A Secure Exam Protocol Without Trusted Parties

Giampaolo Bella, Rosario Giustolisi, Gabriele Lenzini, Peter Ryan

A Secure Exam Protocol Without Trusted Parties

Giampaolo Bella[1], Rosario Giustolisi[2], Gabriele Lenzini[2⋆], and Peter Y. A. Ryan[2]

[1] Dipartimento di Matematica e Informatica, Università di Catania, Italy
[2] SnT, University of Luxembourg

Abstract. Relying on a trusted third party (TTP) in the design of a security protocol introduces obvious risks. Although the risks can be mitigated by distributing the trust across several parties, it still requires at least one party to be trustworthy. In the domain of exams this is critical because parties typically have conflicting interests, and it may be hard to find an entity who can play the role of a TTP, as recent exam scandals confirm. This paper proposes a new protocol for paper-based and computer-based exams that guarantees several security properties without the need of a TTP. The protocol combines oblivious transfer and visual cryptography to allow candidate and examiner to jointly generate a pseudonym that anonymises the candidate's test. The pseudonym is revealed only to the candidate when the exam starts. We analyse the protocol formally in ProVerif and prove that it satisfies all the stated security requirements.

1 Introduction

This paper considers written exams and studies how to guarantee secure and fair examination although any participant can cheat.

The security of exam protocols has been brought to the public attention by recent surveys and scandals [11, 14, 15]. They show that information technology makes cheating easier and that candidates and authorities have interest in frauds. For example, in the Atlanta public school scandal [11], the authorities raised all the markings to improve the school's ranking and get more public funds. In spite of that, previous exam systems still consider candidate's cheating as the sole security threat, while exam authorities and examiners are assumed to be fully trusted. A deeper understanding of the exam's threats would be also useful for similar assessment systems, such as public tenders, personnel selections, and project reviews. As in the case of exams, the security of such systems should not rely on TTP.

Recently a few works argued about the security of exam with corrupted examiners (e.g., [4, 16]); however, their designs still assume some trusted parties.

We propose a new security protocol for exams that requires no trusted party while meeting a set of stringent security properties that extend the requirements

⋆ Supported by CORE-FNR, project C11/IS/1183245 STAST.

for ones defined by Dreier et al. [8, 9]. Our protocol relies on oblivious transfer and visual cryptography techniques to generate a pseudonym that anonymises a candidate's test. No participant learns the pseudonyms until the exam starts. Candidates take the exam in a test center, and testing is the only face-to-face phase, while the other phases are remote. Our protocol suits both paper-based and computer-based examination.

Contribution. This paper provides three main contributions. First, it extends a set of security requirements for exams with three new authentication and one accountability property. Second, it proposes a new exam protocol that satisfies the extended requirements without relying on a TTP. Finally, it formalises the protocol in ProVerif and proves the protocol ensures all the properties.[1]

Outline. The paper is organized as follows. Section 2 outlines the related work. Section 3 describes and formalises the desired properties our protocol aims to ensure, and defines the threat model. Section 4 details the protocol. Section 5 describes the formal analysis of our protocol in ProVerif [5], and discusses the results. Section 6 outlines future work and concludes the paper.

2 Related work

The majority of works on exam protocols describe security requirements only informally (e.g., [22, 3]) with a few exceptions. Dreier *et al.* [8] propose a formal framework in the applied π-calculus to define and analyse authentication and privacy requirements for exams. They analyse two existing electronic exam protocols as case studies. Foley *et al.* [12] introduce a formalisation of confidentiality requirements for Computer Supported Collaborative Working. They propose exams as case study with no references to specific exam protocols.

Other works propose secure exam protocols, but argue informally their security. Castella-Roca *et al.* [6] introduce an exam protocol which guarantees a number of authentication and privacy properties in presence of a fully trusted exam manager. Bella *et al.* [4] describe WATA IV, a protocol that also relies on visual cryptography and considers corrupted examiner, but assumes an honest-but-curious anonymiser. Huszti and Pethő [16] propose an exam protocol with minimal trust requirements, but a trusted *registry*. Giustolisi *et al.* [13] describe *Remark!*, an internet-based exam protocol that ensures authentication and conditional anonymity requirements with minimal trust assumption. The protocol generates pseudonyms via an exponentiation mixnet, which assumes at least one honest mix server.

Maffei *et al.* [18] suggest anonymous credential schemes to guarantee privacy in course evaluation systems without relying on a TTP. Their approach seems to be alternative to ours, as we use oblivious transfer and visual cryptography.

Formal approaches have been proposed in the area of conference management systems, a domain close to exams. Arapinis *et al.* [2] propose and formally

[1] Our code is available at http://apsia.uni.lu/stast/codes/exams/pv_isec15.tar.gz

analyse ConfiChair, a cryptographic protocol that addresses secrecy and privacy risks coming from a malicious cloud. Their work has been recently extended to support any cloud-based system such as public tender management and recruitment process. Kanav *et al.* [17] introduce CoCon, a formally verified implementation of conference management system that guarantees confidentiality. All of the mentioned systems, however, assume trusted managers.

3 Security requirements and threat model

An exam basically involves two main roles: the *candidate*, who takes the test, and the *examiner*, who evaluates it. A typical exam runs in phases: at *preparation*, the exam is set up, for instance, the candidate registers and questions are generated; at *testing* the candidate gets the questions and takes the exam, while the examiner collects the answered test; at *marking* the examiner evaluates the test; at *notification*, the candidate is informed of her mark. Some duties may be assigned to sub-roles. For instance, an administrative office might ensure the registration of candidates, and notify them the marks.

We start considering the security requirements that other works argued to be relevant for exams [8, 4], and extend this set with five new requirements, including novel authentication and accountability properties. To express our requirements unambiguously, we use the applied π-calculus [1] as done in [8]. Therefore, we assume that our requirements refer to a model of an exam modelled as applied π-calculus process. The applied π-calculus defines events to formulate correspondence assertions (authentication), and uses observational equivalence to express indistinguishability (privacy).

Authentication In the applied π-calculus, an event is a message output $e(\vec{a})$: e is the event channel and $\vec{a}$ a, possible empty, list of arguments. The message appears in the trace as soon as the execution of the process reaches the event. To formalise authentication, we use some of the events defined in [8] as follows.

The term id_c is the candidate's identity, *pid* is the form identifier, *ques* indicates the exam questions, *ans* indicates the candidate's answers, *mark* is the mark. The event $reg(id_c)$ denotes a successful registration of the candidate id_c. The event $submitted(id_c, ques, ans, pid)$ is emitted when the candidate submits her answer, while $collected(id_c, ques, ans, pid)$ is emitted by the examiner when he collects the answer. Finally, the event $notified(id_c, mark, pid)$ is emitted when the examiner notifies and registers a mark to the candidate.

We add $requested(id_c, pid)$ to the set of events outlined above. It is emitted by the candidate process at notification, where candidate id_c sends the request to learn her mark using the identifier *form*.

The first requirement we consider is *Answer Authenticity*, which informally says that the examiner collects the answer as submitted by the candidate.

Definition 1 (Answer Authenticity) *An exam protocol ensures Answer Authenticity if the event* $collected(id_c, ques, ans, pid)$ *is preceded by the event* $submitted(id_c, ques, ans, pid)$ *in every execution trace of the protocol.*

Answer Origin Authentication says that the examiner only collects answers originated by registered candidates.

Definition 2 (Answer Origin Authentication) *An exam protocol ensures Answer Origin Authentication if the event collected*$(id_c, ques, ans, pid)$ *is preceded by the event reg*(id_c) *in every execution trace of the protocol.*

We define three novel authentication requirements: *Mark Authenticity*, *Candidate Authorisation*, and *Notification Request Authentication*.

Definition 3 (Mark Authenticity) *An exam protocol ensures Mark Authenticity if the event notified*$(id_c, mark, pid)$ *is preceded by the event submitted*$(id_c, question, answer, pid)$ *in every execution trace of the protocol*

Mark Authenticity says that the mark is correctly registered to the corresponding candidate. Dreier et al. defines it as "the candidate is notified with the mark delivered by the examiner" [8]. Despite looking similar, our formulation expresses something different: in ours the authenticator is the examiner because he emits the event *notified*, while in Dreier et al. the authenticator is the candidate, since the event is emitted in the candidate process. We think that Def. 3 avoids overlapping definitions because, as we shall see later, Mark Verifiability considers the candidate as authenticator. It describes that a candidate can check if she has been notified with the correct mark despite a corrupted examiner.

The second novel requirement, *Candidate Authorisation*, describes that only registered and authenticated candidates can take the exam.

Definition 4 (Candidate Authorisation) *An exam protocol ensures Candidate Authorisation if the event submitted*$(id_c, ques, ans, pid)$ *is preceded by the event reg*(id_c) *in every execution trace of the protocol.*

The last additional requirement is *Notification Request Authentication*. It says that a mark can be associated with the candidate only if she requests to learn her mark. This unusual requirement is useful in the exam scenarios of some universities, where candidates have to skip the next exam session if they get a fail, unless they withdraw from the exam before notification.

Definition 5 (Notification Request Authentication) *An exam protocol ensures Notification Request Authentication if the event notified*$(id_c, mark)$ *is preceded by the event requested*(id_c, pid) *in every execution trace of the protocol.*

Privacy For reason of space, we present only the informal definitions of the privacy requirements. The formal definition in applied π-calculus can be found in [8].

The first relevant privacy requirement is *Anonymous Marking*, which says that no one can learn the author of a test before it is marked. In other words, no one but the author can link the test with the candidate identity until after the marking phase. *Question Indistinguishability* says that no candidate learns

the question before the testing phase. A strong requirement is *Mark Privacy*, which describes that no one, besides the examiner and the concerned candidate, learns the marks. It implies that marks cannot be public. The last privacy requirement is *Mark Anonymity*, which says that no one, besides the examiner and the concerned candidate, can learn the mark assigned to a candidate. Note that Mark Privacy is intuitively stronger than Mark Anonymity: a system that publishes the marks cannot guarantee Mark Privacy while may still ensure Mark Anonymity provided no one can link a mark to a candidate identity.

Verifiability and Accountability We propose two properties, one for verifiability, and one for accountability of exams. Generally speaking, a protocol is verifiable with respect to a specific property if the protocol provides a test for the property, and the test is sound and complete [9]. *Mark Verifiability* says that the candidate can verify she has been notified with the mark assigned to her test. Mark Verifiability subsumes the existence of an algorithm `testMV` that outputs *true* if the candidate has been notified with the mark assigned to her test, or *false* otherwise. In the applied π-calculus, `testMV` is a process that emits the event $OK(id_c, pid, mark)$ when it is supposed to output *true* and KO when it supposed to output *false*. The event $published(pid)$ is emitted when a test identified with pid is available. The event $assigned(id_c, pid, mark)$ is emitted by the candidate at end of notification.

We say that `testMV` is sound if the event $OK(id_c, pid, mark)$ is preceded by the events $assigned(id_c, pid, mark)$ and $published(pid)$ in every execution trace of the protocol. We say that `testMV` is complete if the event KO is emitted in no execution trace of the protocol when the test fed with correct data.

Definition 6 (Mark Verifiability) *An exam protocol ensures Mark Verifiability if* `testMV` *is sound and complete.*

Finally, we introduce an accountability requirement, namely *Testing Dispute Resolution*. Accountability allows to identify which party is responsible for a protocol failure. In the case of exam, a candidate should be able to submit a test and receive the corresponding mark. If she fails in any of these, Testing Dispute Resolution describes that the participant who caused such failure can be identified.

We formally model Testing Dispute Resolution similarly to Mark Verifiability, with a difference: we use (non)reachability of the event *Cguilty* or *Eguilty* also to prove soundness. In the applied π-calculus, `dispute` is a process that emits the event *Cguilty* when the candidate is the culprit and *Eguilty* if the examiner is the culprit. If the protocol executes the process `dispute` then either the examiner or the candidate is corrupted. Thus, regarding soundness, the idea is to check that `dispute` cannot return an honest party instead of the corrupted one.

We say that `dispute` is sound with respect to a corrupted examiner and honest candidate if the event *Cguilty* is emitted in no execution trace of the protocol. Similarly, we say that `dispute` is sound with respect to a corrupted

candidate and honest examiner if the event *Eguilty* is emitted in no execution trace of the protocol. Finally, we say that `dispute` is complete if neither the event *Eguilty* nor *Cguilty* are emitted in any execution trace of the protocol with honest roles.

Definition 7 (Testing Dispute Resolution) *An exam protocol ensures Testing Dispute Resolution if* `dispute` *is sound and complete.*

3.1 Threat model and assumptions

According our exam terminology, we consider the threats coming from the three following adversaries. (1) Corrupted candidates, who want to be assigned with a mark higher than an objective evaluation of their answers deserve. They thus may not follow the protocol and collude each other to achieve their goal. (2) A corrupted examiner, who wants to assess a candidate unfairly. (3) An intruder, who wants to get exam's private information or tamper with tests and marks, and may corrupt candidates or the examiner.

We assume that (a) remote communications between examiner and candidate occur via TLS; (b) model answers are kept secret from candidates until after testing; (c) during the testing, invigilators supervise candidates to mitigate cheating, and (d) an authenticated append-only bulletin board is available.

4 The protocol

In a nutshell, the protocol works as follows. At preparation, candidate and examiner jointly generate the candidate's pseudonym (an alphanumeric *pid*) as a pair of visual cryptography shares, by means of an oblivious transfer scheme. One share is hold by the candidate, who prints it on a paper sheet together with the candidate ID and signatures meant for integrity and accountability purposes. The other share is held by the examiner, who prints it on a transparency printout. Each share alone does not reveal the pseudonym, which is revealed only when the shares are overlapped. This is possible only at testing, when the candidate and the examiner physically meet, and the examiner hands his transparency to the candidate. Any dispute that happens at testing can be solved thanks to the signatures printed with the printouts. The candidate can write the pseudonym down into the answer sheet, and the testing concludes when all answer sheets are returned to the examiner. At marking, the examiner evaluates the answers, and assigns a mark to each pseudonym, which she commits and publishes on a bulletin board. At notification, a candidate can retrieve her mark by proving she owns the share that (re)-reveals the pseudonym. The examiner's share is required for this phase, but there is no need for the candidate and the examiner to meet. The candidate sends her share and the signatures to the examiner, and any dispute happening at notification can be again solved using the signatures associated with the shares.

The protocol combines a few cryptographic primitives, namely visual cryptography, commitment, and oblivious transfer schemes:

1. *Candidate* calculates $y_i = g^{x_i} h^{\gamma_i}$ where:
   - $x_i \in_{\mathcal{R}} \mathbb{Z}_q^*$.
   - $\gamma_i \in_{\mathcal{R}} [1, k]$.
   - $i = 1, 2, \ldots, l$ with $l > n$.
2. *Candidate*→*Examiner*: $y_1, y_2, \ldots, y_l$.
3. *Examiner* calculates $\beta_{ij} \leftarrow_{\pi_{\mathcal{R}}} (\alpha_i \oplus c_j)$, $\omega_{ij} = \langle a_{ij}, b_{ij} \rangle \leftarrow \langle g^{r_{ij}}, \beta_{ij} \left(\frac{y_i}{h^j}\right)^{r_{ij}} \rangle$, $com = h^s \prod_{i=1}^{l} {g_i}^{\alpha_i}$, and $sign1 = Sign_{SSK_E}\{idC, ex, com\}$ where:
   - $\alpha_i \in_{\mathcal{R}} [0, 1]^{t \times u}$.
   - $s, r_{ij} \in_{\mathcal{R}} \mathbb{Z}_q^*$.
   - $g_i \in_{\mathcal{R}} \mathbb{G}_q$.
   - $i = 1, 2, \ldots, l$.
   - $j = 1, 2, \ldots, k$.

   or runs the challenge procedure against $y_1, y_2, \ldots, y_l$.
4. *Examiner*→*Candidate*: $(\omega_{11}, \ldots, \omega_{1k}), \ldots (\omega_{l1}, \ldots, \omega_{lk})$ and *sign1*.
5. *Candidate* calculates $\chi_i \in [1, l]$ and $\sigma_j \in [1, l]$ where:
   - $i = 1, 2, \ldots, m$.
6. *Candidate*→*Examiner*: $\chi_1, \chi_2, \ldots, \chi_m$ and $\sigma_1, \sigma_2, \ldots, \sigma_n$.
7. *Examiner* calculates $ev_{\chi_i} = \langle \alpha_{\chi_i}, (\beta_{\chi_i 1}, \beta_{\chi_i 2}, \ldots, \beta_{\chi_i k}), (r_{\chi_i 1}, r_{\chi_i 2}, \ldots, r_{\chi_i k}) \rangle$ and $sign2 = Sign_{SSK_E}\{idC, ex, (\sigma_1, \sigma_2, \ldots, \sigma_n)\}$ where
   - $i = 1, 2, \ldots, m$.
   - $j = 1, 2, \ldots, k$.

   and prints $transp = \langle (\alpha_{\sigma_1}, \alpha_{\sigma_2}, \ldots, \alpha_{\sigma_n}), idC, ex, QR3 \rangle$ where
   - $QR3 = idC, ex, (\alpha_1, \alpha_2, \ldots, \alpha_l, s)$.
8. *Examiner*→*Candidate*: $ev_{\chi_1}, ev_{\chi_2}, \ldots, ev_{\chi_m}$ and *sign2*.
9. *Candidate* checks ev_{χ_i}, calculates $\beta_{\sigma_j} = \frac{b_{\sigma_j \gamma_j}}{(a_{\sigma_j \gamma_j})^{x_{\sigma_j}}}$ where
   - $i = 1, 2, \ldots, m$.
   - $j = 1, 2, \ldots, n$.

   and prints $paper = \langle (\beta_{\sigma_1}, \beta_{\sigma_2}, \ldots, \beta_{\sigma_n}), idC, ex, QR1, QR2 \rangle$ where
   - $QR1 = idC, ex, sign1$.
   - $QR2 = idC, ex, sign2$.

10. *Candidate* $\xrightarrow{hands}$ *Examiner*: idC'
11. *Examiner* checks if $idC' = idC$
12. *Examiner* $\xrightarrow{hands}$ *Candidate*: $transp, test\_question$
13. *Candidate* calculates $pid = (\alpha_1, \alpha_2, \ldots, \alpha_n) \oplus (\beta_1, \beta_2, \ldots, \beta_n)$ and writes $test\_answer = (answers, pid)$

    or runs the Testing Dispute Resolution algorithm if no pseudonym appears.
14. *Candidate* $\xrightarrow{hands}$ *Examiner*: $test\_answer$

15. *Examiner* calculates $c = g^v h^{mark}$ and $sign3 = Sign_{SSK_E}\{pid, c\}$ where:
    - $v \in_{\mathcal{R}} \mathbb{Z}_q^*$.
    - $mark \in M$.
16. *Examiner*→ $\mathcal{BB}$: *sign3*
17. *Candidate*→*Examiner*:$(\beta_1, \beta_2, \ldots, \beta_n)$, *sign1*, *sign2*, *sign3*
18. *Examiner* calculates $sign4 = Sign_{SSK_E}\{idC, ex, pid, mark, v\}$
19. *Examiner*→*Candidate*: *sign4*

Fig. 1. Our protocol divided in phases

Visual Cryptography. It is a secret sharing scheme, first devised by Naor and Shamir [19] that allows a visual decryption of a ciphertext. A secret image is "encrypted" by splitting it into a number of image *shares*. In the 2-out-of-2 version, which is the one adopted in our protocol, the secret image is split into two shares. When the shares are overlapped, they reveal the secret image. Many schemes for visual cryptography have been proposed over the years. We use the Naor and Shamir scheme for our protocol, but we conjecture that any other visual scheme can be used as well.

Commitment schemes. A commitment scheme is used to bind a committer to a value. The committer publishes a commitment that hides a value, which remains secret until the committer reveals it. Should he reveal a different value, this would be noticed, because two identical commitments hide the same value. Our protocol uses a generalized Pedersen commitment scheme [20], which guarantees unconditional hiding and allows the commitment to many values at once.

Oblivious transfer. Oblivious transfer schemes allow a chooser to pick some pieces of information from a set a sender offers him, in such a way that (a) the sender does not learn which pieces of information the choosers picks, and (b) the chooser learns no more than the pieces of information he picks. Our protocol adopts Tzeng's oblivious transfer scheme [21]. In Tzeng's scheme, the chooser commits to some elements from a set, and sends the commitments to the sender. This, in turn, obfuscates all the set's elements, and the chooser will be able to de-obfuscate only the elements he has committed to. It guarantees unconditional security for the receiver's choice, and it is efficient since it works with the sender and receiver's exchanging only two messages.

4.1 Description of the protocol in detail

We describe our protocol in reference to the four exam phases. In the description we assume a few public parameters, namely:

n	length of the candidate's pseudonym
$C = \{s_1, \ldots, s_k\}$	alphabet of pseudonym's characters
$c_j \in \{0,1\}^{t \times u},\ j = 1, \ldots, k$	$(t \times u)$-pixel representation of a character
idC	candidate ID
ex	exam code
SPK_E	examiner's public key
M	set of possible marks
$g, h \in_{\mathcal{R}} \mathbb{G}_q$	commitment generators

Preparation. The goal of preparation is to generate a candidate's pseudonym, which is a string of n characters taken from alphabet C, and to encode it into two visual cryptographic shares. Both candidate and examiner cannot know the pseudonym until they meet at testing, when the candidate learns her pseudonym by overlapping the examiner's share with hers. The underlying idea is that the

candidate provides a commitment to an index into an array. The examiner fills the array with a secret permutation of the characters, and only when the two secrets are brought together is the selection of a character determined.

This phase is inspired by one of the schemes used to print a secret, proposed by Essex *et al.* [10]. We tailor the scheme in such a way to be able to generate a pseudonym. More specifically, we extend it to support an algorithm to resolve a dispute that may arise when the overlapping of the shares will not reveal any intelligible pseudonym. The main technical differences between our preparation and the original scheme are: (a) a modified oblivious transfer protocol that copes with several secret messages in only one protocol run; (b) the generation of signatures that will be used for accountability in the resolution of disputes.

Figure 1 gives the description of the steps of preparation. The protocol begins with the candidate providing a sequence of l commitments y_i to an index into an array of length k. (steps 1-2).

In detail, the parameter l, is chosen so that the $l - n$ elements can be later used for a cut-and-choose audit. The examiner can challenge the candidate to check whether the committed choices are in fact in the interval $[1, k]$. Otherwise, the examiner generates a sequence of randomly chosen $t \times u$ images, indicated as $\alpha_1, \ldots, \alpha_l$ in Figure 1. A sequence of k images, $(\beta_{i1}, \ldots, \beta_{ik})$, are generated from α_i and each possible character c_j. The sequence is randomly permuted and repeated for all i, resulting in l sequences of $(\beta_{11}, \ldots, \beta_{1k})$, $\ldots$, $(\beta_{l1}, \ldots, \beta_{lk})$. The secret permutation and the commitment allow that the selection of character is determined only when the two secrets are brought together.

The examiner then generates the obfuscation ω_{ij} from each β_{ij} and generates a commitment on each α_i, indicated as *com* (step 3), which is signed and sent with the sequences of obfuscations $(\omega_{11}, \ldots, \omega_{1k})$,$\ldots$, $(\omega_{l1}, \ldots, \omega_{lk})$ to the candidate (step 4). The obfuscation allows the candidate to retrieve only the elements whose indexes correspond to the choices she committed in step 1 (y_i).

The candidate performs a cut-and-choose audit, selecting a random set of $l-n$ sequences amongst the ω. Doing so, she can check whether the examiner generated the sequence of images correctly. The remaining substitutions $\sigma_1, \sigma_2, \ldots, \sigma_n$ select the indexes of the images that make the pseudonym. Thus, the visual share of the examiner consists of the concatenated images $(\alpha_{\sigma_1}, \ldots, \alpha_{\sigma_n})$ (step 5-6).

The examiner then generates the proofs for the cut-and-choose audit, and prints the visual share in a transparency printout. This also include all the elements $\alpha_1, \ldots, \alpha_l$ and the value used for their commitment (step 7), which are stored in the form of QR code. The examiner then sends the proofs and the signed substitutions σ to the candidate (step 8). In turn, the candidate checks the proofs, de-obfuscates the elements ω, and retrieves the visual share consisting of the concatenated image $(\beta_{\sigma_1}, \beta_{\sigma_2}, \ldots, \beta_{\sigma_n})$. She finally prints the share, together with the two signatures, on a paper printout (step 9). At this point, both candidate and examiner have a visual share, which once overlapped reveal an intelligible sequence of characters that serves as pseudonym.

The candidate's paper printout includes two QR codes (*QR1*, and *QR2*) while the examiner's transparency only one (*QR3*). All the three QR codes share the

same candidate identity idC and exam identifier ex. *QR1* and *QR2* encode the two signatures of the examiner, respectively on commitment of the elements α and on the substitutions σ, while *QR3* encodes the elements α.

Testing. The candidate brings the paper printout, and the examiner the transparencies. The examiner authenticates the identity of the candidate by checking her identity document (step 10-11). He then gives the candidate her corresponding transparency and a copy of the questions (step 12). The candidate overlaps her paper printout with the transparency, and learns her pseudonym, which writes it on the answer sheet (step 13). If no pseudonym appears, then this may happen only if the candidate or the examiner misprinted their printouts, and the Testing Dispute Resolution outlined in Algorithm 1 reveals the party that is accountable for the misbehaviour. At the end of the phase, the candidate returns the answer sheet anywhere in the pile of tests (step 14), and takes both transparency and paper printouts home.

Marking and Notification. At marking the examiner evaluates the anonymous tests; at notification, the candidate to learn her mark, but only if she wants to. The examiner evaluates the answers and generates a commitment on the assigned mark (step 15). Then, he signs both mark and pseudonym found on the answer sheet, and publishes the signature on a bulletin board (step 16).

Notification opens for a fixed time, during which the candidate can remotely request to learn and register her mark. She has to send the ordered sequences of $\beta_1, \ldots, \beta_n$ and all the signatures so far she collected to examiner (step 17). The examiner checks the signatures, overlaps the given sequence with the corresponding sequences of $\alpha_1, \ldots, \alpha_n$, and learns the pseudonym. Again, if no registered pseudonym appears, Dispute Resolution reveals the party who misbehaved. The examiner signs the mark and the secret parameter used to commit the mark (step 18), and sends the signature to the candidate (step 19). In so doing, the candidate can verify the assigned mark against the bulletin board.

Dispute resolution. A corrupted examiner may misprint the visual share printed on the transparency. Thus, the candidate retrieves no intelligible pseudonym when she overlaps the visual shares, making her answers impossible to be anonymous. On the other hand, a corrupted candidate may misprint her paper printout and charge the examiner for misprinting the transparency. Should, such a dispute would arise, Algorithm 1 provides an efficient way to find the culprit.

We assume that the invigilator has an electronic device with a camera, such as a smart phone or tablet, which stores the public key of the examiner. The input of the Algorithm are the two QR codes printed on the paper printout (QR1 and QR2) and the QR code printed on the transparency (QR3), which the invigilator scans with the device camera.

First, the algorithm checks the correctness of the signatures encoded in QR1 and QR2. It also checks whether the candidate identity and the exam identifier

Data: Public parameters: $(C, n, g_i, h, idC, SPK_E)$

- *paper* = $((\beta_{\sigma_1}, \beta_{\sigma_2}, \ldots, \beta_{\sigma_n}), idC', ex', sign1, sign2)$ where:
  - *sign1* = $Sign_{SSK_E}\{idC'', ex'', com\}$
  - *sign2* = $Sign_{SSK_E}\{idC''', ex''', (\sigma'_1, \sigma'_2, \ldots, \sigma'_n)\}$
- *transp* = $(\alpha_{\sigma''_1}, \alpha_{\sigma''_2}, \ldots, \alpha_{\sigma''_n}), idC', ex', (\alpha'_1, \alpha'_2, \ldots, \alpha'_l, s)$.

Result: Corrupted participant
if *sign1 is verifiable with* SPK_E ***and*** *sign2 is verifiable with* SPK_E ***and*** $idC = idC' = idC'' = idC'''$ ***and*** $ex = ex' = ex'' = ex'''$ **then**
  if $com \neq h^s \prod_{i=1}^{l} {g_i}^{\alpha'_i}$ ***or*** $pid{=}(\alpha'_{\sigma'_1}, \alpha'_{\sigma'_2}, \ldots, \alpha'_{\sigma'_n}) \oplus (\beta_{\sigma_1}, \beta_{\sigma_2}, \ldots, \beta_{\sigma_n})$ **then**
    return Examiner
  else
    return Candidate
else
  return Candidate

Algorithm 1: Dispute resolution

reported on the paper printout match the ones in QR1 and QR2. If any one of the checks fails then the candidate misprinted her paper printout thus she is the culprit. Otherwise, the algorithm uses the data in QR3 to check the correctness of the examiner's commitment and that no pseudonym appears using the α elements indexed with the σ substitutions encoded in QR3. If any one of these checks fails then the examiner misprinted the transparency and thus he is guilty, otherwise the candidate is the culprit.

5 Analysis

We analyse our protocol in ProVerif, a security protocol verifier that allows the automatic analysis of authentication and privacy properties in the Dolev-Yao model [7]. The input language of ProVerif is a variant of the applied π-calculus.

5.1 Modelling Choices

No private channels. We model TLS and face-to-face communications among the roles using shared key cryptography rather than private channels. This choice is motivated because the attacker cannot monitor communications via ProVerif's private channels, and even know if any communication happens. We think this is a too strong assumption that may miss attacks. By renouncing to private channels, we achieve stronger security guarantees when analysing our protocol. Moreover, our choice has a triple advantage: it allows the attacker to learn when a candidate registers for the exam or is notified with a mark; it suffices to share the key with the attacker when either the candidate or the examiner is corrupted; it increases the chance the ProVerif verification terminates. Thus, the attacker has more discretional power because he can observe when a candidate is given the questions and when she submits the answers.

Data: Public parameters: (g, h, SPK_E)
- $sign3 = Sign_{SSK_E}\{pid, c\}$
- $idC, pid', mark, v$.

Result: Whether the candidate was notified with the mark assigned to her test.
if $pid = pid'$ ***and*** $c = g^v h^{mark}$ **then**
  return *true*
else
  return *false*

Algorithm 2: The `testMV` for our protocol

Equational theory. We use the following equational theory to model the cryptographic primitives needed in our protocol.

Probabilistic symmetric key	$sdec(senc(m, k, r), k) = m$
Signature	$getmess(sign(m, ssk)) = m$ $checksign(sign(m, ssk), spk(ssk)) = m$
Visual cryptography	$overlap(share, gen\_share(m, share)) = m$ $overlap(share, share) = share$
Obfuscation	$deobf(obf(r, m, \mathtt{sel}, commit(r', \mathtt{sel})), r') = m$

The theory for probabilistic symmetric key and signature specifications are well-known in ProVerif. We introduce a novel theory to model oblivious transfer and visual cryptography. The function *obf* allows the examiner to obfuscate the elements $\beta_1, \ldots, \beta_i$, while the function *deobf* returns the correct element β_{sel} to the candidate, depending on the choice she committed. We also provide the theory for the Pedersen commitment scheme with the function *commit*. Finally, we model the generation of a visual cryptography share with *gen_share*, and their overlapping with the function *overlap*.

We verify Anonymous Marking in presence of a corrupted examiner. We add the process *collector* that simulates the desk where candidates leave their tests. Question Indistinguishability considers corrupted candidates, while Mark Privacy and Mark Anonymity both consider corrupted eligible candidates. To analyse Mark Verifiability, we define the algorithm `testMV` for our protocol as depicted in Algorithm 2. We model an honest candidate, corrupted examiner and co-candidates to prove the soundness of `testMV`. In particular, we use correspondence assertions to verify the soundness of the algorithm in ProVerif, and (non)reachability of the event *KO* to verify completeness. We check the two soundness properties that regard Testing Dispute Resolution considering a corrupted examiner in one, and corrupted candidates in the other.

A limitation of the formal model is the specification of the cut-and-choose audit due to the powerful ProVerif's attacker model. In fact, if the attacker plays the cutter's role, he might cut the set of elements such that the subset audited by the chooser is correct, while the other subset not. Although in reality the probability of success of this attack for a large set of elements is small, it is a valid attack in ProVerif irrespective of the number of elements. In our case, the

chooser is the candidate and the cutter the examiner. We thus have a false attack when the examiner is corrupted, namely controlled by the attacker. In this case, we avoid this situation by allowing the candidate to check all the elements of the set. This is sound because the candidate plays the role of the chooser, thus she is honest and follows the protocol although she knows the extra information.

Results. Table 1 outlines the results of our analysis. ProVerif confirms that our protocol guarantees all the authentication properties despite allowing an unbounded number of corrupted eligible co-candidates. Thus, our properties hold although the attacker can register to the exam. Concerning privacy properties, ProVerif proves that our protocol guarantees Anonymous Marking, Question Indistinguishability, Mark Privacy, and Mark Anonymity. Finally, our protocol is Mark Verifiable because `testMV` is sound and complete, and ensures Testing Dispute Resolution: ProVerif shows that our protocol charges the misbehaving party and not the honest, if the `dispute` algorithm is executed (soundness), and the algorithm is not executed when both examiner and candidate roles are honest (completeness).

Property	Result	Time
Candidate Authorisation	✓	8s
Answer Authenticity	✓	7s
Answer Origin Auth.	✓	7s
Notification Request Auth.	✓	8s
Mark Authenticity	✓	8s

Property	Result	Time
Anonymous Marking	✓	27s
Question Indist.	✓	<1s
Mark Privacy	✓	28m 41s
Mark Anonymity	✓	52m 12s
Mark Verifiability	✓	<1s
Testing Dispute Res.	✓	<1s

Table 1. The result of the formal analysis in ProVerif with a machine Intel i7, 8GB

6 Conclusion and Future Work

We propose a new protocol for exams without the requirement of a trusted role. The underlying idea is to combine oblivious transfer and visual cryptography to generate a pseudonym which anonymises the test for the marking. A formal analysis in ProVerif confirms the protocol ensures all the stated properties.

As future work we intend to extend our design to yield a larger set of verifiability properties. Moreover, to extend the application scenarios of our protocol, we intend to modify the notification phase in order to avoid the involvement of the candidate at notification. To achieve this, we envisage a temporal deanonymization solution similar to the one in Remark! [13]. Regarding the formal analysis, we aim to study compositional proofs that integrate computational proofs of the cryptographic primitives used in our protocol with the symbolic ones obtained in ProVerif. Finally, we intend to implement a prototype of the

protocol, and verify if different visual cryptography schemes can be used to increase the perceptual security of an examination.

References

1. Abadi, M., Fournet, C.: Mobile Values, New Names, and Secure Communication. In: POPL 01. ACM (2001)
2. Arapinis, M., Bursuc, S., Ryan, M.: Privacy-supporting cloud computing by in-browser key translation. J. of Computer Security 21(6), 847–880 (2013)
3. Auernheimer, B., Tsai, M.: Biometric Authentication for Web-Based Course Examinations. In: HICSS '05. p. 294b. IEEE (2005)
4. Bella, G., Giustolisi, R., Lenzini, G.: Secure Exams Despite Malicious Management. In: PST '14. pp. 274–281. IEEE (2014)
5. Blanchet, B.: An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In: CSFW '01. pp. 82–96. IEEE (2001)
6. Castella-Roca, J., Herrera-Joancomarti, J., Dorca-Josa, A.: A Secure E-Exam Management System. In: ARES 2006. IEEE (2006)
7. Dolev, D., Yao, A.C.: On the Security of Public Key Protocols. IEEE Trans. on Information Theory 29(2), 198–208 (1983)
8. Dreier, J., Giustolisi, R., Kassem, A., Lafourcade, P., Lenzini, G., Ryan, P.Y.A.: Formal Analysis of Electronic Exams. In: SECRYPT 2014. SciTePress (2014)
9. Dreier, J., Giustolisi, R., Kassem, A., Lafourcade, P., Lenzini, G.: On the verifiability of (electronic) exams. Tech. Rep. TR-2014-2, Verimag (2014)
10. Essex, A., Clark, J., Hengartner, U., Adams, C.: How to Print a Secret. In: HotSec 09. USENIX Association (2009)
11. Flock, E.: APS embroiled in cheating scandal. Washington Post, July, 2011
12. Foley, S.N., Jacob, J.L.: Specifying Security for Computer Supported Collaborative Working. J. of Computer Security 3, 233–253 (1995)
13. Giustolisi, R., Lenzini, G., Ryan, P.: Remark!: A Secure Protocol for Remote Exams. In: Security Protocols XXII, LNCS, vol. 8809, pp. 38–48. Springer (2014)
14. Guénard, F.: La Fabrique des Tricheurs: La fraude aux examens expliquée au ministre, aux parents et aux professeurs. Jean-Claude Gawsewitch (2012)
15. Hallak, J., Poisson, M.: Corrupt Schools, Corrupt Universities: What Can be Done? Ethics and corruption in education, Education Planning, UNESCO (2007)
16. Huszti, A., Pethö, A.: A secure Electronic Exam System. Publicationes Mathematicae Debrecen 77(3-4), 299–312 (2010)
17. Kanav, S., Lammich, P., Popescu, A.: A Conference Management System with Verified Document Confidentiality. In: CAV 2014, LNCS, vol. 8559, pp. 167–183. Springer (2014)
18. Maffei, M., Pecina, K., Reinert, M.: Security and Privacy by Declarative Design. In: CSF 2013. pp. 81–96. IEEE (2013)
19. Naor, M., Shamir, A.: Visual cryptography. In: EUROCRYPT (1994)
20. Pedersen, T.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: EUROCRYPT 91, LNCS, vol. 576, pp. 129–140. Springer (1991)
21. Tzeng, W.G.: Efficient 1-out-of-n Oblivious Transfer Schemes with Universally Usable Parameters. IEEE Trans. on Computers 53(2), 232–240 (2004)
22. Weippl, E.: Security in E-Learning, Advances in Information Security, vol. 16. Springer (2005)