Remark!: A Secure Protocol for Remote Exams

Rosario Giustolisi, Gabriele Lenzini, and Peter Y.A. Ryan

SnT - University of Luxembourg

Abstract. This paper is about secure remote examination. It presents *Remark!*, an electronic exam protocol which achieves several authentication, (conditional) anonymity, privacy, and verifiability properties without trusted third parties. *Remark!* is primarily designed for invigilated Internet-based exams but it also fits computer-based exams with candidates taking their exam in classrooms.

1 Introduction

There is a growing requirement to evaluate skills of people remotely and hence increase in interest in how to design secure remote electronic exams (in short, e-exams).

Traditional exams consist at least of four phases: registration, testing, marking, and notification. During the registration phase, a new exam is arranged, usually by a manager, and candidates enrol for it. During the testing phase, the candidates receive and take a test and submit their answers. During the marking phase, examiners assess the answers and assign a mark. Finally, during the notification phase, candidates learn their marks.

E-exams are organized similarly, and with the same principals involved in running the e-exam: candidates, one or more examiners, and a manager. The role of a candidate and that of an examiner are obvious. The role of the manager is to register eligible candidates and examiners for an examination, to assign the test questions to the candidates and, once they have submitted their answers, to distribute the answered test to examiners and gather the marks. Finally, the manager notifies the candidates with marks. Depending on the specific implementation, the manager's duties can be further split among other principals, such as question committee, invigilator, collector, and notifier.

To our knowledge only few works propose exam protocols with security in mind. Castella-Roca *et al.* [5] propose an e-exam system with a fully trusted manager. Huszti & Petho [2] propose an e-exam scheme with fewer trust requirements on principals.

Contribution. This paper first identifies threats and requirements for e-exams, and proposes *Remark!*, an e-exam protocol primarily designed for invigilated internet-based exams. Besides the protocol also suits computer-based testing where candidates take the exam at examination venue, such as a classroom or a test centre. Our protocol achieves authentication, verifiability, and (conditional) anonymity without relying on trusted parties.

2 Threat Model, Security Requirements, and Assumptions.

Threats. E-exams are subject to threats from outsiders as well as from insiders, since each role has different capabilities and incentives to misbehave. *Remark!* is designed to withstand the following threats:

1. An intruder impersonating a candidate during the testing.
2. An intruder tampering with a candidate's test answer and mark.
3. A candidate trying to get an higher mark than that deserved (overmarked).
4. A candidate trying to discover who is the examiner evaluating her test.
5. The manager tampering with the marks.
6. The examiner trying to assign a biased mark to a submitted test.
7. An examiner colluding with a candidate to overmark her test answer.

Security Requirements. We have identified several fundamental security requirements that a secure e-exam should fulfil. The list outlined above takes inspiration and extends the requirements described in [3] :

p1: ***Test Answer Authentication*****:** the manager only accepts test answers submitted by the registered candidates. This means that the candidate, the test assigned to her, and their association should be authenticated and preserved, for instance against collusion among candidates.

p2: ***Examiner Authentication*****:** the manager only accepts evaluations made by a registered examiner. This rather obvious requirement means that the mark assigned to a test answer is authentic.

p3: ***Anonymous Marking*****:** no one learns the author of a test answer before it has been marked. This requirement states that only the candidate who wrote the answers knows the association between her identity and the test. Notably, this should resist to collusion between examiner and manager.

p4: ***Anonymous Examiner*****:** no candidate learns the identity of the examiner who evaluates their test answers. This requirement ensures that no candidate can coerce an examiner before and after he evaluates her test answer.

p5: ***Question Secrecy*****:** no candidate learns the test questions before the testing phase begins. This ensures a desirable grade of fairness among candidates as no one knows the questions in advance, provided that no one is illegitimately allowed to know the answers beforehand.

p6: ***Question Privacy*****:** the manager does not learn which test question is assigned to a specific candidate. This requirement ensures that the manager cannot identify a candidate by looking at its questions once it has been submitted for evaluation.

p7: ***Mark Privacy*****:** the candidate learns only her mark and not those of other candidates. This is a rather standard requirement, despite not everywhere applied, meaning that the mark of a test is known only by the author of test, and possibly by the manager, who may need it for registering the mark.

p8: *Test Verifiability*: the candidate can verify that her test is considered for evaluation. This requirement states that the candidate has a way to check whether her submitted test has been accepted by the manager.

p9: *Mark Verifiability*: the candidate can verify that the manager registers the mark assigned to her by the examiner. This ensures that the candidate notices if the mark assigned to her test is different from the one registered by the manager.

Assumptions. Our design and our analysis rely on the following assumptions:

1. Each principal is given a public/private pair of keys.
2. The candidate holds a smart card, carrying the candidate's identity visibly engraved, that stores her private key security (*i.e.*, it cannot be extracted).
3. To mitigate plagiarism, candidates are invigilated during the testing. This can be done remotely by using software such as ProctorU [1]).
4. The model answers are kept secret from the candidates until after the exam has completed. Note: the examiners may be provided with the model answers.
5. An authenticated, append-only, bulletin board is available. On it, everyone is guaranteed to see the same data. Write access will be restricted however to the appropriate entities. (*e.g.*, see [4]).
6. Secure and private channels, such as SSL, are available.

3 The Protocol

Remark! relies on several servers that implement an *exponentiation mixnet* [6]. The peculiarity of this kind of mixnet is that each mix server blinds its entries by a common exponent value in contrast to a conventional re-encryption mixnet. Here, it is assumed that at least one server among the ones in a mixnet behaves honestly. Thus, if the mixnet is made of m servers and r_i is the exponent value of ith server, then the mixnet, once given the input X, outputs $X^{\overline{r}_m}$, where $\overline{r}_m = \prod_{i=1}^{m} r_i$.

Remark! makes use of exponentiation mixnet at registration, to create the pseudonyms for the candidates and examiners. The mixnet is required also at notification, to revoke the candidates pseudonyms and retrieve the candidates' identities. (The generation of pseudonyms for candidates is separated from that for examiners because, at notification, only the identities of candidates should be revealed.)

A *bulletin board* (bulletin board) is used to publish the pseudonyms, the questions, the tests and the marks. The bulletin board is also used by the mixnet's servers to publish their intermediate shuffling (see later). In so doing anyone can check the authenticity of each mix step.

The following paragraphs detail how to use exponentiation mixnets to generate pseudonyms, and describe all the phases of *Remark!*. The protocol's steps are also illustrated synthetically in Appendix in form of a message sequence chart. In the reminder, $\langle X_i \rangle$ is a shorthand for the list $\langle X_1, \ldots X_n \rangle$, and $\overline{r}_k$ is a shorthand for $\prod_{i=1}^{k} r_i$ (so, $\overline{r}_1 = r_1$, and $\overline{r}_2 = r_1 r_2$, *etc.*) and $\overline{\pi}_k$ for $\pi_k \circ \cdots \circ \pi_1$, (so, $\overline{\pi}_1 = \pi_1$, and $\overline{\pi}_2 = \pi_2 \circ \pi_1$, *etc.*).

Registration

The registration uses a exponentiation mixnet to generate pseudonyms for the candidates and examiners, in two different runs. Without loss of generalization the pseudonyms for the candidates are assumed to be generated first.

In particular, let us assume n eligible candidates of identities $C_1, \ldots, C_n$. Let g denote a generator of a multiplicative subgroup $\mathbb{G}$ of order q. Each C_i has a public/pair keys (PK_i, SK_i), each $PK_i = g^{SK_i}$. The identities of the candidates as well as their public keys are public.

The first mix server mix_1 takes $\langle PK_i \rangle$ —the list of the public keys of the candidates— generates a fresh random $r_1 \in \{1, q-1\}$, and computes $\langle PK_i^{r_1} \rangle$ —the list of the public keys to the r_1. Then mix_1 signs and sends this list in secret shuffled order (*i.e.*, it sends $\langle PK^{r_1}_{\pi_1(i)} \rangle$, where π_1 is the permutation of indexes applied by mix_1), along with g^{r_1} to the next mix server. Further mix servers repeat these steps as required. Each mix server signs and publishes the shuffled list on the bulletin board, as shown in Figure 1. The last mixserver, mix_m, publishes also $g^{\overline{r}_m}$. Note that if the bulletin board has an access control mechanism, (*i.e.*, if only mix servers can publish data therein) the signatures are no longer required.

		mix_1	mix_2		mix_m	
C_1	PK_1	$PK^{\overline{r}_1}_{\overline{\pi}_1(1)}$	$PK^{\overline{r}_2}_{\overline{\pi}_2(1)}$	$\cdots$	$PK^{\overline{r}_m}_{\overline{\pi}_m(1)}$	$= \overline{PK}_1$
C_2	PK_1	$PK^{\overline{r}_1}_{\overline{\pi}_1(2)}$	$PK^{\overline{r}_2}_{\overline{\pi}_2(2)}$	$\cdots$	$PK^{\overline{r}_m}_{\overline{\pi}_m(2)}$	$= \overline{PK}_2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	
C_n	PK_n	$PK^{\overline{r}_1}_{\overline{\pi}_1(n)}$	$PK^{\overline{r}_2}_{\overline{\pi}_2(n)}$	$\cdots$	$PK^{\overline{r}}_{\overline{\pi}_m(n)}$	$= \overline{PK}_n$
	g	$g^{\overline{r}_1}$	$g^{\overline{r}_2}$	$\cdots$	$g^{\overline{r}_m}$	$= h_C$

Fig. 1. Using exponentiation mixnet to generate pseudonyms. All the terms within the box are published on the bulletin board.

While the intermediate steps should be posted to a bulletin board, there is no need for the intermediate $g^{\overline{r}_1}, \ldots, g^{\overline{r}_{m-1}}$ terms be posted. This is to avoid each candidate tracking their intermediate pseudonyms through the mixnet: although such eventuality is not an attack, it is an undesired feature. The last mix server mix_m publishes the final $h_C = g^{\overline{r}_m}$, and the list of pseudonyms $\langle \overline{PK}_i \rangle = \langle PK^{\overline{r}_m}_{\overline{\pi}_m(i)} \rangle$. A zero-knowledge proof could be required to prove that the mix servers behave correctly. Once the shuffled pseudonyms and the corresponding signatures have been posted along with h_C, each candidate, say C_k, can recognize her pseudonym among those in the shuffled list $\langle \overline{PK}_i \rangle$ by computing $h_C^{SK_k}$ and finding the match. The pseudonym from now on serves as the pseudo identity for C_k.

After the pseudonyms of candidates have been published, the mixnet generates the pseudonyms for examiners in a similar way. Since a different random

value is used by the mix servers to generate the examiner pseudonyms, a different h_E is published at the end of the mix.

Testing

Before starting the testing phase, the manager generates the test questions, signs them with its private key SK_M, and encrypts each test question under a candidate pseudonym. A test question is a list of questions. Depending on the examination, different tests can be made of distinct or permuted questions.

As soon as the testing starts, the manager authenticates the candidate via remote invigilation software. In particular, the manager checks whether the candidate details printed on the top of the smart card matches the candidate identity. When all candidates have been authenticated, the manager publishes the encrypted test questions in the bulletin board. Once all the candidates have received their test questions, they are allowed to work on their test answers.

When the candidate concludes the test answer, she can submit the test as follows: the candidate appends her pseudonym and the test answer to the test question, so the filled test is $T_C = \langle ques, ans, \overline{PK} \rangle$. Then, she signs T_C with her private key SK_C using the generator h_C instead of g. Thus, the signature can be verified using the candidate's pseudonym $\overline{PK}_C$. The candidate then encrypts the signed test with the public key of the manager PK_M, and submits it to the manager. The manager collects and decrypts the test, which is signed using the manager private key SK_M. The manager then encrypts the signed test under the corresponding candidate's pseudonym, and publishes the encryption as receipt.

Marking

The manager encrypts the signed test under an eligible examiner pseudonym $\overline{PK}_E$, which is on the bulletin board. The corresponding examiner assigns a mark to the test, which is appended to the signed test, thus generating the evaluation $M_C = \langle Sig\{T_C\}_{SK_M}, mark \rangle$. The examiner then signs M_C with his private key SK_E and the generator h_E. The examiner finally encrypts M_C under PK_M and submits his evaluation to the manager.

Notification

The manager receives the encrypted evaluation from the examiner, which are decrypted and re-encrypted under the corresponding candidate pseudonym $\overline{PK}_C$. The manager publishes all the test evaluations together. Then, the manager asks the mixnet to reveal the random values r used to generate the candidates pseudonyms. In so doing, the candidate anonymity is revoked, and the mark can finally be registered.

Notification (alternative) Some universities allow candidates decide whether to get the mark or to withdraw their test entirely without any mark being notified and registered. This particular way to run a final exam is adopted, for instance, by those universities where candidates are conceded with a limited amount of failures during the exam season, mainly to discourage them from taking the exam without adequate preparation. Other universities, again to discourage candidates to sit at the exam just 'to try it out', have a rule saying that if a candidate chooses to know her mark and this turns out to be a fail, then she has to skip the next exam session. By giving a candidate the possibility to withdraw a test without knowing the mark, those universities soften the severity of such rules, by letting a candidate spare wasting one of her attempt token when she realizes, by her own, to have performed insufficiently.

Remark! can include such requirement via an alternative notification phase. In this case, the manager publishes a public commit of the mark, instead of the mark. Then, if a candidate wants to know her mark, she proves the knowledge of her private key. If so, the manager reveals the commitment parameter, and the candidate can check the commitment. Notably, the notification does not involve the mixnet.

4 Security Analysis

We discuss informally the security of *Remark!* and give arguments supporting the claim that it achieves our security requirements. We organize our argumentation in four sections. The first section discusses authenticity properties, the second anonymity properties, the third privacy properties, and the last verifiability properties.

Authentication Test Answer Authentication (p1) is achieved because the manager only accepts the test whose signature can be only verified with a pseudonym published by the mixnet. No one but the candidate who holds the corresponding private key can generate a correct signature. Colluding candidates who switch their smart cards are detected by invigilation.

Examiner Authentication (p2) holds because the manager encrypts the test with the examiner's pseudonym. Only the examiner who holds the corresponding private key obtains the test, and the manager accepts the evaluation only if it can check the signature using the corresponding examiner's pseudonym.

Anonymity The pseudonym guarantees the anonymity of the test submitted by the candidate, who signs the test with her private key and the generator h_C. The mix servers cannot associate a pseudonym to a candidate's identity, unless all of them collude. Even if a malicious examiner colludes with the manager, Anonymous Marking (p3) holds until all the mix servers reveal their secret exponents.

Remark! ensures Anonymous Examiner (p4) because the manager encrypts the test with the examiner's pseudonym. The examiner can fairly evaluate the

anonymous test answers without fear of being coerced by any candidate as examiners' pseudonyms are not revoked by the mixnet. Moreover, if the examination board consists of different examiners, a candidate has no guarantee that a colluding examiner will evaluate her test answers.

Privacy Question Secrecy (p5) is achieved because the manager publishes the test question once the candidate is under invigilation.

The manager cannot learn which test question is assigned to a specific candidate because the test question are encrypted with the anonymous candidate's pseudonym. Thus, *Remark!* ensures Question Privacy (p6).

The protocol also ensures Mark Privacy (p7) because the mark is encrypted with the candidate's pseudonym and then published on the bulletin board. Thus, each candidate only learns her corresponding mark. Notably, only the manager learns the mark after the mixnet reveal the secret exponents.

Verifiability Each mix server publishes its generated list of pseudonyms (the intermediated and the last), which are signed and with a zero-knowledge proof of correctness (*e.g.*, that all pseudonyms are generated by using the same exponential value). This allow any observer to verify the authenticity and the correctness of the pseudonyms. Once the final pseudonyms are published on the bulletin board, each eligible candidate and examiner can only find their corresponding pseudonym.

Test Verifiability (p8) is guaranteed because the manager publishes the receipt after it receives a valid signature (i.e. the manager can verify a signature using a pseudonym as verification key). Thus the candidate can verify that her test is considered for evaluation. Moreover, she can also prove that her test has been accepted because the manager signs the receipt.

Finally, *Remark!* ensures Mark Verifiability (p9). In fact, the marks are published before the mixnet reveals their secret exponents. Thus, the candidate can verify that the manager registers the correct mark once the mixnet revokes her anonymity. Note that both the manager and the examiner sign the test to which the mark is assigned. Since the mark is signed by the examiner, if the manager registers an incorrect mark, the candidate can prove to an authority the correct mark the examiner assigned to her test.

5 Conclusion

This paper proposes *Remark!*, an e-exam protocol that achieves heterogeneous security properties (authentication, privacy, anonymity, and verifiability) in a realistic threat model with few security assumptions. Notably, it requires no trusted parties but that only one mix server behave honestly. *Remark!* can resist against collusion of candidates, examiner and manager, or examiner and candidate. Although the paper presents an informal analysis of the protocol, a preliminary formal analysis of *Remark!* in the symbolic model confirms that it

ensures all the nine fundamental security requirements. Future work is to identify more interesting security properties for remote exam, and verify whether *Remark!* can ensure them. We also plan to build a prototype.

References

1. ProctorU. `http://www.proctoru.com/`.
2. Huszti A. and Pethõ A. A Secure Electronic Exam System. *Publicationes Mathematicae Debrecen*, 77:299–312, 2010.
3. Bella G., Giustolisi R., and Lenzini G. What Security for Electronic Exams? In *Proc. of 8th Int. Conf. on Risk and Security of Internet and Systems (CRiSIS), 2013*, 2013.
4. Benaloh J., Ryan P.Y.A., and Teague V. Verifiable postal voting. In *Security Protocols XXI*, volume 8263 of *Lecture Notes in Computer Science*, pages 54–65. Springer Berlin Heidelberg, 2013.
5. Castellà-Roca J., Herrera-Joancomartí J., and Dorca-Josa A. A Secure E-Exam Management System. In *Proc. of the Int. Conf. on Availability, Reliability and Security (ARES), 2006*, pages 864–871, 2006.
6. Haenni R. and Spycher O. Secure internet voting on limited devices with anonymized dsa public keys. In *Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, EVT/WOTE'11. USENIX Association, 2011.

A Appendix: Message Sequence Chart

Notation. A test question is denoted by *quest*, and a test answer by *ans*. SK_X and PK_X denotes the ElGamal private and public keys of the principal X. We assume a common public generator g for the keys of all principals. $\overline{PK}_X$ denotes the pseudonym of the principal X, and r_{X_i} is the secret value used by the mix server i when processing the batch of the role X. The terms *Enc* and *Sig* denote respectively the encryption and signature functions of a message. In particular, the notation $Sig\{msg\}_{\overline{SK}_X, h_X}$ denotes the message *msg* and its signature using the private key SK_X and the parameter h_X rather than g.