit some events when reached.

We make one assumption: the EA only accepts one exam-test per pseudonym. This is realistic as the authority collects only one exam copy from each candidate, which then has to leave the exam room. This assumption is necessary for *Answer Origin Authentication* to hold. Otherwise, the attacker can simply re-submit the candidate's exam-test, and thus the EA will collect twice the same exam-test from the same candidate. Hence, the property is destroyed.

Property	Result	Time
Answer Origin Authentication	✓(EA)	< 1 s
Form Authorship	✓(C, EA)	< 1 s
Form Authenticity	✓(E, EA)	< 1 s
Mark Authenticity	✓(C, E, EA)	< 1 s
Question Indistinguishability	✓(EA, QC)	< 1 s
Anonymous Marking	✓(C, E, EA)	< 1 s
Anonymous Marking*	✓(C, EA)	< 1 s
Anonymous Examiner	$\times$	< 1 s
Anonymous Examiner†	✓(E, EA)	< 1 s
Mark Privacy	$\times$	< 1 s
Mark Anonymity	✓(C, E, EA)	30 s

Fig. 6: Summary of our analysis on the formal model of the Grenoble protocol. (*) E corrupted, but cannot open the glued part. (†) private channel between EA and E.

Privacy properties: ProVerif shows that *Question Indistinguishability* is satisfied by Grenoble exam if QC and EA are honest. Otherwise, one of them could reveal the exam questions, and thus break its secrecy. *Anonymous Marking* is satisfied if EA, E, and the two candidates are honest. However, since it is desirable for *Anonymous Marking* to hold even if the examiner is corrupted, we also consider the case where we have a corrupted E. In that case we assume that the examiner still cannot open the glued part (which would trivially break *Anonymous Marking*), as this would be detectable. Given this assumption, ProVerif confirms that *Anonymous Marking* is satisfied by Grenoble exam even if E is corrupted. Concerning *Anonymous Examiner*, ProVerif finds a counterexample even if all parties are honest. The attacker can distinguish which "unseen" exam-test is accepted by the examiner to mark (the one he can "seen" using his secret key). This is not a real attack, since the examiner will only accept exam-tests from the exam authority, not an attacker. If we assume a private channel between the EA and E, ProVerif confirms that *Anonymous Examiner* is satisfied by Grenoble exam even with corrupted candidates. ProVerif finds an attack against *Mark Privacy* (when all parties are honest), this was expected as in Grenoble exam the marks are published in clear-text by the EA. However, *Mark Anonymity* is satisfied in case where we have honest EA, E and two Cs.

7 Conclusion

We define the first formal framework for the security analysis of traditional and electronic exam protocols. We show how to model exam protocols in the applied π-calculus, and define nine relevant security properties: four authentication properties and five privacy properties.

Using ProVerif, we analyze the security of two electronic exam protocols and one traditional exam. Our analysis shows that the e-exam proposed by Huszti *et al.* [8] indeed satisfies none of the nine properties. It security was only argued informally, while we show that authentication is compromised because of inaccuracies in the protocol design, and most of attacks invalidating privacy exploit a vulnerability in a component that the protocol uses, namely the RARC. The attacks compromise secrecy and anonymity of the messages taking advantage of the absence of a proof of knowledge of the submitted message to the RARC, a vulnerability that allows the attacker to use the RARC as a decryption oracle. Such a proof of security is not explicitly required in the original specification of the RARC, and is certainly missing in the H&P protocol: the "exam authority" is required to forward questions and answers without knowing them, and thus cannot prove knowledge of them when submitting them to the RARC. We proposed a few modifications on the H&P protocol in order to guarantee a subset of the authentication properties. ProVerif confirms that the modified protocol ensures these properties. However, even when assuming a perfect RARC ensuring anonymity, we still have attacks on privacy properties. Thus, we think that fixing the RARC is not sufficient – the protocol requires fundamental changes.

Also Remark!, the second protocol analyzed, has been only informally argued to be secure in the original paper. Our analysis identified a weakness that violates *Form Authenticity* when a candidate is corrupted. We propose a fix and formally verify that the (fixed) protocol satisfies all the properties herein considered.

The third protocol, Grenoble exam, is used at Grenoble University but never formally analyzed. Our analysis using ProVerif shows that it satisfies eight properties, and fails concerning *Mark Privacy*.

Generally speaking, our framework and our analysis bring exams into the attention of the security community. E-exams and in general computer-based assessment tools are becoming widespread; some of them supported by e-learning platforms such as the massive open online courses (MOOC). Nevertheless, they call for being formally proved secure, since most of them have not been subjected to any rigorous security analysis. Since they are complex systems and exposed to unprecedented cheating attacks, their vulnerabilities can be very subtle to be discovered. Often they are argued be secure only informally and the assumptions used in that argument, such as that authorities are trusted, are not explicitly stated; they may be even unjustified in reality. The same situation appears also for traditional exams. With this work we set the first research step on the formal understanding of such systems and establishes a framework for the automatic analysis of their security properties.

As a future work we intend to analyze more protocols designed for computer-based tests although obtaining protocol's specifications from the providers is not an easy task. Other interesting research directions include the definition of novel properties such as verifiability, reliability, and accountability for exams.

Acknowledgement. We would like to thank the authors of [8] for the helpful discussions on our findings concerning their protocol.

References

1. Hjeltnes, T., Hansson, B.: Cost Effectiveness and Cost Efficiency in E-learning. QUIS - Quality, Interoperability and Standards in e-learning, Norway (2005)
2. Weippl, E.: Security in E-learning. Volume 6 of Advances in Information Security. Springer Science + Business Media (2005)
3. Copeland, L.: School cheating scandal shakes up atlanta. `http://www.usatoday.com/story/news/nation/2013/04/13/atlanta-school-cheatring-race/2079327/` (April 2013)
4. Watson, R.: Student visa system fraud exposed in BBC investigation. `http://www.bbc.com/news/uk-26024375` (February 2014)
5. Abadi, M., Fournet, C.: Mobile values, new names, and secure communication. In: POPL, ACM (2001) 104–115
6. Blanchet, B.: An efficient cryptographic protocol verifier based on prolog rules. In: CSFW, IEEE Computer Society (2001) 82–96
7. Blanchet, B., Abadi, M., Fournet, C.: Automated verification of selected equivalences for security protocols. J. Log. Algebr. Program. **75**(1) (2008) 3–51
8. Huszti, A., Pethő, A.: A secure electronic exam system. Publicationes Mathematicae Debrecen **77** (2010) 299–312
9. Giustolisi, R., Lenzini, G., Ryan, P.Y.A.: Remark!: A secure protocol for remote exams. In: Security Protocols XXII - 22nd International Workshop Cambridge, UK, Revised Selected Papers. Volume 8809., Springer (2014) 38–48
10. Dreier, J., Giustolisi, R., Kassem, A., Lafourcade, P., Lenzini, G., Ryan, P.Y.A.: Formal analysis of electronic exams. In: SECRYPT 2014 - Proceedings of the 11th International Conference on Security and Cryptography, SciTePress (2014) 101–112
11. Castellà-Roca, J., Herrera-Joancomartí, J., Dorca-Josa, A.: A secure e-exam management system. In: ARES, IEEE Computer Society (2006)
12. Herrera-Joancomartí, J., Prieto-Blázquez, J., Castellà-Roca, J.: A secure electronic examination protocol using wireless networks. In: ITCC (2), IEEE Computer Society (2004)
13. Bella, G., Costantino, G., Coles-Kemp, L., Riccobene, S.: Remote management of face-to-face written authenticated though anonymous exams. In: CSEDU (2), SciTePress (2011)
14. Giustolisi, R., Lenzini, G., Bella, G.: What security for electronic exams? In: 2013 International Conference on Risks and Security of Internet and Systems (CRiSIS). (2013) 1–5
15. Furnell, S., Onions, P., Knahl, M., Sanders, P., Bleimann, U., Gojny, U., Röder, H.: A security framework for online distance learning and training. Internet Research **8**(3) (1998) 236–242
16. Dreier, J., Lafourcade, P., Lakhnech, Y.: Vote-independence: A powerful privacy notion for voting protocols. In: FPS. Volume 6888 of LNCS., Springer (2011) 164–180
17. Dreier, J., Lafourcade, P., Lakhnech, Y.: A formal taxonomy of privacy in voting protocols. In: ICC, IEEE (2012) 6710–6715
18. Dreier, J., Lafourcade, P., Lakhnech, Y.: Defining privacy for weighted votes, single and multi-voter coercion. In: ESORICS. Volume 7459 of LNCS., Springer (2012) 451–468

19. Backes, M., Hritcu, C., Maffei, M.: Automated verification of remote electronic voting protocols in the applied pi-calculus. In: CSF, IEEE Computer Society (2008) 195–209
20. Delaune, S., Kremer, S., Ryan, M.: Verifying privacy-type properties of electronic voting protocols. Journal of Computer Security **17**(4) (jul 2009) 435–487
21. Delaune, S., Kremer, S., Ryan, M.: Verifying properties of electronic voting protocols. In: Proceedings of the IAVoSS Workshop On Trustworthy Elections (WOTE'06), Cambridge, UK (jun 2006) 45–52
22. Dong, N., Jonker, H.L., Pang, J.: Analysis of a receipt-free auction protocol in the applied pi calculus. In: Proc. 7th Workshop on Formal Aspects in Security and Trust (FAST'10). Volume 6561 of LNCS., Springer (2010) 223–238
23. Dreier, J., Lafourcade, P., Lakhnech, Y.: Formal verification of e-auction protocols. In: POST. Volume 7796 of LNCS., Springer (2013) 247–266
24. Dreier, J., Jonker, H., Lafourcade, P.: Defining verifiability in e-auction protocols. In: ASIACCS, ACM (2013) 547–552
25. Arapinis, M., Bursuc, S., Ryan, M.: Privacy-supporting cloud computing by in-browser key translation. Journal of Computer Security **21**(6) (2013) 847–880
26. Dreier, J., Jonker, H., Lafourcade, P.: Secure auctions without cryptography. In: Fun with Algorithms - 7th International Conference. Volume 8496., Springer (2014) 158–170
27. Meadows, C., Pavlovic, D.: Formalizing physical security procedures. In: Proc. 8th workshop on Security and Trust Management (STM12). Volume 7783 of LNCS. (2013) 193–208
28. Basin, D., Capkun, S., Schaller, P., Schmidt, B.: Formal Reasoning about Physical Properties of Security Protocols. ACM Transactions on Information and System Security (2011) 1–28
29. Blaze, M.: Toward a broader view of security protocols. In: Proc. Security Protocols workshop 2004. Volume 3957 of LNCS., Springer Verlag (2006) 106–120
30. Dolev, D., Yao, A.C.: On the security of public key protocols. Information Theory, IEEE Transactions on **29**(2) (1983) 198–208
31. Delaune, S., Kremer, S., Ryan, M.D.: Coercion-resistance and receipt-freeness in electronic voting. In: Proceedings of the 19th IEEE Computer Security Foundations Workshop (CSFW'06), Venice, Italy, IEEE Computer Society Press (jul 2006) 28–39
32. Ryan, P.Y.A., Schneider, S.A., Goldsmith, M., Lowe, G., Roscoe:, A.W.: The Modelling and Analysis of Security Protocols: The CSP Approach. Addison-Wesley Professional (2000)
33. Ryan, M., Smyth, B.: Applied pi calculus. In: Formal Models and Techniques for Analyzing Security Protocols. IOS Press (2011)
34. Ryan, P.Y.A., Schneider, S.A.: Process algebra and non-interference. J. Comput. Secur. **9**(1-2) (January 2001) 75–103
35. Golle, P., Jakobsson, M.: Reusable anonymous return channels. In: Proc. of the 2003 ACM workshop on Privacy in the electronic society. WPES '03, ACM (2003) 94–100
36. Elgamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. Information Theory, IEEE Transactions on **31**(4) (1985) 469–472
37. Shamir, A.: How to share a secret. Commun. ACM **22**(11) (1979) 612–613
38. Backes, M., Maffei, M., Unruh, D.: Zero-knowledge in the applied pi-calculus and automated verification of the direct anonymous attestation protocol. In: IEEE Symposium on Security and Privacy, 2008. (2008) 202–215
39. Haenni, R., Spycher, O.: Secure internet voting on limited devices with anonymized dsa public keys. In: Proc. of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections. EVT/WOTE'11, USENIX (2011)