BlindReview: Anonymous and End-to-End Verifiable Peer Review

***Abstract*—We introduce BlindReview, an anonymous and end-to-end verifiable peer review system that cryptographically guarantees both privacy and auditability throughout the reviewing process. We formally define these security properties and provide rigorous proofs that BlindReview satisfies them. We also present an implementation demonstrating our protocol's practicality. This work serves as a foundation for verifiable and privacy-preserving peer review, offering a concrete solution to enhance transparency and reduce bias in the academic peer review process.**

***Index Terms*—Peer review, privacy, anonymity, verifiability**

I. Introduction

One of the central goals of security research is to rigorously define and analyse protocols to ensure they meet precise security requirements. From secure messaging to electronic voting systems, the academic literature and the real world abound in examples of protocols designed to achieve properties ranging from data confidentiality, integrity and user anonymity to end-to-end verifiability (see, for example, [1]–[4]). These formalisations not only guide protocol design but also serve as a benchmark for evaluating the robustness and fairness of systems that impact users at scale.

It is therefore surprising that the very systems underpinning academic advancement, namely peer review platforms, have largely escaped this kind of formal scrutiny. Widely used systems such as EasyChair [5] and HotCRP [6], for instance, are not end-to-end verifiable, and place full trust in the conference chairs, who enjoy a complete view of the submissions, conflict of interests, paper bids and assignments, and reviews. In all of the phases of the reviewing process, chairs make decisions that are crucial to the acceptance or rejection of a submission, but these are not currently auditable.

In this work, we take a first step towards addressing this gap by formally modelling the peer review process as a protocol. The act of formalisation itself is a significant contribution: it brings clarity to implicit trust assumptions and highlights opportunities for security and fairness enhancements. Additionally, we incorporate into the model stronger properties, not yet achieved by current systems, such as reviewer anonymity with respect to the chairs. Our motivation for this stems from the fact that enabling and maintaining reviewer anonymity throughout the reviewing process can help reduce potential bias in discussions and decision-making, thereby enhancing the overall fairness of the process.

We provide a novel, practical peer review protocol, BlindReview, satisfying our formalisation and provably achieving privacy, including reviewer and author anonymity, and end-to-end verifiability. More specifically, we make the following contributions.

- We provide a concrete construction, BlindReview, that is based on well-established cryptographic primitives, namely, ElGamal encryption, Pedersen commitments and Schnorr signatures, and utilises Schnorr zero-knowledge proof systems and signatures of knowledge to provide privacy and verifiability of each phase of the protocol.
- We introduce formal security models for our construction, defining privacy, anonymity and a series of verifiability properties that, when combined, capture end-to-end verifiability. We also show that BlindReview satisfies all these properties.
- We implement BlindReview using Rust, and show its practical performance on a variety of parameters (i.e., accounting for different size conferences and number of paper assignments).

Our construction demonstrates that formal rigor and stronger privacy are compatible with real-world usability and can serve as a foundation for more trustworthy scholarly infrastructure.

Paper Structure. The remainder of the paper is structured as follows. After discussing related work, we establish our system model and state our security goals in Section II. In Section III we give details of our construction, including its formalisation. We provide a security analysis in Section IV, discussing both privacy and verifiability in terms of models and proofs. We conclude in Section V with details on how to instantiate BlindReview and a performance analysis derived by our implementation results.

A. Related work

Several reviewing systems exist (e.g., EasyChair, HotCRP) and are currently used to establish what papers deserve publication through a peer-review process. Research related to such systems has either proposed to extend them to enhance security, or developed tools to analyse the system and detect potential attacks.

Proposal of new systems. Motivated by cloud computing applications, Arapinis, Bursuc and Ryan [7] study the security and privacy of conference management tools such as EasyChair, and propose ConfiChair, a solution providing, in particular, privacy guarantees with respect to the cloud (e.g., confidentiality of the paper and the review with respect to the system administrator). The authors achieve this essentially by symmetrically encrypting everything that is uploaded to the cloud. However, the issue of trust in the conference chair is not addressed. Indeed, it is mentioned and assumed that if an author or reviewer is not willing to trust the chair, they would not participate in the conference. We dispute this and consider it important that the actions of the chair are also auditable. In [8], Lo, Phan and Goi discuss some security issues present in the WSaR submission and review software [9], such as vulnerability to browser caching, and password-related weaknesses. Specifically, the authors propose an email-based password distribution protocol to be used in conjunction with WSaR in order to mitigate these issues. The paper focuses on the web security of the protocol, and does not consider the issue of verifiability. Our work is distinct from existing reviewing schemes in the literature as we target privacy and end-to-end verifiability, requiring in particular that the chair's actions are verifiable.

Tools for analysis. Cho and Fu [10] address the challenge of query optimisation whilst ensuring confidentiality in the HotCRP reviewing system. The authors model the system's information flow to prove certain search query optimisations do not leak information. These results are not concerned with providing anonymity and verifiability of the reviewing process, and therefore are distinct from the work in this paper. In [11], [12] the authors propose a language model to analyse the access control policies of web-based collaborative systems, and use this to identify some attacks on EasyChair. We note that these attacks (e.g., a reviewer can submit two reviews on the same paper) do not hold for BlindReview since the verifiability properties that we define, and that BlindReview satisfies, would detect them.

In contrast to existing literature, we propose cryptographically-enabled privacy and verifiability by design. We formally model these properties and provide rigorous proofs that they are satisfied, thereby laying the foundations for secure peer review.

II. System Model and Security Goals

We define a peer review system with respect to three entities. The *program chair*, denoted C, is responsible for running the system and ultimately makes decisions regarding which papers are accepted. The *program committee*, denoted PC, is the set of entities responsible for reviewing papers. Each PC member, denoted R, bids on submissions and is assigned papers to review. For clarity, we say that the PC is the set of entities that can be assigned papers to review. Then, when a PC member is assigned a paper, we say that they are a *reviewer* of that paper. *Authors*, denoted A, submit papers for review.

With these entities, we model a peer review system that consists of eight phases, noting that existing reviewing systems such as EasyChair and HotCRP naturally fit into this model. We illustrate our peer review system model in Figure 1. Briefly, the phases proceed as follows.

0) *Setup and Key Generation*. System public parameters are generated and all entities generate a public/secret key pair.
1) *Paper Submission*. Authors construct paper submissions, which include a conflicts of interest list, and sends the submission to the chair.
2) *Paper Distribution*. Once the submission phase is complete, the chair distributes to each PC member all papers that the PC member does not have a conflict of interest with.
3) *Paper Bidding*. PC members express a preference for each distributed submission.
4) *Paper Assignment*. For each submission, the chair collects all the bids received from the program committee and assigns the submission to PC members. The chair ensures that each PC member is assigned the same number of papers to review, and that submissions are assigned to PC members with the highest bids.
5) *Paper Review*. Each reviewer generates and submits to the chair a score and review text for each assignment. We assume that the assigned reviewers may discuss the paper and may change their review before submission to the chair. Accordingly, the review submitted in this phase is the final review following any discussion.
6) *Decision Communication*. The chair collects reviews for each submission and selects a decision outcome for each paper. The decision and reviews are sent to the submission author.
7) *Camera Ready*. If the submission is accepted, the author submits a camera ready version of their submission to the chair.

We state the security goals of a peer review system next, leaving their formal definition for Section IV.

a) Anonymity and Privacy: We start by considering *anonymity*, which focuses on protecting the anonymity of authors and PC members (even from the chair) throughout the reviewing process. We then define *privacy*, which captures anonymity as well as submission content confidentiality and conflict privacy. The latter properties capture, respectively, the intuition that the content of a submission is revealed only to the chair and PC members that are not in conflict with the submission,

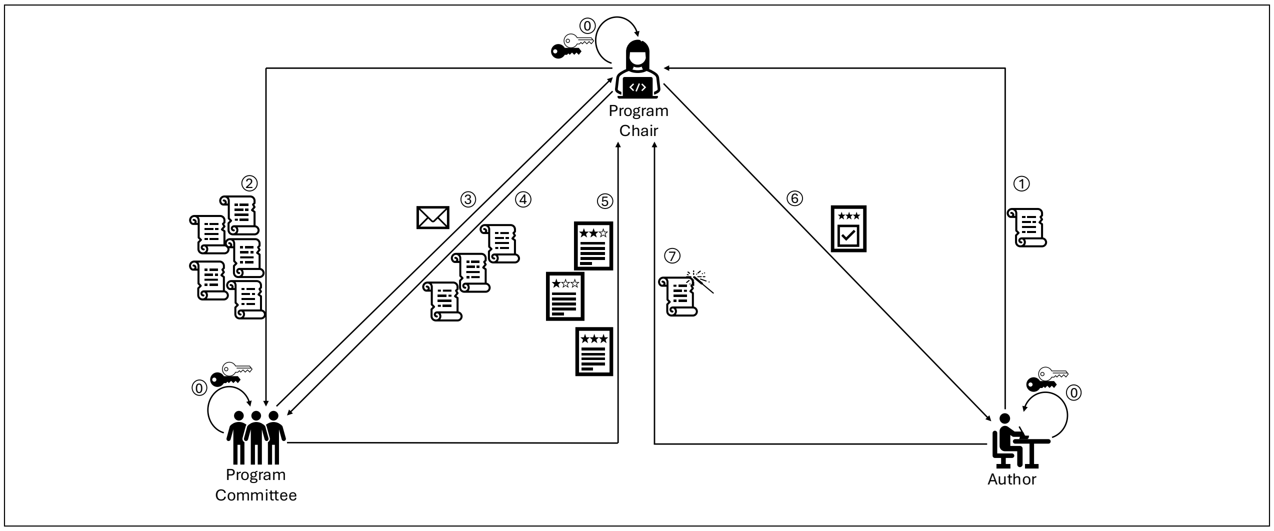

Figure 1: A peer review system consisting of phases 0 – 7 and run between a program chair, program committee and a set of authors.

and the requirement that the list of conflicts of a paper remains private throughout the reviewing process. We formalise anonymity and privacy in Definition 2, and demonstrate that our protocol satisfies privacy in the presence of honest-but-curious attackers, and anonymity in the presence of dishonest attackers.

b) End-to-End Verifiability: We design our system so that each protocol phase is verifiable. Accordingly, in every phase, there is a verification algorithm to check that the computation was performed correctly. Formally, we define five verifiability properties, corresponding to the submission, bidding, assignment, review and decision, and camera ready phase, respectively. We say that a reviewing protocol is *end-to-end verifiable* (Definition 4) if it satisfies all five verifiability properties, and show that BlindReview does.

III. Our peer review system BlindReview

In this section, we present the full technical details of our protocol BlindReview. We assume there are N submissions and M PC members. We require that each submission is reviewed by 3 distinct PC members.

BlindReview requires a group generation algorithm $\mathcal{G}$ that outputs a description of a group $\mathbb{G}$, its order q and two generators $g, \bar{g} \in \mathbb{G}$. We use the ElGamal encryption scheme $(\mathsf{Enc}_g, \mathsf{Dec}_g)$ [13], the Schnorr signature scheme $(\mathsf{Sig}_g, \mathsf{Ver}_g)$ [14], and the Pedersen commitment scheme $(\mathsf{Com}_{(g,\bar{g})}, \mathsf{Ope}_{(g,\bar{g})})$ [15], all of which operate on the group $\langle g \rangle$. Our construction also requires a hash function $\mathsf{H}\colon \{0,1\}^* \to \mathbb{G}$. BlindReview makes use of several zero-knowledge proof systems, based on standard Σ protocols that are made non-interactive by the Fiat-Shamir transform [16]. We present full details of the zero-knowledge proof systems in Appendix B. Finally, we define an algorithm $\mathsf{Shuffle}$ that takes as input a set of M elements and outputs a permutation of the input set. With these primitives, we now describe each phase of BlindReview.

a) Setup and Key Generation: We define setup and key generation algorithms in Figure 2. A peer review protocol is defined with respect to a security parameter λ and a number of reviews to be generated by each PC member ℓ. Taking these parameters as input, algorithm Setup runs group generation algorithm $\mathcal{G}$ and returns the set of public parameters $\mathtt{init} = (\ell, \mathbb{G}, q, g, \bar{g})$. For ease of reading, we assume that $\mathtt{init}$ is an implicit input to all algorithms.

Following setup, each entity generates a public/secret key pair by running their respective key generation algorithm. In particular, the program chair runs algorithm $\mathsf{C.KGen}$ to generate a key pair $(\mathsf{pkc}, \mathsf{skc})$ for the El Gamal encryption scheme, and each PC member runs $\mathsf{R.KGen}$ to generate key pair $(\mathsf{pkr}, \mathsf{skr})$ for the Schnorr signature scheme. Additionally, algorithm $\mathsf{R.KGen}$ generates a standard proof of knowledge of correct key construction. Accordingly, we also provide a reviewer key verification algorithm $\mathsf{VerKGen}$ that outputs 1 if the zero-knowledge proof verifies. The author, running algorithm $\mathsf{A.KGen}$, generates a key pair $(\mathsf{ska}_1, \mathsf{pka}_1)$ that is used to generate and verify a PC conflict list, and a Schnorr signature scheme key pair $(\mathsf{ska}_2, \mathsf{pka}_2)$. Two further elements, ska_3 and ska_4, are chosen uniformly at random from the set $\mathbb{Z}_q^*$, and are used as randomness to generate Pedersen commitments in the paper submission phase. The author's secret key is $\mathsf{ska} = (\mathsf{ska}_1, \mathsf{ska}_2, \mathsf{ska}_3, \mathsf{ska}_4)$, and the public key is $\mathsf{pka} = (\mathsf{pka}_1, \mathsf{pka}_2)$. Note that the author's key pair is tied to a unique submission, meaning authors generate a key pair for each new submission.

```
$\mathsf{Setup}(\lambda, \ell)$
    $(\mathbb{G}, q, g, \bar{g}) \xleftarrow{\$} \mathcal{G}(1^\lambda)$
    $\mathtt{init} \leftarrow (\ell, \mathbb{G}, q, g, \bar{g})$
    $\textbf{return } \mathtt{init}$

$\mathsf{C.KGen}()$
    $\mathsf{skc} \xleftarrow{\$} \mathbb{Z}_q^*$
    $\mathsf{pkc} \leftarrow g^{\mathsf{skc}}$
    $\textbf{return } (\mathsf{pkc}, \mathsf{skc})$

$\mathsf{R.KGen}()$
    $\mathsf{skr} \xleftarrow{\$} \mathbb{Z}_q^*$
    $\mathsf{pkr} \leftarrow g^{\mathsf{skr}}$
    $\rho \leftarrow \mathsf{ZK.Prove}(\mathsf{pkr}, \mathsf{skr})$
    $\textbf{return } (\mathsf{pkr}, \mathsf{skr}, \rho)$

$\mathsf{R.VerKGen}(\mathsf{pkr}, \rho)$
    $\textbf{return } \mathsf{ZK.Verify}(\mathsf{pkr}, \rho)$

$\mathsf{A.KGen}()$
    $(\mathsf{ska}_1, \mathsf{ska}_2, \mathsf{ska}_3, \mathsf{ska}_4) \xleftarrow{\$} (\mathbb{Z}_q^*)^4$
    $\mathsf{ska} \leftarrow (\mathsf{ska}_1, \mathsf{ska}_2, \mathsf{ska}_3, \mathsf{ska}_4)$
    $\textbf{for } i \in [2] : \mathsf{pka}_i \leftarrow g^{\mathsf{ska}_i}$
    $\tau \leftarrow \mathsf{ZK.Prove}(\mathsf{pka}_1, \mathsf{ska}_1)$
    $\mathsf{pka} \leftarrow (\mathsf{pka}_1, \mathsf{pka}_2)$
    $\textbf{return } (\mathsf{pka}, \mathsf{ska}, \tau)$

$\mathsf{A.VerKGen}(\mathsf{pka}, \tau)$
    $\textbf{parse } \mathsf{pka} \textbf{ as } (\mathsf{pka}_1, \mathsf{pka}_2)$
    $\textbf{return } \mathsf{ZK.Verify}(\mathsf{pka}_1, \tau)$
```

Figure 2: Setup and key generation algorithms for the BlindReview protocol.

```
$\mathsf{Submit}(\mathsf{pkc}, \mathsf{ska}, \{\mathsf{pkr}_i\}_{i \in [M]}, \mathtt{alist}, \mathtt{content}, \mathtt{RC})$
    $\textbf{parse } \mathsf{ska} \textbf{ as } (\mathsf{ska}_1, \mathsf{ska}_2, \mathsf{ska}_3, \mathsf{ska}_4)$
    $p_1 \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_3, \mathtt{alist})$
    $p_2 \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_4, \mathtt{content})$
    $\textbf{for } j \in [M]$
        $\textbf{if } j \in \mathtt{RC} : x_j \leftarrow \mathsf{pkr}_j^{\mathsf{ska}_1}$
        $\textbf{else } x_j \xleftarrow{\$} \mathbb{G}$
    $p_3 \leftarrow \mathsf{Shuffle}(\{x_j\}_{1 \le j \le M})$
    $r \xleftarrow{\$} \mathbb{Z}_q^*$
    $p_4 \leftarrow g^r$
    $p_5 \leftarrow H(\mathsf{pkc}^r) \oplus [\mathsf{ska}_1 || \mathsf{ska}_4 || \mathtt{content}]$
    $p_6 \leftarrow \mathsf{ZK}\{r : g^r = p_4\}$
    $p_7 \leftarrow \mathsf{Sig}_g(\mathsf{ska}_2, [p_1 \| p_2 \| p_3 \| p_4 \| p_5 \| p_6])$
    $\textbf{return } \mathbf{p} \leftarrow (p_1, p_2, p_3, p_4, p_5, p_6, p_7)$

$\mathsf{VerSubmit}(\mathsf{pka}, \mathbf{p})$
    $\textbf{parse } \mathbf{p} \textbf{ as } (p_1, p_2, p_3, p_4, p_5, p_6, p_7)$
    $\textbf{parse } \mathsf{pka} \textbf{ as } (\mathsf{pka}_1, \mathsf{pka}_2)$
    $\textbf{return } \mathsf{Ver}_g(\mathsf{pka}_2, [p_1 \| p_2 \| p_3 \| p_4], p_7)$
    $\wedge \mathsf{ZK.Verify}((g, p_4), p_6)$
```

Figure 3: Paper submission phase algorithms for the BlindReview protocol.

PC member and chair keys, on the other hand, are long-term, in the sense that they can be used for multiple conferences.

b) Paper Submission: To submit a paper for review, an author must generate a paper submission $\mathbf{p} = (p_1, p_2, p_3, p_4, p_5, p_6, p_7)$ that is computed using algorithm Submit (Figure 3). The author must first prepare the paper content `content`, author list `alist` and a set of indices `RC` that correspond to PC members that are in conflict with the submission. These values are input to algorithm Submit, which also takes as input the chair's public key pkc and the author's secret key ska, and computes a paper submission as follows.

First, a commitment p_1 to the author list `alist` is generated using ska_3 from the author's secret key. Looking forward (this means that authors cannot be added to, or removed from, a paper following submission). If the paper is accepted and a camera ready version submitted, the author can provide the chair with ska_3 to open the commitment and verify the author list. A commitment p_2 to the content of the paper using ska_4 is also generated. Commitment p_2 is crucial to ensure that the chair does not modify the content of the submission before the paper distribution phase.

Algorithm Submit then constructs a conflict vector p_3. That is, for all M PC members, the author creates an element x_j such that, if the PC member pkr_j is a conflict of interest (i.e., $j \in \mathtt{RC}$), $x_j \leftarrow \mathsf{pkr}_j^{\mathsf{ska}_1}$. Else, x_j is chosen uniformly at random from the group $\mathbb{G}$. The x_j values are shuffled to support privacy of the conflicts, and p_3 is set to be the output of the shuffle.

Element p_5 is computed as a masking of the tuple $(\mathsf{ska}_1, \mathsf{ska}_4, \mathtt{content})$ for the chair, and p_6 proves knowledge of the masking randomness. By unmasking element ska_1, the chair can, during the paper distribution phase, determine the PC members in conflict with the submission and ensure that the paper is not distributed to conflicting PC members. Moreover, unmasking ska_4 enables the chair to check that the submission content in p_1 matches the submission content in p_4, and allows the chair to demonstrate to PC members that the distributed paper content matches the submitted content.

Finally, algorithm Submit produces signature p_7 on the tuple $(p_1, p_2, p_3, p_4, p_5, p_6)$ using ska_2 and returns the submission $\mathbf{p} = (p_1, p_2, p_3, p_4, p_5, p_6, p_7)$.

We also define a public verification algorithm VerSubmit that can be run by any entity to ensure that a submission is not modified. Practically, we require that the chair runs VerSubmit before paper distribution, and PC members run it before the paper bidding phase.

c) Paper Distribution: Following the submission phase, the chair can desk reject papers; all other papers proceed to the distribution phase. We define algorithms for the distribution phase in Figure 4. BlindReview includes an algorithm Distribute that generates the distribution vector for a single reviewer. To distribute all N submissions to all M PC members, the chair must run algorithm Distribute M times, i.e., once for each PC member. The algorithm takes as input the chair's secret key skc, all submissions $\{\mathbf{p}_i\}_{i\in[N]}$, the author public key corresponding to each submission $\{\mathsf{pka}_i\}_{i\in N}$ and a PC member public key pkr. For each submission $\mathbf{p}_i$, algorithm Distribute first checks that the paper content included in ciphertext $p_{i,5}$ and commitment $p_{i,2}$ are identical. To achieve this, $p_{i,5}$ is unmasked using the chair's secret key and $p_{i,4}$ to obtain the paper content $\texttt{content}_i$ and commitment randomness $\mathsf{ska}_{i,4}$. These values can then be used to check that commitment $p_{i,2}$ opens correctly. If the commitment opens to paper content $\texttt{content}_i$, algorithm Distribute proceeds to distribute the paper content. Indeed, a distribution tuple v_i is generated for each paper submission $\mathbf{p}_i$ as follows. If the PC member is not in the conflict vector (i.e., $\mathsf{pkr}^{\mathsf{ska}_{i,1}} \notin p_{i,3}$), v_i is set to be $(i, \texttt{content}_i, \mathsf{ska}_{i,4})$. That is, the PC member is given the content of the paper and the secret key part that can be used to open the commitment to the paper content, thus enabling the PC member to check that the chair has not modified the paper content. Else, v_i is $(i, \bot, \bot)$. Algorithm Distribute returns a distribution vector $(v_1, \ldots, v_N)$. In this way, the PC member is provided with all submissions that have not declared a conflict of interest with them.

```
Distribute(skc, $\{(\mathbf{p}_i, \mathsf{pka}_i)\}_{i\in[N]}$, pkr)
1 : for $i \in [N]$
2 :   parse $\mathbf{p}_i$ as $(p_{i,1}, p_{i,2}, p_{i,3}, p_{i,4}, p_{i,5}, p_{i,6}, p_{i,7})$
3 :   $(\mathsf{ska}_{i,1} \| \mathsf{ska}_{i,4} \| \texttt{content}_i) \leftarrow H(p_{i,4}^{\mathsf{skc}}) \oplus p_{i,5}$
4 :   if $\mathsf{Ope}_{(g,\bar{g})}(\mathsf{ska}_{i,4}, \texttt{content}_i, p_{i,2}) = 0$   return $\bot$
5 :   if $\mathsf{pkr}^{\mathsf{ska}_{i,1}} \notin p_{i,3}$ : $v_i \leftarrow (i, \texttt{content}_i, \mathsf{ska}_{i,4})$
6 :   else $v_i \leftarrow (i, \bot, \bot)$
7 : return $(v_1, \ldots, v_N)$

VerDistribute(skr, $\{(v_i, \mathbf{p}_i, \mathsf{pka}_i)\}_{i\in[N]}$)
1 : for $i \in [N]$
2 :   parse $\mathbf{p}_i$ as $(p_{i,1}, p_{i,2}, p_{i,3}, p_{i,4}, p_{i,5}, p_{i,6}, p_{i,7})$
3 :   if $v_i = (i, \bot, \bot) \wedge \mathsf{pka}_{i,1}^{\mathsf{skr}} \notin p_{i,3}$   return 0
4 :   if $v_i = (i, \texttt{content}_i, \mathsf{ska}_{i,4})$
5 :     if $\mathsf{pka}_{i,1}^{\mathsf{skr}} \in p_{i,3}$   return 0
6 :     if $\mathsf{Ope}_{(g,\bar{g})}(\mathsf{ska}_{i,4}, \texttt{content}_i, p_{i,2}) = 0$
7 :       return 0
8 : return 1
```

Figure 4: Paper distribution phase algorithms for the BlindReview protocol.

Reviewers can verify that their paper distribution vector is correct by running algorithm VerDistribute. For each distribution tuple v_i, if the tuple is empty, i.e., $v_i = (i, \bot, \bot)$, algorithm VerDistribute computes the value $\mathsf{pka}_{i,1}^{\mathsf{skr}}$ and checks if this value appears in $p_{i,3}$. Recalling that $\mathsf{pka}_{i,1}^{\mathsf{skr}} \in p_{i,3}$ means that the PC member is not in conflict, if $\mathsf{pka}_{i,1}^{\mathsf{skr}} \in p_{i,3}$, algorithm VerDistribute would expect v_i to include the paper content. Thus, if $\mathsf{pka}_{i,1}^{\mathsf{skr}} \notin p_{i,3}$, verification of the distribution vector fails and the algorithm aborts. Else, if $v_i = (i, \texttt{content}_i, \mathsf{ska}_{i,4})$, algorithm VerDistribute checks that commitment $p_{i,2}$ opens to $\texttt{content}$, and that $\mathsf{pka}_{i,1}^{\mathsf{skr}} \notin p_{i,3}$. As expected, verification fails if the commitment does not open, or if the PC member is in the conflict list. If all tests pass, VerDistribute returns 1. If any test fails, the algorithm returns 0.

d) Paper Bidding: Once the chair has completed distribution of submissions, the BlindReview protocol can enter the paper bidding phase (Figure 5). In this phase, each PC member runs algorithm Bid N times, once for each paper submission. First, the PC member pkr selects a bid score m that indicates the PC member's willingness to review the submission (note that $m = 0$ indicates the PC member is declaring a conflict with the paper, and submission bids default to $m = 0$ if the PC member is in conflict.). Then, algorithm Bid proceeds to generate a bid vector b in the following way. The algorithm first computes a hash of the submission raised to the power of skr ($\gamma \leftarrow \mathsf{H}(\mathbf{p})^{\mathsf{skr}}$), and an ephemeral public key $\mathsf{pk} \leftarrow h^{\mathsf{skr}}$ for some h chosen uniformly at random from the group $\mathbb{G}$. Then, algorithm Bid generates a zero-knowledge proof of knowledge pk has been generated from the secret key of one of the PC members, by showing that there exists a pkr_i such that $\log_g(\mathsf{pkr}_i) = \log_h(\mathsf{pk})$:

$$\mathsf{ZK}\{\mathsf{skr} : \bigvee_{i\in[M]} g^{\mathsf{skr}} = \mathsf{pkr}_i \wedge h^{\mathsf{skr}} = \mathsf{pk}\}.$$

Then, algorithm Bid generates a zero-knowledge proof of knowledge that the PC member is not in conflict with the submission. That is, for each $x_j \in p_3$, algorithm Bid generates a proof that $x_j \neq \mathsf{pka}_1^{\mathsf{skr}}$ that is consistent with the following relation:

$$\mathsf{ZK}\{\mathsf{skr} : \hat{x}_j \neq \mathsf{pka}_1^{\mathsf{skr}} \wedge \mathsf{pk} = h^{\mathsf{skr}} \wedge \gamma = H(\mathbf{p})^{\mathsf{skr}}\}.$$

If the PC member is in conflict, the proof is set as empty. All zero knowledge proofs are collected in a set $\hat{\pi}$. Finally, algorithm Bid generates a signature σ over the tuple $(\mathbf{p} \| \mathsf{pka} \| h \| \mathsf{pk} \| \hat{\pi} \| m)$ and outputs the bid $b = (h, \gamma, \mathsf{pk}, \hat{\pi}, m, \sigma)$.

Anyone can verify a bid b for a submission $\mathbf{p}$ by author pka by running algorithm VerBid, which checks π and the set of zero-knowledge proofs $\hat{\pi}$ included in the bid.

```
Bid(pkc, skr, $\{\mathsf{pkr}_i\}_{i\in[M]}$, pka, $\mathbf{p}$, $v$, $m$)
1 :  parse $\mathbf{p}$ as $(p_1, p_2, p_3, p_4, p_7)$
2 :  parse $p_3$ as $\{\hat{x}_1, \ldots, \hat{x}_M\}$
3 :  $\gamma \leftarrow \mathsf{H}(\mathbf{p})^{\mathsf{skr}}$
4 :  $h \xleftarrow{\$} \mathbb{G}$
5 :  $\mathsf{pk} \leftarrow h^{\mathsf{skr}}$
6 :  $\pi \leftarrow \mathsf{ZK.Prove}((g, \{\mathsf{pkr}_i\}_{i\in[M]}, h, \mathsf{pk}), \mathsf{skr})$
7 :  if $m = 0$ : $\hat{\pi} \leftarrow \perp$
8 :  else
9 :    for $j \in [M]$
10 :      $\hat{\pi}_j \leftarrow \mathsf{ZK.Prove}((\mathbf{p}, p_3[j], \gamma, \mathsf{pk}, h, \mathsf{pka}), \mathsf{skr})$
11 :    $\hat{\pi} \leftarrow (\hat{\pi}_j)_{j\in[M]}$
12 :  $\sigma \leftarrow \mathsf{Sig}_h(\mathsf{skr}, [\mathbf{p}\|\mathsf{pka}\|h\|\gamma\|\mathsf{pk}\|\hat{\pi}\|m])$
13 :  $b \leftarrow (h, \gamma, \mathsf{pk}, \pi, \hat{\pi}, m, \sigma)$
14 :  return $b$
```

```
VerBid(pka, $\{\mathsf{pkr}_i\}_{i\in[M]}$, $\mathbf{p}$, $b$)
1 :  parse $b$ as $(h, \gamma, \mathsf{pk}, \pi\hat{\pi}, m, \sigma)$
2 :  if $\hat{\pi} = \perp$ return 1
3 :  if $\mathsf{ZK.Verify}((g, \{\mathsf{pkr}_i\}_{i\in[M]}, h, \mathsf{pk}), \pi) = 0$
4 :    return 0
5 :  parse $\hat{\pi}$ as $(\hat{\pi}_1, \ldots, \hat{\pi}_M)$
6 :  for $j \in [M]$
7 :    if $\mathsf{ZK.Verify}((\mathbf{p}, p_3[j], \gamma, \mathsf{pk}, h, \mathsf{pka}), \hat{\pi}_j) = 0$
8 :      return 0
9 :  return $\mathsf{Ver}_h(\mathsf{pk}, [\mathbf{p}\|\mathsf{pka}\|h\|\gamma\|\mathsf{pk}\|\hat{\pi}\|m], \sigma)$
```

```
VerBidPool($\{\mathsf{pka}_i, \mathbf{p}_i, \{b_{i,j}\}_{j\in[M]}\}_{i\in[N]}, \{\mathsf{pkr}_i\}_{i\in[M]}$)
1 :  for $i \in [N]$
2 :    for $j \in [M]$
3 :      parse $b_{i,j}$ as $(h_{i,j}, \gamma_{i,j}, \mathsf{pk}_{i,j}, \pi_{i,j}, \hat{\pi}_{i,j}, m_{i,j}, \sigma_{i,j})$
4 :      if $\mathsf{VerBid}(\mathsf{pka}_i, \{\mathsf{pkr}_{j_*}\}_{j*\in[M]}, \mathbf{p}_i, b_{i,j}) \neq 1$
5 :        return 0
6 :    if $\exists (j_1, j_2) \in [M]^2, j_1 \neq j_2 \wedge \gamma_{i,j_1} = \gamma_{i,j_2}$
7 :      return 0
8 :  return 1
```

Figure 5: Paper bidding phase algorithms for the BlindReview protocol.

If any proof fails to verify, algorithm VerBid returns 0. If all proofs verify, algorithm VerBid returns the output of the signature verification algorithm on input the signature σ that is included in the bid b.

Additionally, anyone can check that each paper has received M bids from distinct PC members by verifying the bids and checking that the relevant γ values are distinct, as prescribed by the VerBidPool algorithm.

e) Paper Assignment: The BlindReview protocol proceeds to the interactive paper assignment phase once all bids are received. The algorithms for this phase are formalised in Figure 6. In our construction, we require that each PC member is assigned ℓ papers to review and that each submission is reviewed by three distinct PC members. The chair initiates paper assignment by running algorithm Assign, which selects a bid for a submission and assigns a PC member that produced the bid to review the paper. PC members can accept or reject the assignment by running algorithm AcceptAssign. The chair continues to assign submissions until all submissions are assigned three reviewers, and PC members will accept assignments until they have been assigned ℓ submissions. Thus, at best, the program chair must produce $3N$ assignments in total, i.e., $3N$ is a lower bound on the number of assignments.

Algorithm Assign, run by the chair, takes as input the secret key of the chair, and the set of all bids and corresponding bid values received for a submission, where we write that the set of all bids for a particular submission is the set BP. We assume that, throughout the interactive assignment phase, the chair updates the set BP when papers are assigned to reviewers. The highest bid value is chosen, and the associated bid vector is signed using the chair's secret key. Algorithm Assign returns an assignment a that includes the bid vector and the signature. Assignments can be verified by running algorithm VerAssign, which performs two checks. Firstly, VerAssign takes as input the set of bid vectors for a submission $\{b_i\}_{i\in[\mathsf{BP}]}$ and the assignment a, and checks that the bid included in the assignment is also in the set $\{b_i\}_{i\in[\mathsf{BP}]}$. If this check passes, algorithm VerAssign proceeds to verify the signature included in the assignment, taking as input the chair's public key to do so. If the signature verifies, VerAssign returns 1. If any check fails, the algorithm returns 0.

PC members accept or reject assignments by running algorithm AcceptAssign, which takes as input the PC member's secret key skr, the assignment a, the submission $\mathbf{p}$, the bid vector included in the assignment b, a set of bid vectors $\{b_i\}_{i\in[\mathsf{AB}]}$ where AB is the set of assigned bids, and a response. That is, AB is updated after each positive run of AcceptAssign to include all bids that have been assigned and accepted by reviewers. Algorithm AcceptAssign first creates a set of indices S such that S contains all assignments in AB that correspond to the PC member with secret key skr. Thus, the size of set S will be equal to the number of reviews that the PC member has accepted. The algorithm then proceeds to construct

```
$\mathsf{Assign}(\mathsf{skc}, \{(b_i, m_i)\}_{i\in[\mathsf{BP}]})$
1 : for $i$ s.t. $m_i = \max_{j\in[\mathsf{BP}]} m_j$ : $\sigma \leftarrow \mathsf{Sig}_g(\mathsf{skc}, b_i)$
2 : $a \leftarrow (b, \sigma)$
3 : return $a$

$\mathsf{AcceptAssign}(\mathsf{skr}, a, \mathbf{p}, b, \{b_i\}_{i\in[\mathsf{AB}]}, \mathtt{response})$
1 : parse $b$ as $(h, \gamma, \mathsf{pk}, \hat{\pi}, m, \sigma')$
2 : for $i \in [\mathsf{AB}]$
3 :     parse $b_i$ as $(h_i, \gamma_i, \mathsf{pk}_i, \hat{\pi}_i, m_i, \sigma_i)$
4 :     $S \leftarrow \{i \,:\, \mathsf{pk}_i = h_i^{\mathsf{skr}}\}$
5 : $\sigma \leftarrow \mathsf{Sig}_h(\mathsf{skr}, [a\|b\|\mathtt{response}])$
6 : if $\mathtt{response} = \mathtt{reject}$,
7 :     $\phi \leftarrow \mathsf{ZK.Prove}((\ell, b, \{b_i\}_{i\in[\mathsf{AB}]}), (\mathsf{skr}, S))$
8 : else $\phi \leftarrow \perp$
9 : return $\alpha \leftarrow (\mathtt{response}, \sigma, \phi)$

$\mathsf{VerAssign}(\mathsf{pkc}, \{b_i\}_{i\in[\mathsf{BP}]}, a)$
1 : parse $a$ as $(b, \sigma)$
2 : return $(b \in \{b_i\}_{i\in[\mathsf{BP}]} \wedge \mathsf{Ver}_g(\mathsf{pkc}, b, \sigma))$

$\mathsf{VerAccAssign}(a, b, \alpha, \{b_i\}_{i\in[\mathsf{BP}]})$
1 : parse $b$ as $(h, \gamma, \mathsf{pk}, \hat{\pi}, m, \sigma')$
2 : parse $\alpha$ as $(\mathtt{response}, \sigma, \phi)$
3 : if $\mathtt{response} \notin \{\mathtt{accept}, \mathtt{reject}\}$   return 0
4 : if $\mathsf{Ver}_h(\mathsf{pk}, [a, b, \mathtt{response}], \sigma) = 0$   return 0
5 : if $\mathsf{ZK.Verify}((\ell, b, \{b_i\}_{i\in[\mathsf{AB}]}), \phi) = 0$   return 0
6 : return $(1, \mathtt{response})$
```

Figure 6: Paper assignment algorithms for the BlindReview protocol.

and output an assignment response α that consists of three elements: a response `response`, a signature σ and a zero-knowledge proof of response correctness ϕ. If S is of size ℓ, i.e., the PC member has been assigned ℓ papers to review, the response `response` can be set to `reject` and the tuple $(\mathtt{response}, a, b)$ is signed using skr. Then, a zero-knowledge proof that set S is of size ℓ and contains indices for assignments accepted by the PC member is generated. That is, they generate a zero-knowledge proof for the following relation:

$$\mathsf{ZK}\left\{\mathsf{skr}, S : |S| = \ell \wedge \mathsf{pk} = h^{\mathsf{skr}} \wedge \left(\bigwedge_{i\in S} \mathsf{pk}_i = h_i^{\mathsf{skr}}\right)\right\}.$$

Accordingly, a PC member can only reject an assignment if they have already accepted ℓ papers to review. If the response is set to `accept` the tuple $(\mathtt{response}, a, b)$ is signed using skr. The zero-knowledge proof is set as empty. If S is neither less than, or equal to ℓ, the response, signature and zero-knowledge proof are set to empty. Algorithm VerAccAssign takes as input an assignment a, a bid vector b and a response α, and outputs a bit to indicate whether the acceptance is valid or not. In particular, it checks that the response is either `accept` or `reject`, and that the signature (and zero-knowledge proof, if available) verify.

f) Paper Review: Reviewers can generate reviews for a submission by running algorithm Review, as formalised in Figure 7. The reviewer first creates the review content w, and collects their secret key skr, the public keys of all PC members $\{\mathsf{pkr}_i\}_{i\in[M]}$, submission $\mathbf{p}$, author public key pka and bid b. Then, algorithm Review generates a signature on the review content with skr. Anyone can run algorithm VerReview to verify the signature on the review and that this was authored by a reviewer not in conflict with the submission (guaranteed by VerBid outputting 1). As is typical in the reviewing process, reviewers can discuss submissions, generate comments and update reviews. These updates can be signed by running algorithm Review, and the program chair can consider such updates alongside the review.

g) Decision Communication: For each submission $\mathbf{p}$ and corresponding public key pka, the chair collects all reviews and review contents $\{(w_i, z_i)\}_{i\in[3]}$ and produces a decision $\tilde{w}$ (e.g., accept or reject). The chair then runs algorithm Decision (Figure 8), which outputs a signature $\tilde{z}$ over the tuple $(\mathbf{p}, \mathsf{pka}, \{(w_i, z_i)\}_{i\in[3]}, \tilde{w})$. Algorithm VerDecision can be run by anyone to verify the signature $\tilde{z}$ and check the three reviews are distinct and authored by reviewers not in conflict with the paper.

h) Camera Ready: If a submission is accepted, the author generates a camera ready version of their paper `CRcontent`. The author then runs algorithm CameraReady (Figure 9) to generate a signature t_3, using $\mathsf{ska}_2 \in \mathsf{ska}$, over the tuple $(\mathbf{p}, \mathtt{alist}, \mathtt{content}, \mathtt{CRcontent})$. The camera ready version of the submission, output by algorithm CameraReady, is $t = (\mathsf{ska}_3, \mathsf{ska}_4, t_3)$. Accordingly, anyone can check that the commitments included in the submission $\mathbf{p}$ open correctly. Indeed, we define a final algorithm VerCameraReady that functions as follows. Parsing the submission $\mathbf{p}$ as $(p_1, p_2, p_3, p_4, p_5, p_6, p_7)$, algorithm CameraReady checks that p_1 opens to `alist` and p_2 opens to `content`. In this way, the chair can verify that the author list and paper content are the same as those submitted. We consider the process of checking similarity of the camera ready and submitted versions of the paper as out of scope of the cryptographic protocol.

```
$\mathsf{Review}(\mathsf{skr}, \{\mathsf{pkr}_i\}_{i\in[M]}, b, \mathbf{p}, \mathsf{pka}, w)$
1 : parse $b$ as $(h, \gamma, \mathsf{pk}, \hat{\pi}, m, \sigma)$
2 : $\omega \leftarrow \mathsf{Sig}_h(w, \mathsf{skr})$
3 : return $z \leftarrow (w, \omega)$
```

```
$\mathsf{VerReview}(\{\mathsf{pkr}_i\}_{i\in[M]}, \mathbf{p}, \mathsf{pka}, b, z)$
1 : parse $b$ as $(h, \gamma, \mathsf{pk}, \hat{\pi}, m, \sigma)$
2 : parse $z$ as $(w, \omega)$
3 : return $\mathsf{Ver}_h(w, \omega, \mathsf{pk}) \wedge \mathsf{VerBid}(\mathsf{pka}, \{\mathsf{pkr}_i\}_{i\in[M]}, \mathbf{p}, b)$
```

Figure 7: Paper review algorithms for the BlindReview protocol.

```
$\mathsf{Decision}(\mathsf{skc}, \mathbf{p}, \mathsf{pka}, (w_i, z_i)_{i\in[3]}, \tilde{w})$
1 : $\tilde{z} \leftarrow \mathsf{Sig}_g(\mathsf{skc}, [\mathbf{p}\|\mathsf{pka}\|(w_i, z_i)_{i\in[3]}\|\tilde{w}])$
2 : return $(\tilde{w}, \tilde{z})$
```

```
$\mathsf{VerDecision}(\mathsf{pka}, \mathsf{pkc}, \mathbf{p}, (b_i, w_i, z_i)_{i\in[3]}, \tilde{w}, \tilde{z})$
1 : for $i \in [3]$
2 :     parse $b_i$ as $(h_i, \gamma_i, \mathsf{pk}_i, \hat{\pi}_i, m_i, \sigma_i)$
3 : return $\gamma_1 \neq \gamma_2 \neq \gamma_3 \wedge \mathsf{Ver}_g(\mathsf{pkc}, [\mathbf{p}\|\mathsf{pka}\|(w_i, z_i)_{i\in[\![3]\!]}\|\tilde{w}], \tilde{z})$
4 :  $\wedge$ for $i \in [3]$ $\mathsf{VerReview}(\{\mathsf{pkr}_j\}_{j\in[M]}, \mathbf{p}, \mathsf{pka}, b_i, z_i)$
```

Figure 8: Decision communication algorithms for the BlindReview protocol.

The program chair has both versions and can determine the legitimacy of the camera ready paper. Then, it checks if the signature t_3 verifies. If all checks pass, CameraReady returns 1.

Design priorities and practical limitations Our protocol accounts for several design consideration, which we highlight here.

Privacy with respect to the chair: One of our design objectives is to provide privacy with respect to the chair. This is a strong property and it does not address all potential threats, such as collusion rings. However, it prioritises protection against bias or undue influence by the chair, which we believe is an important and underexplored concern in the peer-review process.

Conflicts of interest: We assume that the chair has no conflicts of interest and does not act as a reviewer. Conflicts are declared by authors; however, due to anonymity, the chair cannot validate them in the early phases of the reviewing process. The chair can instead verify and, if necessary, challenge authors declared conflicts a posteriori (at the camera-ready phase, when the authors' anonymity is removed), making this aspect of the process largely self-regulating. Note that, if desired, authors can reveal their identity to the chair before the camera-ready phase by opening their commitment without affecting the security of the rest of the protocol.

Reviewers explicitly declare a conflict by setting $m = 0$. An interesting extension of the protocol would allow reviewers to declare affiliations and lists of conflicted authors in a privacy-preserving manner; however, this would require non-trivial modifications and is outside the scope of the current design.

Assignment: We require that reviews are online during the assignment phase. Whilst this can be viewed as a practical limitation, we note that no interaction is required from reviewers beyond maintaining a connection, which allows the protocol to automatically generate proofs once assignment thresholds are reached. If reviewers are online within a coordinated time window, the assignment phase completes in under three hours. The current assignment mechanism is intentionally simple. It can be readily extended to incorporate additional parameters, such as reviewer confidence scores, which could be combined with bid values to enable more refined matching. Similarly, while we currently fix the number of reviewers per paper for simplicity, this is not a fundamental limitation and can be adjusted without affecting the core protocol.

IV. Security Analysis of BlindReview

A. Privacy Analysis

(2-a) In this section, we present our model and prove privacy and anonymity for our protocol. Intuitively, privacy is the property that no private information is leaked[1], and with anonymity that no private information *about the parties' identities* is leaked. In other words, anonymity ensures that authors and assigned reviewers are unknown to each other, and, crucially, no party (including the chair) is able to associate bids/evaluations with reviewers. However, unlike privacy, anonymity does not protect the content of the papers and the list of reviewer conflicts. Privacy therefore implies anonymity.

Our definitions are based on the real vs. simulation paradigm, and we define the adversary to be any collusion of dishonest parties. We assume that the adversary has access to algorithms that determine the choices and behaviours of honest parties (e.g., assigning marks, reviews). In this way, our model accounts for any information that the adversary may learn about honest parties. We call these *decision algorithms*.

More specifically, we present the following experiment to a distinguisher: we run the real protocol, and extract the view of each party, as well as the information

[1]We later specify what the private information is depending on the parties controlled by the attacker.

```
$\mathsf{CameraReady}(\mathsf{ska}, \mathbf{p}, \texttt{alist}, \texttt{content}, \texttt{CRcontent})$
1 : parse ska as $(\mathsf{ska}_1, \mathsf{ska}_2, \mathsf{ska}_3, \mathsf{ska}_4)$
2 : parse p as $(p_1, p_2, p_3, p_4, p_7)$
3 : $t' \leftarrow \mathsf{Sig}_g(\mathsf{ska}_2, [\mathbf{p}\|\texttt{alist}\|\texttt{content}\|\texttt{CRcontent}])$
4 : $t \leftarrow (\mathsf{ska}_3, \mathsf{ska}_4, t')$
5 : return $(\texttt{alist}, \texttt{content}, \texttt{CRcontent}, t)$
```

```
$\mathsf{VerCameraReady}(\mathsf{pka}, \mathbf{p}, \texttt{alist}, \texttt{content}, \texttt{CRcontent}, t)$
1 : parse $t$ as $(\mathsf{ska}_3, \mathsf{ska}_4, t_3)$
2 : parse p as $(p_1, p_2, p_3, p_4, p_7)$
3 : if $\mathsf{Ope}_{(g,\bar{g})}(\mathsf{ska}_3, \texttt{alist}, p_1) = \bot$   return 0
4 : if $\mathsf{Ope}_{(g,\bar{g})}(\mathsf{ska}_4, \texttt{content}, p_2) = \bot$   return 0
5 : return $\mathsf{Ver}_g(\mathsf{pka}_2, [\mathbf{p}\|\texttt{alist}\|\texttt{content}\|\texttt{CRcontent}], t_3)$
```

Figure 9: Camera ready algorithms for the BlindReview protocol.

learned by the parties controlled by the adversary (i.e., the content of the reviewed papers, and the conflicts with the reviewers controlled by the adversary) and information leaked inherently by our review assignment method (i.e., the order of anonymous bids, the marks, and the history of anonymous assignations with acceptances and rejections). We then simulate the views of users controlled by the adversary based on public information and these leaks. Finally, we draw a bit b. If $b = 0$, we give the views of the parties controlled by the adversary in the real protocol to the distinguisher; otherwise, we give it the views generated by the simulator. The protocol is private if for any collusion controlled by the adversary, there exists a simulator such that the advantage of any distinguisher is negligible regardless of the private information (paper contents, authors and conflicts) chosen as input.

Anonymity is defined in a similar way, but only guarantees the anonymity of reviewers and authors. The content of papers and conflicts are therefore considered public, i.e., known to all reviewers. Formally, this means that the simulator has access to this information in addition to that for privacy.

More formally, we first define a run of the real protocol for our security model (Definition 1). To simulate the choices made by honest parties we provide five decision algorithms as input to entities as follows: GenMark generates a mark from the paper content; GenReview generates a review from the paper content; GenResponse generates an assignment response that depends on the paper and other accepted assignments of a reviewer; GenDecision generates a decision from the paper reviews; and GenCR generates a camera-ready version depending on the reviews and decision it receives.

Definition 1 (BlindReview protocol for privacy)**.** *Consider M reviewer key pairs $(\mathsf{pkr}_i, \mathsf{skr}_i, \rho_i)$ and a chair key pair $(\mathsf{pkc}, \mathsf{skc})$, respectively generated by the algorithms $\mathsf{R.KGen}()$ and $\mathsf{C.KGen}()$. Let PK be the set of all the corresponding public keys. Throughout the protocol, we assume each party knows PK. BlindReview$\langle (A_i)_{i\in[N]}, (R_i)_{i\in[M]}, C\rangle$, involves the following parties modelled by algorithms that are polynomial in security parameter λ:*

- *N authors $\{A_i(\texttt{alist}_i, \texttt{content}_i, \texttt{RC}_i)\}_{i\in[N]}$ with binary strings $\texttt{alist}_i, \texttt{content}_i$, and $\texttt{RC}_i$ as input.*
- *M reviewers $\{R_i(\mathsf{skr}_i, \mathsf{GenMark}_i, \mathsf{GenReview}_i, \mathsf{GenResponse}_i)\}_{i\in[M]}$ with key skr_i and algorithms $\mathsf{GenMark}_i$, $\mathsf{GenReview}_i$ and $\mathsf{GenResponse}_i$ as input.*
- *Chair $C(\mathsf{skc}, \mathsf{GenDecision})$ with key skc and algorithm GenCR_i as input.*

Then the BlindReview protocol for privacy BlindReview$\langle (A_i)_{i\in[N]}, (R_i)_{i\in[M]}, C\rangle$ is defined in Figure 10.

We implicitly assume that, throughout the protocol, if an entity can verify data via a verification algorithm from the BlindReview construction, then the verification algorithm is run and the data verified. If a verification fails, the entity aborts the protocol.

The protocol stores the history of marks and assignments made during its run (and whether they were accepted or not). This information is learned by all reviewers during the protocol since they receive each bid and each assignment/response to verify them, and is therefore part of the inherent information leakage intended by our bid/assignment method. They are stored in the lists bidHistory and assignmentHistory. For each reviewer of index i, the protocol also stores the list $\mathsf{reviewerBid}_i$, which indicates, for a given reviewer, which marks correspond to the bids they have produced. At the end of the protocol, the tuple $\mathsf{leak}_i = (\mathsf{bidHistory}, \mathsf{reviewerBid}_i, \mathsf{assignmentHistory})$ contains everything that the protocol has inherently leaked as information to reviewer R_i, i.e., the marks of the bids, the marks that correspond to the reviewer, and the history of assignments and responses to those assignments. The list correspondingBid is used to maintain the correspondence between the marks history and the actual cryptographic bids, and is not returned as information leakage.

(2-a) In our privacy and anonymity model given in Definition 2, the simulator can also use the information about the papers inherently leaked to the parties in addition to these protocol leaks. We call this *content leaks*, formalised by the variable cleak. For instance, If the chair is honest but the collusion contains dishonest reviewers, cleak contains the information about the papers learned by these reviewers during the protocol, i.e., the content of the papers for which none of these reviewers are in conflict, and the list of dishonest reviewers in conflict for each of the other papers. Moreover, if the chair is dishonest, cleak contains the content of *all* papers and *all* conflicts, since this information is known to the chair (this information is needed to distribute the papers correctly). Similarly, in the case of anonymity, the content of the papers and the conflicts are not meant to be hidden, and are therefore included in cleak, regardless of the chair's honesty.

The *privacy* and *anonymity* of our protocol are defined as follows.

Definition 2. *BlindReview is* private *if:*

- *for any set of* attacking authors $(A_i)_{i\in\mathsf{AA}}$ *where* $\mathsf{AA} \subseteq [N]$*;*
- *any set of* attacking reviewers $(R_i)_{i\in\mathsf{AR}}$ *where* $\mathsf{AR} \subseteq [M]$ *and any set of corresponding alternative reviewer key generation algorithms* $(\mathsf{AR.KGen}_i)_{i\in\mathsf{AR}}$*;*
- *any boolean* AC *and, if* $\mathsf{AC} = 1$*, any* attacking chair C *with alternative chair key generation algorithm* $\mathsf{AC.KGen}$*;*

there exists a polynomial time algorithm Sim *such that:*

- *For any set of* N *authors* $(A_i)_{i\in[N]}$ *with respective inputs* $(\mathtt{alist}_i, \mathtt{content}_i, \mathtt{RC}_i, \mathsf{GenCR}_i)$ *containing the set of attacking authors* $(A_i)_{i\in\mathsf{AA}}$*;*
- *for any vector of (honestly) generated keys* $(\mathsf{pkr}_i, \rho_i, \mathsf{skr}_i) \leftarrow \mathsf{R.KGen}()$ *for all* $i \in [M]\backslash\mathsf{AR}$*, and any set of* M *reviewers* $(R_i)_{i\in M}$ *with respective inputs* $(\mathsf{GenMark}_i, \mathsf{GenReview}_i, \mathsf{GenResponse}_i)$ *containing the set of attacking reviewers* $(R_i)_{i\in\mathsf{AR}}$*;*
- *if* $\mathsf{AC} = 0$*, for any honest chair* C *with input* $\mathsf{GenDecision}$*;*

the following is negligible for any polynomial time distinguisher $\mathcal{D}$*:*

$$\epsilon_{\mathcal{D}}^{\mathsf{priv}}(\lambda) = \Big| \Pr\Big[\mathbf{Exp}^{\mathsf{priv}}_{\mathit{BlindReview},1,\mathcal{D}M,N}(\lambda,\ell) = 1\Big] - \Pr\Big[\mathbf{Exp}^{\mathsf{priv}}_{\mathit{BlindReview},0,\mathcal{D},M,N}(\lambda,\ell) = 1\Big]\Big|$$

where the experiment $\mathbf{Exp}^{\mathsf{priv}}_{\mathit{BlindReview},b,\mathcal{D},M,N}(\lambda,\ell)$ *is defined as in Figure 11 and the function* $\mathsf{contentLeaks}(\mathsf{AA}, \mathsf{AR}, \mathsf{AC}, (\mathtt{content}_i, \mathtt{RC}_i)_{i\in[N]})$ *returns a set* cleak *such that:*

- *if* $\mathsf{AC} = 1$*,* $\mathsf{cleak} = (\mathtt{content}_i, \mathtt{RC}_i)_{i\in[N]}$
- *else,* cleak *contains:*
  - $(\mathtt{content}_i, \mathtt{RC}_i)_{i\in\mathsf{AA}}$
  - *For all* $i \in [N]\backslash\mathsf{AA}$*, a set* $\mathtt{RC}'_i = \mathtt{RC}_i \cap \mathsf{AA}$ *(i.e., the indices of the dishonest reviewers in conflict with the author* i*)*
  - *For all* $i \in [N]\backslash\mathsf{AA}$*, if* $\mathtt{RC}'_i \neq \mathsf{AA}$ *(i.e., at least one reviewer is not in conflict with author* i*), the value* $\mathtt{content}_i$*.*

When the attacking parties generate their keys correctly and follow the protocol correctly, BlindReview is said to be private against honest-but-curious *adversaries, otherwise against* dishonest *adversaries. We define* anonymity *in the same way as privacy except that* $\mathsf{contentLeaks}(\mathsf{AA}, \mathsf{AR}, \mathsf{AC}, (\mathtt{content}_i, \mathtt{RC}_i)_{i\in[N]})$ *always returns* $\mathsf{cleak} = (\mathtt{content}_i, \mathtt{RC}_i)_{i\in[N]}$ *if* $\mathsf{AR} \neq \emptyset$ *or* $\mathsf{AC} \neq 0$*, i.e., if the collusion includes at least one reviewer or the chair (otherwise it returns* $\perp$*).*

(2-a) In concrete terms, this definition guarantees that a collusion of parties does not learn anything more than its inputs, the public values, and the values contained in leaks and cleak (which are the information inherently learned during a review protocol using our assignation algorithm by these parties). Once again, we stress that in our model we assume that the decision algorithms are public data known to all. This allows us to take into account the case where an adversary uses what they know about reviewers to de-anonymize reviewers. For example, a high score on a submission about isogeny-based cryptography makes it possible to deduce that the reviewer who most likely assigned this score is a specialist in the field, and thus to eliminate most of the possible reviewers. The attacker can therefore partially deduce the reviewers' decision algorithm. As it is not possible to quantify this partial knowledge, in our model we give access to perfect knowledge of the reviewers' decision algorithm. This idea is similar to the one used to model chosen plaintext attacks – since we do not know what the attacker knows about the plaintexts, we allow them to choose the messages themselves to give them perfect knowledge.

(2-d) To illustrate from another perspective what anonymity and privacy protect for reviewers, consider a scenario inspired by vote-swapping in e-voting where two reviewers (for instance R_1 and R_2) swap their bids and their review for one paper (the rest is the same in both cases). In other words, this means that for this paper, the bid and review of R_1 are generated with skr_2, and and vice versa. In theory, privacy should ensure that an adversary is unable to distinguish whether the swap took place or not, except for trivial cases inherent in the review process. In the following, we show how our model captures this scenario. We assume that the order of bids remains the same and that none of the reviewers are in conflict with the paper (in which case, the two executions would have been trivially distinguishable). Let us consider two cases. In the first, R_1 and R_2 never exceed the limit of ℓ papers. In this case, the leaks (the leak_i and cleak) used by the simulator are the same

Paper submission: Each A_i runs $(\mathsf{pka}_i, \mathsf{ska}_i) \leftarrow \mathsf{A.KGen}()$, then $\mathbf{p}_i \leftarrow \mathsf{Submit}(\mathsf{pkc}, \mathsf{ska}_i, \{\mathsf{pkr}_j\}_{j\in[M]}, \mathtt{alist}_i, \mathtt{content}_i, \mathtt{RC}_i)$, and sends $(\mathsf{pk}_i, \mathbf{p}_i)$ to C.
Paper distribution: C publishes the papers $(i, \mathsf{pka}_i, \mathbf{p}_i)_{i\in[N]}$ on the bulletin board. For all $i \in [M]$, C runs $(\mathsf{contents}_i, v_i) \leftarrow \mathsf{Distribute}(\mathsf{skc}, \{(\mathbf{p}_j, \mathsf{pka}_j)\}_{j\in[N]}, \mathsf{pkr}_i)$ and sends it to R_i.
Bidding: For all $j \in [N]$, each R_i runs $m \leftarrow \mathsf{GenMark}_i(j, v_{i,j})$, then R_i runs $b \leftarrow \mathsf{Bid}(\mathsf{pkc}, \mathsf{skr}_i, \mathsf{pka}_j, \mathbf{p}_j, v_{i,j}, m)$ and sends (j, m, b) to the chair (Note that R_i can choose the order in which he sends these bids). throughout this phase, for all $j \in [N]$, the chair maintains a multisetset BP_j containing all the pairs (m, b) sent by the reviewers during this phase. At the end of this phase, the chair broadcasts BP_j for all $j \in [N]$ to the reviewers.
The protocol maintains a vector $\mathsf{bidHistory}$ containing all the pairs (j, m) and a vector $\mathsf{correspondingBid}$ containing the corresponding b for each (j, m, b) sent by the reviewer. The protocol also generate a binary vector $\mathsf{reviewerBid}_i$ for each reviewer containing 1 at indices corresponding to a bid produced by R_i in $\mathsf{correspondingBid}$, 0 otherwise.
Assignment: C sets $\mathsf{AB} \leftarrow \emptyset$. For all $j \in [N]$, it set $\mathsf{AB}_j \leftarrow \emptyset$, and while $|\mathsf{AB}_j| \neq 3$:

- C runs $(a, m) \leftarrow \mathsf{Assign}(\mathsf{skc}, \mathsf{BP}_j)$ and publishes it on the bulletin board.
- The reviewer R_i who generated b runs $\alpha \leftarrow \mathsf{AcceptAssign}(\mathsf{skr}_i, a, \mathbf{p}_j, b, \mathsf{AB}, \mathtt{response})$ where if R_i has not yet accepted l papers to review then $\mathtt{response} = \mathtt{accept}$, else $\mathtt{response} \leftarrow \mathsf{GenResponse}(\mathtt{content}_j)$. It then publishes $(\mathtt{response}, \alpha)$ on the bulletin board.
- C removes b from BP_j. If $\mathtt{response} = \mathtt{accept}$, C adds b to AB_j and AB.

Throughout this phase, the protocol maintains an ordered vector $\mathsf{assignmentHistory}$ containing all the tuples $(x, \mathtt{response})$, where x is the index of the pair containing the bid b in $\mathsf{correspondingBid}$.
Review: For all $j \in [N]$, C sets $\mathsf{rev}_j \leftarrow \emptyset$. For all $j \in [N]$, and all $b \in \mathsf{AB}_j$, the reviewer R_i who generated b runs $w \leftarrow \mathsf{GenReview}_i(j, v_{i,j})$, then it runs $z \leftarrow \mathsf{Review}(\mathsf{skr}_i, \{\mathsf{pkr}_i\}_{i\in[M]}, \mathbf{p}_j, \mathsf{pka}_j, w, b)$ and sends (w, z) to C, who adds it to rev_j.
Decision: For all $j \in [N]$, C runs $\tilde{w}_j \leftarrow \mathsf{GenDecision}(j, \mathsf{rev}_j)$, then it runs $\tilde{z}_j \leftarrow \mathsf{Decision}(\mathsf{skc}, \mathbf{p}_j, \mathsf{pka}_j, \mathsf{rev}_j, \tilde{w}_j)$ and sends $(\mathsf{rev}_j, \tilde{w}_j, \tilde{z}_j)$ to A_j.
Closing phase: Each party P returns a set view_P containing all the values it has sent and received during the protocol in order. Moreover, the protocol returns $(\mathsf{leak}_i)_{i\in[M]} = (\mathsf{bidHistory}, \mathsf{reviewerBid}_i, \mathsf{assignmentHistory})_{i\in[M]}$.

Figure 10: The run of the protocol BlindReview$\langle (A_i)_{i\in[N]}, (R_i)_{i\in[M]}, C\rangle$ defined in Definition 1.

```
$\mathbf{Exp}^{\mathsf{priv}}_{\mathrm{BlindReview},b,\mathcal{D},M,N}(\lambda, \ell)$
**if** $\mathsf{AC} = 1,\ (\mathsf{pkc}, \mathsf{skc}) \leftarrow \mathsf{DC.KGen}((\mathsf{pkr}_i, \rho_i)_{i\notin \mathsf{AR}})$
**for** $i \in \mathsf{AR}$
  $(\mathsf{pkr}_i, \rho_i, \mathsf{skr}_i) \leftarrow \mathsf{DR.KGen}((\mathsf{pkr}_i, \rho_i)_{i\notin \mathsf{AR}}, \mathsf{pkc})$
  $((\mathsf{view}^0_{A_i})_{i\in[N]}, (\mathsf{view}^0_{R_i})_{i\in[M]}, \mathsf{view}^0_C, (\mathsf{leak}_i)_{i\in[M]}) \leftarrow \mathrm{BlindReview}\langle (A_i)_{i\in[N]}, (R_i)_{i\in[M]}, C\rangle$
**if** $\mathsf{AC} = 0,\ \mathsf{view}^0_C \leftarrow \perp$
$\mathsf{dalgo} \leftarrow ((\mathsf{GenCR}_i)_{i\in N}(\mathsf{GenMark}_i, \mathsf{GenReview}_i, \mathsf{GenResponse}_i)_{i\in[M]}, \mathsf{GenDecision}))$
$\mathsf{cleak} \leftarrow \mathsf{contentLeaks}(\mathsf{AA}, \mathsf{AR}, \mathsf{AC}, (\mathtt{content}_i, \mathtt{RC}_i)_{i\in[N]}$
$((\mathsf{view}^1_{A_i})_{i\in\mathsf{AA}}, (\mathsf{view}^1_{R_i})_{i\in\mathsf{AR}}, \mathsf{view}^1_C) \leftarrow \mathsf{Sim}((\mathsf{pkr}_i, \rho_i)_{i\in[M]}, \mathsf{pkc}, \mathsf{dalgo}, \mathsf{cleak}, (\mathsf{leak}_i)_{i\in\mathsf{AR}})$
$b \leftarrow \mathcal{D}((\mathsf{view}^b_{A_i})_{i\in\mathsf{AA}}, (\mathsf{view}^b_{R_i})_{i\in\mathsf{AR}}, \mathsf{view}^b_C)$
**return** $b$
```

Figure 11: The privacy experiment for Definition 2.

whether or not the reviewers have swapped. The two scenarios can therefore be simulated in the exact same way, so it is not possible to guess the swap in our privacy definition, which reflects the protection of the reviewers' real identities. To illustrate the other case, suppose that R_1 is the world expert on isogenies, and that they are the only reviewer of the PC interested in this topic. Suppose also that $\ell + 1$ papers on isogenies have been submitted to the conference. If there is no swap, R_1 will most likely be assigned the $\ell + 1$ papers, agree to review ℓ of them, and refuse to review the last one. On the other hand, if the reviewers have swapped, then R_2 will agree to review one of the papers, and R_1 will agree to review all the others. An observer will therefore be able to see that no paper assignment on isogenies has been rejected during the assignment phase, and will therefore be able to deduce that a reviewer other than R_1 was assigned a paper on isogenies. Note that this is to be expected, since this information inherently leaks in our assignment method and does not depend on the cryptography used to secure the protocol (i.e., it is the information contained in the leak_i and cleak). However, since the adversary must be able to simulate the rest of the protocol without the private inputs of honest users in our model, it will not be able to deduce anything more about the exact identity of the reviewer who accepted this assignment than what they learn from the assignment acceptances/rejections.

The privacy of our protocol is given in the following theorem.

Theorem 1. *When instantiated with extractable and zero-knowledge proofs, BlindReview satisfies* privacy *in the presence of honest-but-curious attackers under the DDH assumption in the random oracle model.*

As a corollary, BlindReview is anonymous under the same assumptions. The full proof of Theorem 1 is deferred to Appendix C. In what follows, we give an intuition for the proof by informally describing the construction of the simulator. We distinguish between two cases: the case where the chair is dishonest (i.e. controlled by the adversary), and the case where it is honest.

In the case that the chair is controlled by the adversary, the simulator has access to the contents of all papers by honest authors, the list of conflicts for each of these papers, and the decision algorithms of all honest reviewers. The idea of the simulator is to replace each proof/signature of knowledge with a simulation, replace commited/encrypted messages with random messages, and replace Diffie-Hellman between two honest user keys with random group elements. The technical challenge in the proof consists in correctly simulating the behaviour of honest reviewers during the bid/assignment/review phase. To do this, the simulator uses the reviewers' decision algorithms on the content of the papers and the conflicts to assign marks, keeps a list that matches the bids to the reviewers who are supposed to have produced them, and can then simulate the assignments and reviews as in the real protocol. We stress that in this way, privacy can be proven even in the presence of dishonest attackers in this case.

In the case where the chair is honest, the content of the papers for which all dishonest parties are in conflict is not given to the simulator, nor are conflicts with honest reviewers, so it is not possible to simulate the behaviour of honest reviewers for the bidding/assignment phases as easily as in the previous case. In order to simulate these phases correctly, the simulator uses the history of marks/assignments from the execution of the real protocol and follows the same steps in the simulation. However, a dishonest adversary could arbitrarily change their behaviour based on the values of the protocol elements, which would arbitrarily deviate the reviewer's acceptance phase from the actual execution of the protocol. For example, a dishonest reviewer could decide to give the maximum score to a paper if the last bit of one of the zero-knowledge proofs of an honest reviewer's bid is 0, and give the minimum score otherwise. Similarly, they could decide to reject or accept an assignment based on an arbitrary probabilistic event. If they do so, this will disrupt the assignment phase (for example, if the dishonest reviewer rejects an assignment that they accepted in the real protocol, the assignment may be reassigned to an honest reviewer, and that honest reviewer may later reject an assignment that they would have accepted). Considering this case, it is no longer possible to simulate the assignment phase correctly. It is therefore necessary to consider that adversaries will not behave arbitrarily in these phases, which is possible only in the honest-but-curious adversary model. This limitation seems generic and difficult to overcome, which is why we leave the design of a protocol that can be proven privacy-preserving in the dishonest adversary model as an open problem for future work.

Thus, our simulator works in all possible cases of collusion of at least honest-but-curious adversaries, allowing us to conclude the proof. As a notable side result, since our simulation works against fully dishonest attackers in the case where the chair is dishonest, we can deduce the following proposition.

Proposition 1. *When instantiated with extractable and zero-knowledge proofs, BlindReview satisfies* privacy *in the presence of dishonest attackers including the chair under the DDH assumption in the random oracle model.*

(2-c) This result may seem surprising: since an adversary controlling the chair is *more powerful*, one would expect the protocol should be *less secure* in this case.

But this intuition is misleading since actually assuming the chair dishonest means the amount of information that needs to be protected is less, and therefore this is easier to prove, even in a stronger adversary model in our case. This is particularly apparent in extreme cases: if the adversary controls all parties, then they know all private data, and there is nothing left to hide from them; any protocol is therefore unconditionally private when faced with such overly powerful adversary.

Finally, the anonymity of our protocol is given in the following theorem.

Theorem 2. *When instantiated with extractable and zero-knowledge proofs, BlindReview satisfies* anonymity *in the presence of dishonest attackers under the DDH assumption in the random oracle model.*

The full proof of Theorem 2 is also deferred to Appendix C. Recall that in the case of anonymity, the content of the papers and the conflicts are no longer assumed to be hidden from the reviewers, this information can therefore be used in the simulation regardless of whether the chair is honest or not. In the case of dishonest chair, the simulator is exactly the same as for privacy (recall that this case worked with dishonest adversaries). In the other case, the simulation is done in almost the same way, as the same information leak is given to the simulator. The only difference is that the simulator must simulate the behaviour of the chair without knowing its secret key, particularly during the distribution phase (for the other phases, simulating the signatures is sufficient). For the distribution algorithm, the simulator must simulate the decryption of the hash-ElGamal (p_4, p_5), which can be done by extracting the discrete logarithm r of p_4 from the proof of knowledge p_6 and computing $p_5/H(\mathsf{pkc}^r)$ instead of $p_5/H(p_4^{\mathsf{skc}})$. In this way, anonymity can be proven against any collusion of dishonest parties.

Thus, anonymity can be proven even in the dishonest adversary model. Since in practice the content of conflicting papers is sometimes not hidden from reviewers, this may therefore be sufficient in some cases when only seeking anonymity for authors and reviewers, and guarantees security against a stronger attacker model.

B. *Verifiability Analysis*

We define five verifiability properties, namely, submission authenticity (SA), bid eligibility (BE), fair assign (FA), review and decision authenticity (RD) and camera ready authenticity (CA). We define a formal experiment $\mathsf{Exp}^{\mathsf{XX}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda, \ell)$ for $\mathsf{XX} \in \{\mathsf{SA}, \mathsf{BE}, \mathsf{FA}, \mathsf{RD}, \mathsf{CA}\}$. In our experiments, an adversary and challenger interact and run consecutive stages of the BlindReview protocol, and the adversary can corrupt authors and reviewers. At various stages, depending on the value of XX, the adversary outputs a challenge $\mathsf{Ch}_{\mathsf{XX}}$ and the experiment returns a bit that is 1 if and only if the winning conditions for the experiment are satisfied, i.e., $\mathsf{WC}_{\mathsf{XX}} = 1$. We now describe each of our verifiability requirements, focusing on the challenge and winning conditions for each experiment, and then present formal definitions in Definitions 3–4. Our verifiability experiments are outlined in Figure 12.

*a) Submission Authenticity (*SA*):* Informally, submission authenticity provides the chair with the assurance that the submission was created by the submitting author. It captures an attacker that attempts to submit a paper on behalf of a registered author. In the experiment, the adversary and challenger need only interact to run the setup and submission phases, and the adversary is challenged to output a submission and an uncorrupted author index for which algorithm VerSubmit returns 1. That is, $\mathsf{Ch}_{\mathsf{SA}} = (i_*, \mathbf{p}_*)$ and $\mathsf{WC}_{\mathsf{SA}}$ equals 1 if and only if: (i) $\mathsf{VerSubmit}(\mathsf{pka}_{i_*}, \mathbf{p}_*) = 1$ and (ii) $(i_*, \mathbf{p}_*) \notin \mathcal{S}$ (i.e., submission $\mathbf{p}_*$ was submitted by the author corresponding to index i_* and the author was not corrupt).

*b) Bid Eligibility (*BE*):* With bid eligibility we capture the property that there is only one eligible bid per PC member per paper. That is, only PC members without conflicts of interest with a submission can bid for that submission, and a PC member cannot bid more than once on the same paper. In the bid eligibility experiment, the challenger and adversary must run the BlindReview protocol up to and including the bid phase. At the end of the experiment, the adversary outputs an uncorrupted author index and set of bids such that algorithm VerBidPool returns 1. More formally, the adversary outputs $\mathsf{Ch}_{\mathsf{BE}} = (i_*, (b_{i,j}, m_{i,j})_{j\in[M]})$ and we say that $\mathsf{WC}_{\mathsf{BE}}$ equals 1 if and only if (i) $i_* \in A_u$ (the author corresponding to index i_* was not corrupt); (ii) $\mathsf{VerBidPool}(\{\mathsf{pka}_i, \mathbf{p}_i, \{b_{i,j}\}_{j\in[M]}\}_{i\in[N]}, \{\mathsf{pkr}_i\}_{i\in[M]}) = 1$, and (iii) $|B_c| > |R_c| \vee |\{j | b_{i_*,j} \in \mathsf{B}_c \wedge m_{i_*,j} \neq 0\}| > |R_c \backslash \mathtt{RC}_{i_*}|$. In our third winning condition, we define B_c as the subvector of bids $(b_{i_*,j})_{j\in[M]}$ that was not generated by $\mathcal{C}$ and write $R_c \backslash \mathtt{RC}_{i_*}$ to be the set of corrupted reviewers that have no conflict with the paper i_*. Then, condition (iii) means that, to succeed, the adversary must produce more bids than they are supposed to or agree to review more papers than they are supposed to. In the first case, the adversary must either bid twice for a submission with the same PC member key, or produce a bid on behalf of an honest PC member. In the second, the adversary must produce a bid with a non-zero score for a conflicting PC member, bid twice for a corrupt PC member not in conflict, or bid for a non-corrupt reviewer who is not in conflict.

*c) Fair Assign (*FA*):* Fair assign protects against a PC member that wants to do less work than oth-

The verifiability experiment $\mathsf{Exp}^{\mathsf{XX}}_{\mathcal{A},\textbf{BlindReview},M,N}(\lambda,\ell)$:

1) **Setup phase**: Challenger $\mathcal{C}$ completes the setup phase of the protocol, generating key pairs and, if relevant, correct key construction proofs for the chair, M PC members and N authors. $\mathcal{C}$ provides $\mathcal{A}$ with the set of public keys and proofs $(\mathsf{pkc}, \{(\mathsf{pkr}_i, \rho_i)\}_{i\in[M]}, \{(\mathsf{pka}_j, \tau_j)\}_{j\in[N]})$.
2) **Submission phase**: For each author pka_j, $\mathcal{A}$ schedules the Submit protocol and chooses whether or not to control the author.
   - If $\mathcal{A}$ controls pka_j, $\mathcal{C}$ provides $\mathcal{A}$ with the author's secret key ska_j. $\mathcal{A}$ produces, and outputs to $\mathcal{C}$, a submission $\mathbf{p}_j$. In the role of the chair, $\mathcal{C}$ posts $\mathbf{p}_j$ to the submission server.
   - Else, $\mathcal{A}$ outputs $(\mathtt{alist}_j, \mathtt{content}_j, \mathtt{RC}_j)$ and $\mathcal{C}$ runs algorithm Submit on behalf of the author. and posts submission $\mathbf{p}_j$ to the submission server. Index j is added to set A_u and $(j, \mathbf{p}_j)$ is added to a set $\mathcal{S}$.

   $\hookrightarrow$ **If $\mathsf{XX} = \mathsf{SA}$, $\mathcal{A}$ outputs $\mathsf{Ch}_{\mathsf{SA}}$ and the experiment proceeds to the Return phase.**
3) **Distribution phase**: $\mathcal{C}$ outputs a distribution vector $(v_{i,1}, \dots v_{i,N})$ for each reviewer pkr_i.
4) **Bid phase**: For each PC member pkr_i, $\mathcal{A}$ schedules the Bid protocol and chooses whether or not to control the PC member. At the end of the bid phase, $\mathcal{C}$ posts all bids to the submission server.
   - If $\mathcal{A}$ controls pkr_i, $\mathcal{C}$ provides $\mathcal{A}$ with skr_i. $\mathcal{A}$ produces a bid $b_{i,j}$ for each paper submission $\mathbf{p}_j$, outputting $(b_{i,1}, \dots, b_{i,N})$ to $\mathcal{C}$. Index i is added to set R_c and $(b_{i,1}, \dots, b_{i,N})$ is added to set B_c.
   - Else, $\mathcal{A}$ outputs scores $(m_{i,1}, \dots, m_{i,N})$. $\mathcal{C}$ runs algorithm Bid for each $j \in [N]$ to generate $(b_{i,1}, \dots, b_{i,N})$.

   $\hookrightarrow$ **If $\mathsf{XX} = \mathsf{BE}$, $\mathcal{A}$ outputs $\mathsf{Ch}_{\mathsf{BE}}$ and the experiment proceeds to the Return phase.**
5) **Assign phase**: $\mathcal{C}$ runs algorithm Assign to output assignments for each submission. $\mathcal{C}$ maintains a list $\mathcal{L} = \{(a_k, \mathbf{p}'_k, b_k, \mathtt{response}_k, \alpha_k)\}$ that tracks all assignments, responses, and the corresponding bid vector.
   - If $\mathcal{A}$ controls pkr_i, $\mathcal{A}$ generates a response α_i and returns the response to $\mathcal{C}$.
   - Else, $\mathcal{C}$ runs algorithm AcceptAssign to generate a response α_i.

   $\hookrightarrow$ **If $\mathsf{XX} = \mathsf{FA}$, $\mathcal{A}$ outputs $\mathsf{Ch}_{\mathsf{FA}}$ and the experiment proceeds to the Return phase.**
6) **Review phase**: For each submission $\mathbf{p}$, $\mathcal{A}$ schedules the Review protocol on behalf of each PC member.
   - If $\mathcal{A}$ controls pkr_i, $\mathcal{A}$ produces, and outputs to $\mathcal{C}$, a review (w, z) for $\mathbf{p}$.
   - Else, $\mathcal{A}$ outputs the review content w. $\mathcal{C}$ runs algorithm Review to generate the review (w, z).
7) **Decision phase**: $\mathcal{A}$ can query $\mathcal{C}$ on decisions by outputting a tuple $(\mathbf{p}, \mathsf{pka}, \{(w_k, z_k)\}_{k\in[3]}, \tilde{w})$. $\mathcal{C}$ runs algorithm Decision on $\mathcal{A}$'s output and provides a decision output $\tilde{z}$.

   $\hookrightarrow$ **If $\mathsf{XX} = \mathsf{RD}$, $\mathcal{A}$ outputs $\mathsf{Ch}_{\mathsf{RD}}$ and the experiment proceeds to the Return phase.**
8) **Camera ready phase**: $\mathcal{A}$ schedules the CameraReady protocol on behalf of each author pka_j.
   - If $\mathcal{A}$ controls pka_j, $\mathcal{C}$ provides $\mathcal{A}$ with the author's secret key ska_j. $\mathcal{A}$ produces a camera ready version for submission $\mathbf{p}_j$ on behalf of the author pka_j.
   - Else, $\mathcal{A}$ outputs $(\mathbf{p}_j, \mathtt{alist}_j, \mathtt{content}_j, \mathtt{CRcontent}_j)$. $\mathcal{C}$ runs algorithm CameraReady and returns camera ready version t_j to $\mathcal{A}$. Tuple $(\mathbf{p}_j, \mathtt{alist}_j, \mathtt{content}_j, \mathtt{CRcontent}_j)$ is added to list CR_u.

   $\hookrightarrow$ **If $\mathsf{XX} = \mathsf{CA}$, $\mathcal{A}$ outputs $\mathsf{Ch}_{\mathsf{CA}}$ and the experiment proceeds to the Return phase.**
9) **Return phase**: The experiment returns a bit which is 1 if and only if $\mathsf{WC}_{\mathsf{XX}} = 1$.

Figure 12: The verifiability experiment $\mathsf{Exp}^{\mathsf{XX}}_{\mathcal{A},\mathrm{BlindReview},M,N}(\lambda,\ell)$ for our construction BlindReview where $\mathsf{XX} \in \{\mathsf{SA}, \mathsf{BE}, \mathsf{FA}, \mathsf{RD}, \mathsf{CA}\}$.

ers. It ensures that a PC member can only refuse to accept an assignment if they have been assigned the maximum number of papers (ℓ). In our experiment, the adversary and challenger interact until the assignment phase is complete, and we challenge the adversary to $\mathsf{Ch}_{\mathsf{FA}} = (k_*, j_*)$. We say that $\mathcal{L}_*$ is the subset of $\mathcal{L}$ containing the tuples $(a_k, \mathbf{p}'_k, b_k, \mathtt{response}_k, \alpha_k)$ for all $k \le k_*$ such that α_k was produced by the adversary. We write k'_* to be the number of elements in $\mathcal{L}$ such that $\mathtt{response}_k = 1$. We also set i_* as the integer that verifies $i_* \times \ell \le k'_* \le (i_* + 1) \times \ell$. Then, $\mathsf{WC}_{\mathsf{FA}}$ equals 1 if and only if one of the following conditions holds: (i) $\exists (a_k, \mathbf{p}'_k, b_k, \mathtt{response}_k, \alpha_k) \in \mathcal{L}_*$ such that b_k was produced by an honest reviewer (the adversary cannot respond to an assignation on behalf of an honest reviewer); (ii) $|\{(a_k, \mathbf{p}'_k, b_k, \mathtt{response}_k, \alpha_k) \in \mathcal{L}_* | \mathbf{p}'_k = \mathbf{p}_{j_*} \wedge \mathtt{response}_k = 0\}| \ge i_* + 1$ (the adversary cannot reject papers if they have not yet accepted ℓ papers); (iii) a_{k_*} was not generated by the challenger (the adversary cannot assign a paper by themselves).

*d) Review & Decision Authenticity (*RD*):* Intuitively, review & decision authenticity captures the idea that an author can verify that their submission received three distinct reviews authored by non-conflicting reviewers who were assigned the review by the chair and accepted the assignment. It further guarantees that the final decision is made by the chair. In the review & decision authenticity experiment, the adversary outputs an author key index i_*, a set of three tuples of bids/marks/reviews $\{(b_j, m_j, w_j, z_j)\}_{j\in[3]}$, and a decision $(\tilde{w}, \tilde{z})$ such that all bids and the decision verify. The adversary outputs $\mathsf{Ch}_{\mathsf{RD}} = (i_*, (b_j, m_j, w_j, z_j)_{j\in[3]}, \tilde{w}, \tilde{z})$ and $\mathsf{WC}_{\mathsf{RD}}$ equals 1 if and only if: (i) $i_* \in A_u$; (ii) $\forall j \in [3]\colon \mathsf{VerBid}(\mathsf{pka}_{i_*}, \{\mathsf{pkr}_i\}_{i\in[M]}, \mathbf{p}_{i_*}, b_j) = 1$; (iii) $\mathsf{VerDecision}(\mathsf{pka}_{i_*}, \{\mathsf{pkr}_i\}_{i\in[M]}, \mathsf{pkc}, (w_j, z_j)_{j\in[3]}, \tilde{w}, \tilde{z}) = 1$ and (iv) $|R_c \backslash \mathtt{RC}_{i_*}| < |S_c| \vee (\tilde{w}, \tilde{z})$ was not generated by $\mathcal{C}$. We define S_c to be the subvector of reviews $(w_j, z_j)_{j\in[3]}$ that were not generated by $\mathcal{C}$. Then, condition (iv) requires that the adversary has produced more reviews than they should, either by reviewing twice for the same reviewer or generating a review on behalf of an honest or conflicting reviewer. Alternatively, we require that the adversary can produce a decision message different to the one produced by the chair.

*e) Camera-Ready Authenticity (*CR*):* Intuitively, camera-ready authenticity captures the idea that an attacker cannot submit a camera-ready version on behalf of an author. In the experiment, the adversary interacts with the challenger throughout all phases of the BlindReview protocol and is challenged to output a camera-ready submission such that VerCameraReady returns `accept` but it is not the authentic version by the author. More formally, the adversary returns $\mathsf{Ch}_{\mathsf{CA}} = (i_*, \mathbf{p}_*, \mathtt{alist}_*, \mathtt{content}_*, \mathtt{CRcontent}_*, t_*)$. To succeed, we require that the challenge author i_* is honest and that the updated content $\mathtt{CRcontent}_*$ is not obtained by the adversary from the challenger acting on behalf of an honest author. We say that $\mathsf{WC}_{\mathsf{CA}}$ equals 1 if and only if: (i) $\mathsf{VerCameraReady}(\mathsf{pka}_{i_*}, \mathbf{p}_*, \mathtt{alist}_*, \mathtt{content}_*, \mathtt{CRcontent}_*, t_*)$ outputs `accept`; (ii) $i_* \in A_u$ and (iii) $(\mathbf{p}_*, \mathtt{alist}_*, \mathtt{content}_*, \mathtt{CRcontent}_*) \notin CR_u$.

Formally, we obtain the following definitions of XX verifiability for XX $\in \{\mathsf{SA}, \mathsf{BE}, \mathsf{FA}, \mathsf{RD}, \mathsf{CA}\}$, and of end-to-end verifiability, and we state our verifiablity results (with full proofs in Appendix D).

Definition 3 (XX Verifiability)**.** *The BlindReview protocol satisfies* XX verifiability *if, for any probabilistic polynomial time adversary $\mathcal{A}$, there exists a negligible function* negl *such that*

$$\Pr\left[\mathsf{Exp}^{\mathsf{XX}}_{\mathcal{A},\mathit{BlindReview},M,N}(\lambda, \ell) = 1\right] \leq \mathsf{negl}(\lambda)$$

where $\mathsf{Exp}^{\mathsf{XX}}_{\mathcal{A},\mathit{BlindReview},M,N}(\lambda, \ell)$ is the experiment described in Figure 12 and XX $\in \{\mathsf{SA}, \mathsf{BE}, \mathsf{FA}, \mathsf{RD}, \mathsf{CA}\}$.

Definition 4 (End-to-end verifiability)**.** *We say that the BlindReview protocol is* end-to-end verifiable *if it satisfies* XX *verifiability for all* XX $\in \{\mathsf{SA}, \mathsf{BE}, \mathsf{FA}, \mathsf{RD}, \mathsf{CA}\}$ *(Definition 3).*

Theorem 3. *Let $(\mathsf{Sig}_g, \mathsf{Ver}_g)$ be a signature scheme that satisfies unforgeability [17], and let (Com, Ope) be a binding commitment scheme [18]. Let the zero-knowledge proof systems used in the protocol satisfy extractability and zero-knowledge [19]. Then BlindReview is an end-to-end verifiable peer review system.*

The proof of this result follows directly from the security of the used building blocks. Submission authenticity and camera ready authenticity reduce to the EUF-CMA security of the signature scheme. We prove our remaining verifiability properties, namely, bid eligibility, fair assign and review and decision authenticity, via a series of game hops, relying on the extractability and zero-knowledge properties of the proof systems.

V. Implementation and Efficiency Analysis

We present a theoretical evaluation of the asymptotic computational complexities of the algorithms used to instantiate BlindReview (cf. Table II, Appendix A), and we implement our protocol providing the actual execution times obtained from our experiments.

a) Instantiation: We instantiate the commitment scheme with Pedersen's commitment [15], which requires a constant number of exponentiations for both commitment and verification. Signatures and proofs of knowledge of a discrete logarithm are both instantiated with the non-interactive version of the Schnorr protocol [14], [16], which requires a constant number of exponentiations to prove/sign and verify. The proof π in algorithm Bid is instantiated using the transformation given in [20] (which allows to prove the knowledge of a witness in relation to one of several instances) on the variant of the Schnorr protocol for proving discrete logarithms equality [21]. This proof requires a linear number of exponentiations (in M) for its generation and verification. The proofs $\hat{\pi}_j$ are instantiated by a variant of the non-interactive proof proposed in [22] to prove the inequality of two discrete logarithms, which is generated/verified in constant time. The proof of knowledge ϕ in AcceptAssign is instantiated using the transformation given in [20] on the proof of discrete logarithms equality [21]. This proof requires a linear number of exponentiations in MN. In Appendix B, we provide full details of the proofs/signatures of knowledge we use to instantiate our protocol.

b) Implementation: We implemented our protocol in Rust, and our source code is available at [23]. We use the elliptic curve-based group Ristretto (of prime order

	M	N	ℓ	**Conflicts per paper**	**Full protocol execution**
IMACC	21	34	6	1	17 seconds
CSF	61	205	12	3	24 minutes
ESORICS	124	334	10	5	161 minutes

(a) Parameters for our experiments. We set M reviewers and N papers where each paper is reviewed by 3 PC members and each reviewer submits ℓ reviews.

	Submit	Distribute	Bid	Assign	AcceptAssign (accepted)	AcceptAssign (rejected)	AcceptAssign (max)	Review	Decision
IMACC	466	2165	8650	692	1987	106782	151260	35	127
CSF	885	14584	26005	1897	10425	14 seconds	19 seconds	36	249
ESORICS	1506	25454	51297	3850	17032	57 seconds	80 seconds	37	444

(b) Timings for each phase of BlindReview

	Submit	Distribute	Bid (pool)	Assign	AcceptAssign (accepted)	AcceptAssign (rejected)	Review	Decision
IMACC	195	2674	6 seconds	717	646	11262	9689	29028
CSF	313	17025	5 minutes	1920	1730	105361	27906	84246
ESORICS	500	27034	38 minutes	3859	3479	210698	56658	171963

(c) Timings for verification of each phase of BlindReview

Table I: Timings for BlindReview. Timings are in microseconds unless stated otherwise.

$2^{255}-19$) from the Dalek library. In addition, we use the sha2 hash function. All dependencies in our package are *curve25519-dalek*, *rand*, *rand_core*, *subtle*, *sha2*, *hex*, *hex-literal*, and *concat*. In our implementation, we run all the steps of the protocol sequentially and perform verification algorithms on all generated elements. Note that in a concrete use of our protocol, some steps would be executed in parallel (submissions, bids, reviews, etc.). In order to ensure that the assignment phase can always be successfully completed, when the assignment reaches a deadlock (all reviewers have refused to review a paper because they have all reached the limit ℓ), the limit ℓ is incremented and the assignment phase for the current paper is restarted. We ran our implementation on an HP EliteBook 840 14-inch G10 Notebook PC with a 13th Gen Intel® Core™ i5-1345U × 12 processor.

In Table I we show the average time in microseconds taken by each algorithm for relatively small, medium and large conferences (data from IMACC, CSF and ESORICS 2023), averaged over 10 full executions of the protocol. We note that for assignment the execution time depends greatly on the number of assignments already accepted, and whether the reviewer rejects or accepts the assignment (acceptance is very efficient, while rejection requires partial proof of knowledge, which depends on the number of assignments already accepted). Finally, we give the time for the full sequential execution of all the phases of the protocol, e.g., 17 seconds for IMACC, 24 minutes for CSF and under 2.5 hours for ESORICS.

Our implementation demonstrates that the protocol is highly efficient for small and medium conferences and remains practical for larger ones. Performance is most affected by assignment-phase rejections, which can be mitigated by increasing reviewer capacity or availability. Overall, BlindReview achieves verifiable privacy and anonymity at small additional cost.

References

[1] N. Unger, S. Dechand, J. Bonneau, S. Fahl, H. Perl, I. Goldberg, and M. Smith, "Sok: secure messaging," in *2015 IEEE Symposium on Security and Privacy*. IEEE, 2015, pp. 232–249.

[2] T. Haines, J. Mueller, R. Mosaheb, and I. Pryvalov, "Sok: Secure e-voting with everlasting privacy," in *Privacy Enhancing Technologies Symposium (PETS)*, 2023.

[3] K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila, "A formal security analysis of the signal messaging protocol," *Journal of Cryptology*, vol. 33, pp. 1914–1983, 2020.

[4] V. Cortier, D. Galindo, R. Küsters, J. Müller, and T. Truderung, "Sok: Verifiability notions for e-voting protocols," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 779–798.

[5] EasyChair, "Conference management system." [Online]. Available: https://easychair.org/

[6] HotCRP, "Conference management system." [Online]. Available: https://hotcrp.com/

[7] M. Arapinis, S. Bursuc, and M. Ryan, "Privacy supporting cloud computing: Confichair, a case study," in *Principles of Security and Trust - First International Conference, POST 2012,*, ser. Lecture Notes in Computer Science, P. Degano and J. D. Guttman, Eds., vol. 7215. Springer, 2012, pp. 89–108. [Online]. Available: https://doi.org/10.1007/978-3-642-28641-4_6

[8] S.-W. Lo, R. Phan, and B.-M. Goi, "On the security of a popular web submission and review software (wsar) for cryptology conferences," 2007.

[9] S. Halevi, "Web submission and review software (wsar)." [Online]. Available: https://shaih.github.io/websubrev/

[10] R. Cho and D. Fun, "Verifying information confidentiality under query optimization in hotcrp," in *Projects and Close Readings in Software Systems, Harvard University*, 2017.

[11] H. Qunoo and M. Ryan, "Modelling dynamic access control policies for web-based collaborative systems," in *Data and Applications Security and Privacy XXIV, 24th Annual IFIP WG 11.3 Working Conference, Proceedings*, ser. Lecture Notes in Computer Science, vol. 6166. Springer, 2010, pp. 295–302.

[12] ——, "Modelling and analysing dynamic access control policies using atomic actions," in *International Journal of Advanced Computer Technology*, 2013.

[13] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[14] C.-P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology—CRYPTO'89 Proceedings 9*. Springer, 1990, pp. 239–252.
[15] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Annual international cryptology conference*. Springer, 1991, pp. 129–140.
[16] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Conference on the theory and application of cryptographic techniques*. Springer, 1986, pp. 186–194.
[17] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM Journal on computing*, vol. 17, no. 2, pp. 281–308, 1988.
[18] J. Katz and Y. Lindell, *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.
[19] J. Groth, R. Ostrovsky, and A. Sahai, "Perfect non-interactive zero knowledge for np," in *Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28-June 1, 2006. Proceedings 25*. Springer, 2006, pp. 339–358.
[20] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Advances in Cryptology—CRYPTO'94: 14th Annual International Cryptology Conference Santa Barbara, California, USA August 21–25, 1994 Proceedings*. Springer, 2001, pp. 174–187.
[21] D. Chaum and T. P. Pedersen, "Wallet Databases with Observers," in *Advances in Cryptology — CRYPTO' 92*. Springer, 1993.
[22] J. Camenisch and V. Shoup, "Practical Verifiable Encryption and Decryption of Discrete Logarithms," in *Advances in Cryptology - CRYPTO 2003*, D. Boneh, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 126–144.
[23] Anonymous, "BlindReview source code." [Online]. Available: https://anonymous.4open.science/r/BlindReview-B515
[24] M. Blum, P. Feldman, and S. Micali, "Non-Interactive Zero-Knowledge and Its Applications," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, ser. STOC '88. New York, NY, USA: Association for Computing Machinery, 1988, p. 103–112. [Online]. Available: https://doi.org/10.1145/62212.62222
[25] E. Bangerter, T. Briner, W. Henecka, S. Krenn, A.-R. Sadeghi, and T. Schneider, "Automatic generation of sigma-protocols," in *Public Key Infrastructures, Services and Applications*, F. Martinelli and B. Preneel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 67–82.
[26] A. Fiat and A. Shamir, "How To Prove Yourself: Practical Solutions to Identification and Signature Problems," in *CRYPTO' 86*. Springer, 1987.
[27] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, p. 612–613, Nov. 1979. [Online]. Available: https://doi.org/10.1145/359168.359176

Appendix A
Further details on implementation and asymptotic complexity

In Figure 13, we show the polynomial growth of the protocol execution time (where all algorithms are ran sequentially) as a function of the number N of papers submitted. The number of reviewers $M = N/2$ and the number of conflicts $C = \lfloor M/20 \rfloor$ increase proportionally with the number of papers. For each N, each paper receives 3 reviews, and the maximum number of reviews per reviewer is 8. The time is given for $0 \leq N \leq 200$ (for each multiples of 20). Thus, $0 \leq M \leq 100$ and $0 \leq C \leq 5$.

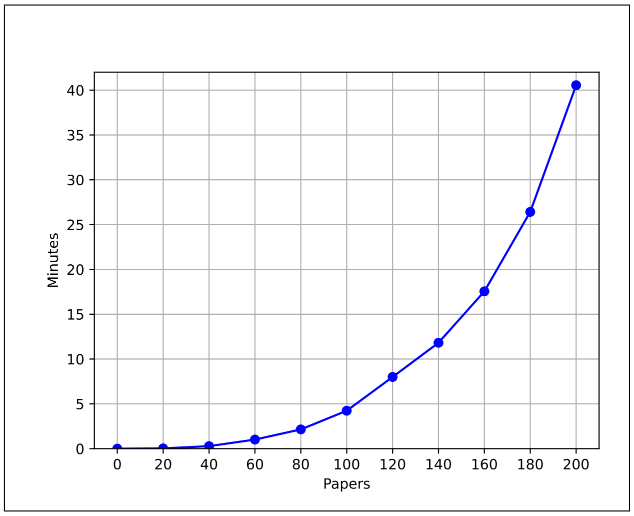

Figure 13: Execution time of BlindReview as a function of the papers

Finally, in Table II, we give the complexity (in number of exponentiations in a prime order group) of each algorithm of our protocol instantiation. We also give the number of times each algorithm is run per user involved in the protocol. For example, the chair runs the Distribute algorithm M times, but each reviewer only runs the VerDistribute algorithm once to verify their own distribution.

Appendix B
Cryptographic Tools

Here, we recall the cryptographic primitives, security properties and hardness assumptions that we require for our BlindReview protocol and its security proofs.

A. Cryptographic Hardness Assumptions

Definition 5 (Decisional Diffie-Hellman (DDH) assumption)**.** *Let $\mathbb{G} = \langle g \rangle$ be a multiplicative group of prime order $p > 2^\lambda$, where λ is a security parameter. The Decisional Diffie-Hellman (DDH) assumption ensures that for any polynomial-time (in λ) adversary $\mathcal{A}$, the following is negligible:*

$$\epsilon_{\mathcal{A}}^{\mathsf{DDH}}(\lambda) = \Big| \Pr\big[(x,y,z) \leftarrow\$ (\mathbb{Z}_p^*)^3 \colon \mathcal{A}(g^x, g^y, g^z) = 1\big] - \Pr\big[(x,y) \leftarrow\$ (\mathbb{Z}_p^*)^2;\ z \leftarrow xy \colon \mathcal{A}(g^x, g^y, g^z) = 1\big] \Big|$$

We define the advantage on DDH as $\epsilon^{\mathsf{DDH}}(\lambda) = \max_{\mathcal{A} \in \mathsf{poly}(\lambda)} \epsilon_{\mathcal{A}}^{\mathsf{DDH}}(\lambda)$.

Definition 6 (Fixed n-Decisional Diffie-Hellman (n-DDH) assumption)**.** *Let $\mathbb{G} = \langle g \rangle$ be a multiplicative group of prime order $p > 2^\lambda$, where λ is a security*

Algorithm	KGen	Submit	Distribute	Bid	Assign	AcceptAssign	Review	Decision
Runs	1	N	M	N	MN	N	3	N
Complexity	$O(1)$	$O(M)$	$O(MN)$	$O(M)$	$O(1)$	$O(MN)^*$	$O(1)$	$O(1)$

(a) Complexity for each phase of BlindReview.

Algorithm	verKGen	VerSubmit	VerDistribute	VerBidPool	VerAssign	VerAccAssign	VerReview	VerDecision
Runs	MN	N	1	1	MN	MN	$3N$	1
Complexity	$O(1)$	$O(1)$	$O(MN)$	$O(NM^2)$	$O(1)$	$O(MN)$	$O(1)$	$O(1)$

(b) Complexity for verification of each phase of BlindReview.

Table II: Number of algorithm runs per user and complexity of each algorithm in our peer review protocol instantiation. Algorithm AcceptAssign requires $O((MN)^2)$ multiplications.

parameter. The Fixed n-Decisional Diffie-Hellman (n-DDH) assumption ensures that for any polynomial-time (in λ) adversary $\mathcal{A}$, the following is negligible:

$$\epsilon_{\mathcal{A}}^{\mathsf{n-DDH}}(\lambda) = \Big|\mathsf{Pr}\big[x \leftarrow\$ \mathbb{Z}_p^*;\ \{\forall\, i \in [n]\colon (y_i, z_i) \leftarrow\$ (\mathbb{Z}_q^*)^2\}\colon \mathcal{A}(g^x, \{(g^{y_i}, g^{z_i})\}_{i\in[n]}) = 1\big] - \mathsf{Pr}\big[(x, y_1, \ldots, y_n) \leftarrow\$ (\mathbb{Z}_p^*)^{n+1};\ \{\forall\, i \in [n]\colon z_i \leftarrow x{\cdot}y_i\}\colon \mathcal{A}(g^x, \{(g^{y_i}, g^{z_i})\}_{i\in[n]}) = 1\big]\Big|$$

We define the advantage on DDH as $\epsilon^{\mathsf{n-DDH}}(\lambda) = \max_{\mathcal{A}\in\mathsf{poly}(\lambda)} \epsilon_{\mathcal{A}}^{\mathsf{DDH}}(\lambda)$.

B. Cryptographic Building Blocks

Definition 7 (IND-CPA). *An encryption scheme E=($E.KGen$, $E.Enc$, $E.Dec$) with security parameter λ is said to be* indistinguishable under adaptive chosen plaintext attacks *(IND-CPA) if for any two-party polynomial-time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the following is negligible:*

$$\epsilon_{\mathcal{A}}^{\mathsf{IND-CPA}}(\lambda) = \Big|\Pr\Big[\mathbf{Exp}_{1,\mathsf{E},\mathcal{A}}^{\mathsf{IND-CPA}}(\lambda) = 1\Big] - \Pr\Big[\mathbf{Exp}_{0,\mathsf{E},\mathcal{A}}^{\mathsf{IND-CPA}}(\lambda) = 1\Big]\Big|$$

where $\mathbf{Exp}_{b,\mathsf{E},\mathcal{A}}^{\mathsf{IND-CPA}}(\lambda)$ is defined in Figure 14. We define the advantage on the IND-CPA security as $\epsilon^{\mathsf{IND-CPA}}(\lambda) = \max_{\mathcal{A}\in\mathsf{poly}(\lambda)} \epsilon_{\mathcal{A}}^{\mathsf{IND-CPA}}(\lambda)$.

Definition 8 (Hiding and binding). *A commitment scheme C=($C.KGen$, $C.Com$, $C.Ope$) with security parameter λ is said to be* hiding *if for any two-party polynomial-time adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, the following is negligible:*

$$\epsilon_{\mathcal{A}}^{\mathsf{hiding}}(\lambda) = \Big|\Pr\Big[\mathbf{Exp}_{1,\mathsf{C},\mathcal{A}}^{\mathsf{hiding}}(\lambda) = 1\Big] - \Pr\Big[\mathbf{Exp}_{0,\mathsf{C},\mathcal{A}}^{\mathsf{hiding}}(\lambda) = 1\Big]\Big|$$

where $\mathbf{Exp}_{b,\mathsf{C},\mathcal{A}}^{\mathsf{hiding}}(\lambda)$ is defined in Figure 14. We define the advantage on hiding as $\epsilon^{hiding}(\lambda) = \max_{\mathcal{A}\in\mathsf{poly}(\lambda)} \epsilon_{\mathcal{A}}^{hiding}(\lambda)$.

Commitment scheme C is said to be binding *if for any polynomial-time adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins the binding experiment in Figure 14 is negligible.*

Definition 9 (EUF-CMA). *A signature scheme S = ($\mathsf{S.KGen}$, $\mathsf{S.Sign}$, $\mathsf{S.Ver}$) with security parameter λ is said to satisfy existential unforgeability against a chosen message attack (EUF-CMA) if for any polynomial-time adversary $\mathcal{A}$, the following is negligible:*

$$\epsilon_{\mathcal{A}}^{\mathsf{EUF-CMA}}(\lambda) = \Pr\Big[\mathbf{Exp}_{\mathsf{S},\mathcal{A}}^{\mathsf{EUF-CMA}}(\lambda) = 1\Big]$$

where $\mathbf{Exp}_{\mathsf{S},\mathcal{A}}^{\mathsf{EUF-CMA}}(\lambda)$ is defined in Figure 14. We define the advantage on EUF-CMA as $\epsilon^{EUF\text{-}CMA}(\lambda) = \max_{\mathcal{A}\in\mathsf{poly}(\lambda)} \epsilon_{\mathcal{A}}^{EUF\text{-}CMA}(\lambda)$.

Definition 10 (Non-Interactive Proofs (NIP) [24]). *Let $\mathcal{R}$ be a binary relation and let $\mathcal{L}$ be a language such that $s \in \mathcal{L} \Leftrightarrow (\exists w, (s, w) \in \mathcal{R})$. A* non-interactive proof *(NIP) for $\mathcal{L}$ is a couple of algorithms* (ZK, $\mathsf{ZK.Verify}$) *such that:*

$\mathsf{ZK}\{w : (s, w) \in \mathcal{R}\}$ *outputs a proof π; and*
$\mathsf{ZK.Verify}(s, \pi)$ *outputs a bit b.*

A non-interactive zero knowledge proof *(NIZKP) is an NIP with the following properties:*

Correctness. For any $s, w, \mathcal{R}$ s.t. $(s, w) \in \mathcal{R}$ and $\pi \leftarrow \mathsf{ZK}\{w : (s, w) \in \mathcal{R}\}$, $\mathsf{ZK.Verify}(s, \pi)$ returns 1. That is,

$$\mathsf{Pr}\big[\pi \leftarrow \mathsf{ZK}\{w : (s, w) \in \mathcal{R} : \mathsf{ZK.Verify}(s, \pi) = 1\big] \geq 1 - \epsilon_\lambda.$$

Extractability. There exists a PPT knowledge extractor Ext and a negligible function ϵ_{Ext} such that for any algorithm $\mathcal{A}^{Sim(\cdot)}(\lambda)$ having access to a simulator that forges proofs for chosen statement and that outputs a fresh pair (s, π) with $\mathsf{ZK.Verify}(s, \pi) = 1$, the extractor $Ext^{\mathcal{A}}(\lambda)$ outputs w such that $(s, w) \in \mathcal{R}$ having access to $\mathcal{A}(\lambda)$ with probability at least $1 - \epsilon(\lambda)$.

$$\mathsf{Pr}\big[(s, \pi) \leftarrow \mathcal{A}^{Sim(\cdot)}(\lambda);\ w \leftarrow Ext^{\mathcal{A}}(\lambda) : \mathsf{ZK.Verify}(s, \pi) = 1\ \wedge\ (s, w) \in \mathcal{R}\big] \geq 1 - \epsilon_\lambda.$$

```
$\mathbf{Exp}^{\mathsf{IND\text{-}CPA}}_{b,\mathsf{E},\mathcal{A}}(\lambda)$
$(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{E.KGen}(1^\lambda)$
$(m_0, m_1) \leftarrow \mathcal{A}_1(\mathsf{pk})$
$c_b \leftarrow \mathsf{E.Enc}(\mathsf{pk}, m_b)$
$b_* \leftarrow \mathcal{A}_2(\mathsf{pk}, c_b)$
return $b_*$
```

```
$\mathbf{Exp}^{\mathsf{hiding}}_{b,\mathsf{C},\mathcal{A}}(\lambda)$
$\mathsf{sk} \leftarrow \mathsf{C.KGen}(1^\lambda)$
$(m_0, m_1) \leftarrow \mathcal{A}_1()$
$c_b \leftarrow \mathsf{C.Com}(\mathsf{sk}, m_b)$
$b_* \leftarrow \mathcal{A}_2(c_b)$
return $b_*$
```

```
$\mathbf{Exp}^{\mathsf{binding}}_{\mathsf{C},\mathcal{A}}(\lambda)$
$(c, m_0, \mathsf{sk}_0, m_1, \mathsf{sk}_1) \leftarrow \mathcal{A}()$
return $(m_0 \neq m_1) \wedge \mathsf{C.Ope}(\mathsf{sk}_0, m_0, c) \wedge \mathsf{C.Ope}(\mathsf{sk}_1, m_1, c)$
```

```
$\mathbf{Exp}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{S},\mathcal{A}}(\lambda)$
$\mathsf{Qsign} \leftarrow \emptyset$
$\mathsf{Qcorrupt} \leftarrow \emptyset$
for $j \in [N]$ : $(\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{S.KGen}(1^\lambda)$
$(\mathsf{pk}_*, m_*, \sigma_*) \leftarrow \mathcal{A}^{\mathsf{OSign},\mathsf{OCorrupt}}(\{\mathsf{pk}_j\}_{j\in[N]})$
return $\mathsf{S.Verify}(\mathsf{pk}_*, m_*, \sigma_*) \wedge (\mathsf{pk}_*, m_*) \notin \mathsf{Qsign} \wedge \mathsf{pk}_* \notin \mathsf{Qcorrupt}$
```

```
$\mathsf{OSign}(\mathsf{pk}, m)$
$\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, m)$
$\mathsf{Qsign} \leftarrow \mathsf{Qsign} \cup \{(\mathsf{pk}, m)\}$
return $\sigma$
```

```
$\mathsf{OCorrupt}(\mathsf{pk})$
$\mathsf{Qcorrupt} \leftarrow \mathsf{Qcorrupt} \cup \{\mathsf{pk}\}$
return $\mathsf{sk}$
```

Figure 14: Security experiments for the encryption scheme E and commitment scheme C.

Zero-knowledge. A proof π leaks no information*, i.e.,, there exists a polynomial time algorithm Sim (called the* simulator*) such that* $\mathsf{ZK}\{w : (s,w) \in \mathcal{R}\}$ *and* $\mathsf{Sim}(s)$ *follow the same probability distribution.*

$$\left|\Pr\left[\mathcal{A}^{\mathsf{ZK}(\cdot)}() = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Sim}(\cdot)}() = 1\right]\right| \leq \epsilon_\lambda$$

where $\mathsf{Sim}(s)$ returns a forged signature and $\mathsf{ZK}(s)$ returns a genuine signature.

A signature of knowledge *is defined in the same way, except that a message is embedded in the proof and used for verification.*

C. Σ-Protocols for our Construction

Schnorr Protocol: We first recall the Schnorr protocol [14], an interactive proof whereby a prover proves to a verifier knowledge of a discrete logarithm of an element y in base g in a group $\mathbb{G}$ of prime order p. This protocol is given in Figure 15.

Prover		**Verifier**
x		(g, y)
$r \xleftarrow{\$} \mathbb{Z}_q^*$		
$R = g^r$	$\xrightarrow{R}$	$c \xleftarrow{\$} \mathbb{Z}_q^*$
$\alpha = r + x \cdot c$	$\xleftarrow{c}$	
	$\xrightarrow{\alpha}$	If $g^\alpha = R \cdot y^c$ then `accept`, else `reject`

Figure 15: Discrete logarithms knowledge proof

Discrete Logarithm Equality: We also utilise a proof that elements have the same discrete logarithm, i.e. for n pairs of elements $(g_i, y_i)_{i\in[n]}$ of a group $\mathbb{G}$ of prime order p, there exists a (secret) x such that $g_i^x = y_i$ for all i. We recall such an (interactive) proof in Figure 16 [21].

Prover		**Verifier**
x		$(g_i, y_i)_{i\in[\![n]\!]}$
$r \xleftarrow{\$} \mathbb{Z}_q^*$		
$\forall i \in [\![n]\!], R_i = g_i^r$	$\xrightarrow{(R_i)_{i\in[\![n]\!]}}$	$c \xleftarrow{\$} \mathbb{Z}_q^*$
$\alpha = r + x \cdot c$	$\xleftarrow{c}$	
	$\xrightarrow{\alpha}$	If $\forall i \in [\![n]\!]$: $g_i^\alpha = R_i \cdot y_i^c$ then `accept` else `reject`

Figure 16: Discrete logarithms equality proof

Discrete Logarithm Inequality: A discrete logarithm inequality proof shows that two elements have different discrete logarithms, i.e. for elements (g_1, y_1, g_2, y_2), there exists a (secret) x such that $g_1^x = y_1$ and $g_2^x \neq y_2$. Such an (interactive) proof is given in [22], and is built from a proof that, for $(g_1, h_1, y_1, g_2, h_2, y_2)$, there exists two secrets x and w such that $g_1^x \cdot h_1^w = y_1$ and $g_2^x \cdot h_2^w = y_2$. This proof is instantiated with the protocol given in Figure 17 where $n = 2$.

Then, the proof of knowledge of x such that $g_1^x = y_1$ and $g_2^x \neq y_2$ given in [21] is formulated in Figure 18.

We also need to prove that, where two elements y_1 and y_2 have the same discrete logarithm in bases g_1 and g_2, a third element y_3 has a different discrete logarithm in base g_3, i.e. we know an x such that $g_1^x = y_1$ and $g_2^x = y_2$ but $g_3^x \neq y_3^x$. This proof, given in Figure 19, follows naturally from the one given in Figure 18 using the protocol given in Figure 17 with $n = 3$.

The proofs in Figures 17–19 are Schnorr-like Σ-protocols [25], which means that they follow the structure given in Figure 20, where $(x, y) \in \mathcal{R}$ for a

Prover		**Verifier**
(w, x)		$(g_i, h_i, y_i)_{i \in [\![n]\!]}$
$r, s \xleftarrow{\$} \mathbb{Z}_q^*$		
$\forall i \in [\![n]\!]:$		
$R_i = g_i^r$		
$S_i = h_i^s$	$\xrightarrow{(R_i,S_i)_{i\in[\![n]\!]}}$	$c \xleftarrow{\$} \mathbb{Z}_q^*$
$\alpha = r + w \cdot c$		
$\beta = s + x \cdot c$	$\xleftarrow{c}$	
	$\xrightarrow{\alpha,\beta}$	If $\forall i \in [\![n]\!]$, $g_i^\alpha \cdot h_i^\beta = R_i \cdot S_i \cdot y_i^c$ then `accept` else `reject`

Figure 17: Building block for discrete logarithms inequality proof

Prover		**Verifier**
x		(g_1, y_1, g_2, y_2)
$r \xleftarrow{\$} \mathbb{Z}_q^*$		
$w' = x \cdot r$		
$x' = r$		
$g'_1 = g_1$; $g'_2 = g_2$		$g'_1 = g_1$; $g'_2 = g_2$
$h'_1 = \frac{1}{y_1}$; $h'_2 = \frac{1}{y_2}$		$h'_1 = \frac{1}{y_1}$; $h'_2 = \frac{1}{y_2}$
$y'_1 = 1$		$y'_1 = 1$
$y'_2 = (g'_2)^{w'} \cdot (h'_2)^{x'}$	$\xrightarrow{y'_2}$	
	The prover proves that they know (w', x') such that $y'_1 = (g'_1)^{w'} \cdot (h'_1)^{x'}$ and $y'_2 = (g'_2)^{w'} \cdot (h'_2)^{x'}$	
		If the verifier accepts this proof and $y'_2 \neq 1$, then `accept` else `reject`

Figure 18: Discrete logarithms inequality proof

given relation $\mathcal{R}$, and where Com, Resp, and Ver are polynomial-time algorithms in $|p|$. Note that a Schnorr-like Σ-protocol can be turned into a non-interactive proof by using a hash function $H : \{0,1\}^* \to \mathbb{Z}_q^*$ modelled by a random oracle [26]. To remove the interaction, the prover can simply choose the challenge $c = H(y, R)$. Furthermore, by adding a message to the hash values, we obtain a signature of knowledge. This signature is EUF-CMA as long as it is extractable and it is computationally difficult to find x from a y such that $(x, y) \in \mathcal{R}$. As a reminder, in this article, all signatures are instantiated by the Schnorr signature, which is a signature of knowledge.

AND Proofs: To prove knowledge of the secret (x_1, x_2) such that $(y_1, x_1) \in \mathcal{R}_1 \wedge (y_2, x_2) \in \mathcal{R}_2$ for some instances and relations y_1, $\mathcal{R}_1$, y_2, and $\mathcal{R}_2$, it is sufficient to prove $(y_1, x_1) \in \mathcal{R}_1$ and $(y_2, x_2) \in \mathcal{R}_2$ using two independent proofs. Recall that the structure of a Schnorr proof simulator is always the same: the simulator $\mathsf{Sim}(y)$ picks at random c and α, builds R

Prover		**Verifier**
x		$(g_1, y_1, g_2, y_2, g_3, y_3)$
$r \xleftarrow{\$} \mathbb{Z}_q^*$		
$w' = x \cdot r$		
$x' = r$		
$\forall i \in [\![3]\!]:$		$\forall i \in [\![3]\!]:$
$g'_i = g_i$		$g'_i = g_i$
$h'_i = \frac{1}{y_i}$		$h'_i = \frac{1}{y_i}$
$y'_1 = 1$; $y'_2 = 1$		$y'_1 = 1$; $y'_2 = 1$
$y'_3 = (g'_3)^{w'} \cdot (h'_3)^{x'}$	$\xrightarrow{y'_3}$	
	The prover proves that they know (w', x') such that $y'_1 = (g'_1)^{w'} \cdot (h'_1)^{x'}$ and $y'_2 = (g'_2)^{w'} \cdot (h'_2)^{x'}$ and $y'_3 = (g'_3)^{w'} \cdot (h'_3)^{x'}$	
		If the verifier accepts this proof and $y'_3 \neq 1$, then `accept` else `reject`

Figure 19: Discrete logarithms inequality proof for 3 elements

Prover		**Verifier**
x		y
Picks r at random		
$R = \mathsf{Com}(y, r)$	$\xrightarrow{R}$	$c \xleftarrow{\$} \mathbb{Z}_q^*$
$\alpha = \mathsf{Resp}(y, r, c)$	$\xleftarrow{c}$	
	$\xrightarrow{\alpha}$	Return $\mathsf{Ver}(y, R, c, \alpha)$

Figure 20: Schnorr-like Σ-protocol

from (y, c, α), and returns (R, c, α). Thus, to prove several independent relations at the same time, the same challenge can be used without compromising the zero-knowledge property. The protocol in Figure 21 proves that $\bigwedge_{i=1}^n (y_i, x_i) \in \mathcal{R}_i$ for n statements y_i of n relations $\mathcal{R}_i$ knowing n secret witnesses x_i, using the Schnorr-like Σ-protocols $(\mathsf{Com}_i, \mathsf{Resp}_i, \mathsf{Ver}_i)$. Note that this is a Schnorr-like Σ-protocol.

Prover		**Verifier**
$(x_i)_{i \in [\![n]\!]}$		$(y_i)_{i \in [\![n]\!]}$
Picks $(r_i)_{i \in [\![n]\!]}$ at random		
$\forall i \in [\![n]\!], R_i = \mathsf{Com}_i(y_i, r_i)$	$\xrightarrow{(R_i)_{i\in[\![n]\!]}}$	$c \xleftarrow{\$} \mathbb{Z}_q^*$
$\forall i \in [\![n]\!], \alpha_i = \mathsf{Resp}_i(y_i, r_i, c)$	$\xleftarrow{c}$	
	$\xrightarrow{(\alpha_i)_{i\in[\![n]\!]}}$	Return : $\bigwedge_{i=1}^n \mathsf{Ver}_i(y_i, R_i, c, \alpha_i)$

Figure 21: AND-proof for the relation $\bigwedge_{i=1}^n (y_i, x_i) \in \mathcal{R}_i$.

Proof of Partial Knowledge: In [20], from n Σ-protocols $(\mathsf{Com}_i, \mathsf{Resp}_i, \mathsf{Ver}_i)$ associated with n relations $\mathcal{R}_i$, Cramer *et. al.* show how to construct a Σ-protocol that allows a prover to prove knowledge of t secrets x_i and t instances y_i among n (without revealing which ones) such that for each of these secrets, $(y_i, x_i) \in \mathcal{R}_i$. Their construction uses a threshold secret sharing, which we will instantiate using Shamir's secret sharing [27]. The latter uses Lagrange interpolation in $\mathbb{Z}_q^*$:

$$L_{(a_i,b_i)_{i\in[\![n]\!]}}(x) = \sum_{j=1}^{n} b_j \prod_{i=1;i\neq j}^{n} \frac{x-a_i}{a_j-a_i}.$$

This construction is shown in Figure 22, and is a Schnorr-like Σ-protocol.

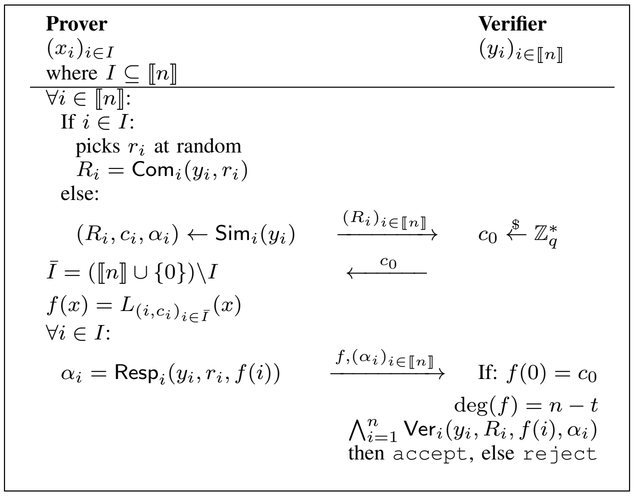
Prover
$(x_i)_{i\in I}$
where $I \subseteq [\![n]\!]$
Verifier
$(y_i)_{i\in[\![n]\!]}$
$\forall i \in [\![n]\!]$:
If $i \in I$:
picks r_i at random
$R_i = \mathsf{Com}_i(y_i, r_i)$
else:
$(R_i, c_i, \alpha_i) \leftarrow \mathsf{Sim}_i(y_i)$
$\xrightarrow{(R_i)_{i\in[\![n]\!]}}$
$c_0 \overset{\$}{\leftarrow} \mathbb{Z}_q^*$
$\xleftarrow{c_0}$
$\bar{I} = ([\![n]\!] \cup \{0\}) \backslash I$
$f(x) = L_{(i,c_i)_{i\in\bar{I}}}(x)$
$\forall i \in I$:
$\alpha_i = \mathsf{Resp}_i(y_i, r_i, f(i))$
$\xrightarrow{f,(\alpha_i)_{i\in[\![n]\!]}}$
If: $f(0) = c_0$
$\deg(f) = n - t$
$\bigwedge_{i=1}^{n} \mathsf{Ver}_i(y_i, R_i, f(i), \alpha_i)$
then accept, else reject

Figure 22: Proof of partial knowledge.

In the particular case where we prove knowledge of only one secret among n, a threshold-free secret sharing scheme is sufficient, so we can use multiplication in $\mathbb{Z}_q^*$ to share the challenge c. The resulting Schnorr-like Σ-protocol, given in Figure 23, allows a prover to prove knowledge of one x_j such that $\bigvee_{i=1}^{n}(y_i, x_j) \in \mathcal{R}_i$ more efficiently.

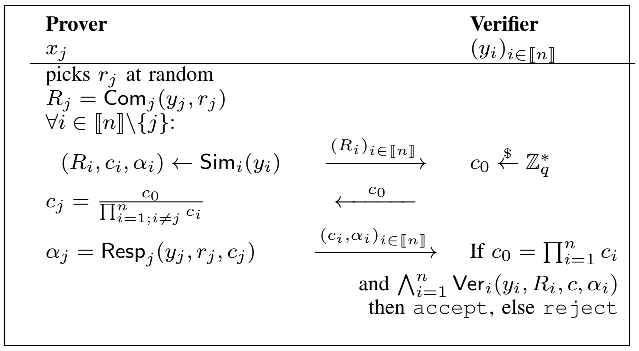
Prover
x_j
Verifier
$(y_i)_{i\in[\![n]\!]}$
picks r_j at random
$R_j = \mathsf{Com}_j(y_j, r_j)$
$\forall i \in [\![n]\!] \backslash \{j\}$:
$(R_i, c_i, \alpha_i) \leftarrow \mathsf{Sim}_i(y_i)$
$\xrightarrow{(R_i)_{i\in[\![n]\!]}}$
$c_0 \overset{\$}{\leftarrow} \mathbb{Z}_q^*$
$c_j = \frac{c_0}{\prod_{i=1;i\neq j}^{n} c_i}$
$\xleftarrow{c_0}$
$\alpha_j = \mathsf{Resp}_j(y_j, r_j, c_j)$
$\xrightarrow{(c_i,\alpha_i)_{i\in[\![n]\!]}}$
If $c_0 = \prod_{i=1}^{n} c_i$
and $\bigwedge_{i=1}^{n} \mathsf{Ver}_i(y_i, R_i, c, \alpha_i)$
then accept, else reject

Figure 23: OR-proof for the relation $\bigvee_{i=1}^{n}(y_i, x_j) \in \mathcal{R}_i$.

Σ-Protocol Complexity: We will now recall the complexity in time of the non-interactive proofs. For basic proofs (Figure 15, 16, 17 and 18), the time is evaluated in the number of exponentiations in the group. Note that the proof in Fig. 22 also requires $O(n^2)$ multiplications on the prover side. For constructions using other generic proofs (Figure 20, 21, 22 and 23), the time is evaluated in number of executions of the least efficient generic proof. Our complexity analysis is summarised in Figure III.

proof	Prover time	Verifier time
Fig. 15	$O(1)$	$O(1)$
Fig. 16	$O(n)$	$O(n)$
Fig. 17	$O(n)$	$O(n)$
Fig. 18	$O(1)$	$O(1)$
Fig. 19	$O(1)$	$O(1)$
Fig. 20	$O(1)$	$O(1)$
Fig. 21	$O(n)$	$O(n)$
Fig. 22	$O(n)$	$O(n)$
Fig. 23	$O(n)$	$O(n)$

Table III: Complexity in time of our Σ-protocols.

Appendix C
Privacy Proofs

We prove the following three properties:

1) *Privacy in the presence of honest-but-curious attackers excluding the chair* (Appendix C-A);
2) *Privacy in the presence of dishonest attackers including the chair* (Appendix C-B), which implies both privacy in the presence of honest-but-curious attackers including the chair, and anonymity in the presence of dishonest attackers including the chair;
3) *Anonymity in the presence of dishonest attackers excluding the chair* (Appendix C-C);

The first and the second properties imply the privacy in the presence of honest-but-curious attackers, which proves Theorem 1. The second and third properties imply the anonymity in the presence of dishonest attackers, which proves Theorem 2. Finally, the second property proves Proposition 1.

A. Privacy in the presence of honest-but-curious attackers excluding the chair.

We define the simulator Sim as follows. $\mathsf{Sim}((\mathsf{pkr}_i, \rho_i)_{i\in[M]}, \mathsf{pkc}, \mathsf{dalgo}, \mathsf{cleak}, (\mathsf{leak}_i)_{i\in\mathsf{AR}})$: Sim acts as the real protocol except for the following cases:

- Sim simulates each proof/signature of knowledge generated by the honest parties.
- Sim runs the extraction algorithms $\mathcal{E}$ on the proofs ρ_i for all $i \in [M]$. We set skr_i as the extracted values. If one extraction fails, Sim aborts.
- Each time that Sim verifies a proof or a signature of knowledge generated by one of the dishonest parties

(i.e., the parties A_i for $i \in \mathsf{AA}$ and R_i for $i \in \mathsf{AR}$), Sim runs the extraction algorithm on this proof and verifies that the statement is indeed related to the extracted secret. If the extraction or verification of the relation fails, Sim aborts.

- When Sim simulates the submission for an honest author A_i (i.e., for $i \in [N] \backslash \mathsf{AA}$):
  - it generates $p_{i,1}$ by building a void commitment $p_{i,1} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,3}, \perp)$;
  - if $\mathtt{RC}'_i = \mathsf{AA}$, then it generates also $p_{i,2}$ by building a void commitment $p_{i,2} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,4}, \mathtt{content}_i)$;
  - it generates the elements x_j for $j \in [N]$ as follows: if $j \in \mathtt{RC}'_i$, then $x_j \leftarrow \mathsf{pka}_{i,1}^{\mathsf{skr}_j}$, else $x_j \xleftarrow{\$} \mathbb{G}$;
  - it generates $p_{i,5}$ by computing $p_{i,5} \leftarrow H(\mathsf{pkc}^{r_i})$ (i.e. $(p_{i,4}, p_{i,5})$ is the hash-ElGamal encryption of the null message).
- Sim skips the distribution for the reviewers R_j such that $j \notin \mathsf{AR}$.
- When Sim simulates the distribution for a dishonest review R_j (where $j \in \mathsf{AR}$) for a paper p_i:
  - if $i \in \mathsf{AA}$ it decrypts $(p_{i,4}, p_{i,5})$ by computing $\mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i \leftarrow H(p_{i,5}^{r_i}) \oplus p_{i,5}$ (where r_i is the witness extracted from $p_{i,6}$)
  - else it does not decrypt $(p_{i,4}, p_{i,5})$ and uses the same values as the ones used in the simulation of the i^{th} paper submission for $\mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i$.
- For each $j \in [N]$, Sim simulates the bidding phase as follows. First, the simulator initialises a vector $\mathsf{correspondingBid}$ of the same size as $\mathsf{bidHistory}$, whose coefficients will be assigned later. It then proceeds as in the real protocol for the dishonest reviewers R_i where $i \in \mathsf{AR}$ to produce the bid b with the mark $m \leftarrow \mathsf{GenMark}_i(j, v_{i,j})$. It then associates this bid b with the tuple (j, m) contained in the vector $\mathsf{bidHistory}$ at an index for which the corresponding bit in $\mathsf{reviewerBid}_i$ is 1, and adds (j, m, b) to the vector $\mathsf{correspondingBid}$ at this index. For each element (j, m) remaining in $\mathsf{bidHistory}$, Sim generates b by picking $h, \mathsf{pk}, \gamma \xleftarrow{\$} \mathbb{G}$ at random and by simulating the proofs and the signature of knowledge. Then, it adds (j, m, b) at the corresponding index to $\mathsf{correspondingBid}$. For each j, the simulator Sim builds BP_j as the multiset of the (m, b) such that $(j, m, b) \in \mathsf{correspondingBid}$.
- For each pair $(x, \mathtt{response})$, the simulator finds the tuple (j, m, b) at the x^{th} position of $\mathsf{correspondingBid}$, and simulates the assignment as in the real protocol except that it simulates the signature of knowledge σ on the statement (g, pkc) and the message b in the assignment a. It simulates the acceptation/rejection of the assignment as follows:
  - If there exists an index $i \in \mathsf{AR}$ such that the x^{th} element of $\mathsf{reviewerBid}_i$ is 1, then it proceeds as R_i in the real protocol.
  - Else, to simulate the generation of the acceptance $(\mathtt{response}, \sigma, \phi)$, it simulates the signature of knowledge σ. If $\mathtt{response} = \mathtt{reject}$, it simulates the proof of knowledge ϕ, else it sets $\phi \leftarrow \perp$.
- The simulator does not generate reviews for the honest reviewers, and skips the Decision phase.

Game G_0: In this game, a challenger $\mathcal{C}$ simulates the case $b = 0$.

Game G_1: Same as G_0 except that $\mathcal{C}$ runs the protocol twice:

$$((\mathsf{view}_{A_i})_{i\in[N]}, (\mathsf{view}_{R_i})_{i\in[M]}, \mathsf{view}_C, (\mathsf{leak}_i)_{i\in[M]}) \leftarrow \mathrm{BlindReview}\langle (A_i)_{i\in[N]}, (R_i)_{i\in[M]}, C\rangle$$

$$((\mathsf{view}'_{A_i})_{i\in[N]}, (\mathsf{view}'_{R_i})_{i\in[M]}, \mathsf{view}'_C, (\mathsf{leak}'_i)_{i\in[M]}) \leftarrow \mathrm{BlindReview}\langle (A_i)_{i\in[N]}, (R_i)_{i\in[M]}, C\rangle.$$

Then $\mathcal{C}$ aborts if $(\mathsf{leak}_i)_{i\in\mathsf{AR}} \neq (\mathsf{leak}'_i)_{i\in\mathsf{AR}}$. Since all parties follow the protocol honestly, the content of the papers, the marks given, and the answers to the assignments are the same from one execution of the protocol to another. We will therefore have:

$$|\Pr[\mathcal{D} \text{ returns 1 on } G_0] - \Pr[\mathcal{D} \text{ returns 1 on } G_1]| = 0.$$

Game G_2: Same as G_1 except that $\mathcal{C}$ runs the extraction algorithm on the proofs ρ_i for all $i \in \mathsf{AR}$. We set skr_i as the extracted values. If one extraction fails, or if $\mathsf{pkr}_i \neq g^{\mathsf{skr}_i}$ for one i, $\mathcal{C}$ aborts the game. Moreover, for all $i \in [N]$ and $j \in [M]$, $\mathcal{C}$ replaces each computation of $\mathsf{pkr}_j^{\mathsf{ska}_{i,1}}$ performed by a honest party by the computation of $\mathsf{pka}_{i,1}^{\mathsf{skr}_j}$. We have:

$$|\Pr[\mathcal{D} \text{ returns 1 on } G_1] - \Pr[\mathcal{D} \text{ returns 1 on } G_2]| \leq M\epsilon^{\mathsf{Ext}}(\lambda),$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

Game G_3: Same as G_2 except that $\mathcal{C}$ runs the extraction algorithm on the proofs/signatures of knowledge generated by the dishonest parties. If one extraction fails, or returns a witness which is not related to the corresponding statement, $\mathcal{C}$ aborts the game. There is at most $2N + MN(M+4) + M < 3MN(M+4)$ proofs/signatures of knowledge generated by the dishonest parties during the protocol, we deduce that:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_2] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_3]| \leq 3MN(M+4)\epsilon^{\mathsf{Ext}}(\lambda),$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

Game G_4: Same as G_3 except that $\mathcal{C}$ simulates each proof/signature of knowledge generated by the honest parties. Since all the proofs/signatures of knowledge we used are perfectly zero-knowledge, we have:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_3] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_4]| = 0.$$

Game G_5: Same as G_4 except that each time that $\mathcal{C}$ simulates the submission of a paper by an honest author, it generates p_1 by committing $\perp$ instead of the authors' description: $p_1 \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_3, \perp)$. We claim that:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_4] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_5]| \leq N\epsilon^{\mathsf{Hiding}}(\lambda),$$

where $\epsilon^{\mathsf{Hiding}}(\lambda)$ is the advantage on the Hiding security of the commitment scheme. We prove this claim by an hybrid argument. We set $G_{5,0} = G_4$, and we define the game $G_{5,i}$ as follows for $0 < i \leq N$:

Game $G_{5,i}$: Same as $G_{5,i-1}$ except that when $\mathcal{C}$ simulates the submission of the paper p_i by the author A_i, if $i \notin \mathsf{AA}$, it generates p_1 by committing $\perp$ instead of the authors' description: $p_{i,1} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,3}, \perp)$. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i}]\Big| \leq \epsilon^{\mathsf{Hiding}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AA}$. If $i \notin \mathsf{AA}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i}]\Big| > \epsilon^{\mathsf{Hiding}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon_{\mathcal{B}}^{\mathsf{Hiding}}(\lambda) > \epsilon^{\mathsf{Hiding}}(\lambda)$.

$\mathcal{B}$ simulates the game $G_{5,i-1}$ to $\mathcal{D}$, except that when it simulates the submission of the paper p_i, it does not choose the key $\mathsf{ska}_{i,3}$, sets $m_0 = \mathtt{alist}_i$ and $m_1 = \perp$, sends (m_0, m_1) to its challenger, receives the commitment c, and sets $p_{i,1} = c$. Note that this commitment is never opened during the experiment. At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{hiding}}_{0,\mathsf{C},\mathcal{D}}(\lambda)$, then $p_{i,1}$ is the commitment of the author list as in the game $G_{5,i-1}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{hiding}}_{0,\mathsf{C},\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{hiding}}_{1,\mathsf{C},\mathcal{D}}(\lambda)$, then $p_{i,1}$ is the commitment of $\perp$ as in the game $G_{5,i}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{hiding}}_{1,\mathsf{C},\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i}]$.

The proof of the claim follows.

Game G_6: Same as G_5 except that each time that $\mathcal{C}$ simulates the submission of a paper by an honest author A_i, if $\mathtt{RC}'_i = \mathsf{AA}$, then it sets $p_{i,2} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,4}, \perp)$. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_5] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_6]\Big| \leq N\epsilon^{\mathsf{Hiding}}(\lambda)$$

where $\epsilon_{\mathsf{Hiding}}(\lambda)$ is the advantage on the Hiding security of the commitment scheme. This claim can be proven by using a similar hybrid argument than for the previous game hop.

Game G_7: Same as G_6 except that at the beginning of the game, $\mathcal{C}$ instantiates a dictionary D, then each time $\mathcal{C}$ encrypts a message m using pkc in a ciphertext $c = (p_{i,4}, p_{i,5})$ using hash-Elgamal during the simulation of the submission for an author A_i such that $i \notin \mathsf{AA}$, it sets $\mathsf{D}[c] \leftarrow m$. Moreover, each time it has to decrypt a ciphertext $c = (p_{i,4}, p_{i,5})$ during the distribution algorithm:

- if $i \notin \mathsf{AA}$, it does not decrypt c and uses $\mathsf{D}[c]$ as the decryption of c instead.
- if $i \in \mathsf{AA}$, it uses the witness r_i extracted from $p_{i,6}$ and decrypts $c = (p_{i,4}, p_{i,5})$ by computing $H(\mathsf{pkc}^{r_i}) \oplus p_{i,5}$.

Since hash-ElGamal is correct and all the extractions returned the correct witnesses, this does not change the distribution of views for $\mathcal{D}$, so:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_6] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_7]| = 0.$$

Game G_8: Same as G_7 except that each time that $\mathcal{C}$ simulates the submission of a paper by an honest author, it generates p_5 by computing $H(\mathsf{pkr}^r)$ (i.e., (p_4, p_5) is a hash ElGamal ciphertext for the binary message $0^{|H(\mathsf{pkr}^{r_i})|}$) and sets $\mathsf{D}[(p_4, p_5)] \leftarrow \mathsf{ska}_1 || \mathsf{ska}_4 || \mathtt{content}$. We claim that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_8] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_7]\Big| \leq N\epsilon^{\mathsf{IND-CPA}}(\lambda),$$

where $\epsilon^{\mathsf{IND-CPA}}(\lambda)$ is the advantage on the $\mathsf{IND-CPA}$ security of hash ElGamal. We prove this claim by an hybrid argument. We set $G_{8,0} = G_7$, and we define the game $G_{8,i}$ as follows for $0 < i \leq N$.

Game $G_{8,i}$**:** Same as $G_{8,i-1}$ except that when $\mathcal{C}$ simulates the submission of the paper p_i by the author A_i, if $i \notin \mathsf{AA}$, it generates $p_{i,5}$ by computing $H(\mathsf{pkr}^{r_i})$ and sets $\mathsf{D}[(p_{i,4}, p_{i,5})] \leftarrow \mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i$. We claim that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i}]\Big| \leq \epsilon^{\mathsf{IND-CPA}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AA}$. If $i \notin \mathsf{AA}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i}]\Big| > \epsilon^{\mathsf{IND-CPA}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon_{\mathcal{B}}^{\mathsf{IND-CPA}}(\lambda) > \epsilon^{\mathsf{IND-CPA}}(\lambda)$.

$\mathcal{B}$ receives a public key pk and set $\mathsf{pkc} \leftarrow \mathsf{pk}$. It simulates the game $G_{8,i-1}$ to $\mathcal{D}$, except that when it simulates the submission of the paper p_i, it sets $m_0 \leftarrow \mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i$ and $m_1 \leftarrow 0^{|m_0|}$, sends (m_0, m_1) to its challenger, receives the ciphertext c, sets $(p_{i,4}, p_{i,5}) \leftarrow c$ and sets $\mathsf{D}[c] \leftarrow m_0$. Note that $\mathcal{B}$ never decrypts a ciphertext using the secret key skc in the $G_{8,i-1}$ game. At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}_{0,\mathsf{C},\mathcal{D}}^{\mathsf{IND-CPA}}(\lambda)$, then $(p_{i,4}, p_{i,5})$ is the encryption of $\mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i$ as in the game $G_{8,i-1}$, so $\Pr\Big[\mathbf{Exp}_{0,\mathsf{C},\mathcal{D}}^{\mathsf{IND-CPA}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}_{1,\mathsf{C},\mathcal{D}}^{\mathsf{IND-CPA}}(\lambda)$, then $(p_{i,4}, p_{i,5})$ is the encryption of the zero binary string as in the game $G_{8,i}$, so $\Pr\Big[\mathbf{Exp}_{1,\mathsf{C},\mathcal{D}}^{\mathsf{IND-CPA}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i}]$.

The proof of the claim follows.

Game G_9**:** Same as G_8 except that if there exists $(i, j, j_*) \in [N] \times [M] \times [M]$ such that $j \notin \mathtt{RC}_i$ and such that during the submission of the paper p_i, the chosen element x_j verifies $x_j = g^{\mathsf{ska}_i \mathsf{skr}_{j_*}}$, then the challenger aborts the game. Since x_j is chosen at random in the uniform distribution on $\mathbb{G}$, we deduce that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_9] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_8]\Big| \leq \frac{NM^2}{|\mathbb{G}|}.$$

Game G_{10}**:** Same as G_9 except that if there exists $(j, j') \in [M]^2$ such that $\mathsf{pkr}_j = \mathsf{pkr}_{j'}$, then the challenger aborts the game. We have:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_10] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_9]\Big| \leq \frac{M^2}{|\mathbb{G}|}.$$

Game G_{11}**:** Same as G_{10} except that each times it simulates the bidding algorithm for the reviewer R_j on paper p_i, the challenger replaces the condition "$\mathsf{pka}_{i,1}^{\mathsf{skr}_j} \in p_3$" by "$j \in \mathtt{RC}_i$". It also replaces the condition "$\mathsf{pka}_{i,1}^{\mathsf{skr}_j} \notin p_3$" by "$j \notin \mathtt{RC}_i$" in the distribution algorithm and its verification. According to the previous game, if G_{10} does not aborts, then $\mathsf{pka}_{i,1}^{\mathsf{skr}_j} \in p_3 \Leftrightarrow j \in \mathtt{RC}_i$, so:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{11}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{10}]\Big| = 0.$$

Game G_{12}**:** Same as G_{11} except that each time that $\mathcal{C}$ simulates the submission of a paper by author A_i such that $i \notin \mathsf{AA}$, it generates the set $\{x_j\}_{1 \leq j \leq M}$ as follows: for all $j \in [M]$, if $j \notin \mathtt{RC}'_i$, then picks $x_j \xleftarrow{\$} \mathbb{G}$, else process as in G_{11}. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{11}]\Big| \leq NM\epsilon^{\mathsf{DDH}}(\lambda)$$

where $\epsilon^{\mathsf{DDH}}(\lambda)$ is the advantage on the DDH assumption. We prove this claim by an hybrid argument. We set $G_{12,1,0} = G_8$, and we define the game $G_{12,i,j}$ as follows for $0 < i \leq N$ and $0 < j \leq M$:

Game $G_{12,i,j}$**:** If $i > 1$ and $j = 1$, same then as $G_{12,i-1,M}$, else same of $G_{12,i,j-1}$, except that when $\mathcal{C}$ simulates the generation of the element x_j during the submission of the paper p_i by the author A_i, if $i \notin \mathsf{AA}$ and $j \notin \mathtt{RC}'_i$, then it picks $x_j \xleftarrow{\$} \mathbb{G}$. We claim that, if $i > 1$ and $j = 1$:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,0}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i-1,M}]\Big| \leq \epsilon^{\mathsf{DDH}}(\lambda),$$

else:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,j}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,j-1}]\Big| \leq \epsilon^{\mathsf{DDH}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AA}$ or $j \notin \mathtt{RC}_i$. If $i \notin \mathsf{AA}$ and $j \in \mathtt{RC}_i \backslash \mathtt{RC}'_i$, we prove the claim by reduction.

Case $i = 1$ **or** $j > 1$**:** Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,j-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,j}]\Big| > \epsilon^{\mathsf{DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon_{\mathcal{B}}^{\mathsf{DDH}}(\lambda) > \epsilon^{\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a DDH triplet (X, Y, Z), and sets $\mathsf{pka}_{i,1} \leftarrow X$ and $\mathsf{skr}_j \leftarrow Y$. It simulates the game $G_{12,i,j-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the random oracle H as follows: when it receives a new input input, it picks a random element q, sets $\mathsf{output} \leftarrow g^q$, and returns output. It stores the tuple $(\mathsf{input}, \mathsf{output}, q)$ so that if it is asked to hash input again, it can return output again.
- when $\mathcal{B}$ simulates the submission of a paper p_i, it generates the set $\{x_{j_*}\}_{1 \le j_* \le M}$ as follows: sets $x_j \leftarrow Z$, and for all $j_* \in [M] \backslash \{j\}$:
  - if $j_* < j$, if $j_* \notin \mathtt{RC}'_i$, then picks $x_j \xleftarrow{\$} \mathbb{G}$, else computes $x_{j_*} \leftarrow \mathsf{pka}_i^{\mathsf{skr}_{j_*}}$;
  - if $j_* > j$, if $j_* \notin \mathtt{RC}_i$, then picks $x_j \xleftarrow{\$} \mathbb{G}$, else computes $x_{j_*} \leftarrow \mathsf{pka}_i^{\mathsf{skr}_{j_*}}$.
- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_{i_*}, it generates h by choosing $\theta \xleftarrow{\$} \mathbb{Z}_q^*$ and computing $h \leftarrow g^\theta$, it generates pk by computing $\mathsf{pk} \leftarrow Y^\theta$, and it generates γ by computing $\gamma \leftarrow Y^q$, where q is the element in the tuple $(p_{i_*}, H(p_{i_*}), q)$ stored by the random oracle H.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then the $x_j = Z$ used in the submission of the paper p_i is the DDH of $\mathsf{pka}_{i,1}$ and pkr_j, as in the game $G_{12,i,j-1}$, so $\Pr\left[\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\right] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,j-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then $x_j = Z$ is a random group element, as in the game $G_{12,i,j}$, so $\Pr\left[\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\right] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12,i,j}]$.

The proof of the claim follows for the case $i = 1$ or $j > 1$. The other case can be proven in the same way.

Game G_{13}**:** Same as G_{12} except that each time that $\mathcal{C}$ simulates the bidding on a paper p_i by a reviewer R_j such that $j \notin \mathsf{AR}$, it chooses the element pk at random: $\mathsf{pk} \xleftarrow{\$} \mathbb{G}$. We claim that:

$$\big| \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{12}] \big| \le NM\epsilon^{\mathsf{DDH}}(\lambda).$$

We prove this claim by an hybrid argument. We set $G_{13,1,0} = G_{12}$, and we define the game $G_{13,i,j}$ as follows for $0 < i \le N$ and $0 < j \le M$:

Game $G_{13,i,j}$**:** If $i > 1$ and $j = 1$, same then as $G_{13,i-1,M}$, else same of $G_{13,i,j-1}$, except that when $\mathcal{C}$ simulates the bidding algorithm of the reviewer R_j on the paper p_i, if $j \notin \mathsf{AR}$, then it picks $\mathsf{pk} \xleftarrow{\$} \mathbb{G}$. We claim that, if $i > 1$ and $j = 1$:

$$\big| \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,0}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i-1,M}] \big| \le \epsilon^{\mathsf{DDH}}(\lambda)$$

else:

$$\big| \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j-1}] \big| \le \epsilon^{\mathsf{DDH}}(\lambda).$$

This is clearly verified when $j \in \mathsf{AR}$. If $i \notin \mathsf{AR}$, we prove the claim by reduction.

Case $i = 1$ **or** $j > 1$**:** Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\big| \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j}] \big| > \epsilon^{\mathsf{DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon^{\mathsf{DDH}}_{\mathcal{B}}(\lambda) > \epsilon^{\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a DDH triplet (X, Y, Z), and sets $\mathsf{skr}_j \leftarrow X$. It simulates the game $G_{13,i,j-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the random oracle H as follows: when it receives a new input input, it picks a random element q, sets $\mathsf{output} \leftarrow g^q$, and returns output. It stores the tuple $(\mathsf{input}, \mathsf{output}, q)$ so that if it is asked to hash input again, it can return output again.
- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_i, it sets $h \leftarrow Y$ and $\mathsf{pk} \leftarrow Z$. It generates γ by computing $\gamma \leftarrow X^q$, where q is the element in the tuple $(p_{i_*}, H(p_{i_*}), q)$ stored by the random oracle H.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then $\mathsf{pk} = Z$ is the DDH of skr_j and the h used by R_j to bid on the paper p_i, as in the game $G_{13,i,j-1}$, so $\Pr\left[\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\right] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then Z is a random group element, as in the game $G_{13,i,j}$, so $\Pr\left[\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\right] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j}]$.

The proof of the claim follows for the case $i = 1$ or $j > 1$. The other case can be proven in the same way.

Game G_{14}**:** Same as G_{13} except that each time that $\mathcal{C}$ simulates the bidding on a paper p_i by a reviewer R_j such that $j \notin \mathsf{AR}$, it chooses the element γ at random: $\gamma \xleftarrow{\$} \mathbb{G}$. Let q_H the number of call to the random oracle during the experiment. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13}]\big| \leq M\epsilon^{\mathsf{q_H-DDH}}(\lambda).$$

We prove this claim by an hybrid argument. We set $G_{14,0} = G_{13}$, and we define the game $G_{14,i}$ as follows for $0 < i \leq M$:

Game $G_{14,i}$**:** Same as $G_{14,i-1}$, except that when $\mathcal{C}$ simulates the bidding algorithm of the reviewer R_i on a paper p_j, if $i \notin \mathsf{AR}$, then it picks $\gamma \xleftarrow{\$} \mathbb{G}$. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14,i}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14,i-1}]\big| \leq \epsilon^{q_H-\mathsf{DDH}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AR}$. If $i \notin \mathsf{AR}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14,i}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14,i-1}]\big| > \epsilon^{\mathsf{q_H-DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon^{q_H-\mathsf{DDH}}_{\mathcal{B}}(\lambda) > \epsilon^{q_H-\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a q_H-DDH triplet $(X, (Y_k, Z_k)_{q\in[q_H]}$, and sets $\mathsf{skr}_i \leftarrow X$. It simulates the game $G_{14,i-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the q^{th} query to the random oracle H as follows: when it receives a new input input_q, it sets $\mathsf{output}_q \leftarrow Y_q$, and returns output_q. It stores the tuple $(\mathsf{input}_q, \mathsf{output}_q)$ so that if it is asked to hash input again, it can return output_q again.
- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_i, it asks $H(p_i)$ to the random oracle, then it generates γ by computing $\gamma \leftarrow Z_q$, where q is the index of the couple of input/output stored by the random oracle such that $(\mathsf{input}_q, \mathsf{output}_q) = (p_i, Y_q)$.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then for each bid of R_j, $\gamma = Z_q$ is the DDH of $\mathsf{skr}_j = X$ and $H(p_i) = Y_q$, as in the game $G_{14,i-1}$, so $\Pr\Big[\mathbf{Exp}^{q_H-\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then for each bid of R_j, $\gamma = Z_q$ is a random element, as in the game $G_{14,i}$, so $\Pr\Big[\mathbf{Exp}^{q_H-\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14,i}]$.

The proof of the claim follows.

Game G_{15}**:** Same as G_{14}, except that instead of rerunning the protocol a second time, we run the simulator on the leaks from the first execution of the protocol, as in the case $b = 1$ of the privacy definition. Remember that since G_1, we have ensured that from one execution of the protocol to the next, the bid phase and the assignment phase follow the same steps. Furthermore, apart from the use of leaks, the way we simulate the second iteration of the protocol is the same as for the simulator we defined at the beginning of the proof. It is therefore not possible to distinguish this game from the previous one. We can deduce that:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{15}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{14}]\big| = 0.$$

Conclusion: From this sequence of games, we deduce the advantage $\epsilon(\lambda)$ of a collusion of honest-but-curious reviewers and authors on privacy:

$$\epsilon(\lambda) \leq 4MN(M+4)\epsilon^{\mathsf{Ext}}(\lambda) + 2N\epsilon^{\mathsf{Hiding}}(\lambda) + N\epsilon^{\mathsf{IND-CPA}}_{\mathsf{HElGamal}}(\lambda) + \frac{(N+1)M^2}{|\mathbb{G}|} + (2N + q_H)M\epsilon^{\mathsf{DDH}}(\lambda)$$

B. Privacy in the presence of dishonest attackers including the chair.

$\mathsf{Sim}((\mathsf{pkr}_i, \rho_i)_{i\in[M]}, \mathsf{pkc}, \mathsf{dalgo}, \mathsf{cleak}, (\mathsf{leak}_i)_{i\in\mathsf{AR}})$**:**

Sim simulates correctly the real protocol except for the following cases:

- Simulates each proof/signature of knowledge generated by the honest parties.
- Runs the extraction algorithms on the proofs τ_i for all $i \in [N]$. We set $\mathsf{ska}_{i,1}$ as the extracted values. If one extraction fails, Sim aborts.
- Each time that Sim verifies a proof or a signature of knowledge generated by one of the dishonest parties (i.e., the parties A_i for $i \in \mathsf{AA}$, R_i for $i \in \mathsf{AR}$, and C), Sim runs the extraction algorithm on this proof and verifies that the statement is indeed related to the extracted secret. If the extraction or verification of the relation fails, Sim aborts.
- For all $i \in [N]\backslash\mathsf{AA}$, when it simulates the submission for the honest author i, Sim generates the element $p_{i,1}$ by computing $p_{i,1} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,3}, \perp)$.
- For each $j \in [N]\backslash\mathsf{AR}$, when it simulates the biding on the paper p_i for the honest reviewer R_j, Sim picks pk and γ at random in $\mathbb{Z}_q^*$, and it replaces the condition "$\mathsf{pka}_{i,1}^{\mathsf{skr}_j} \in p_{3,i}$" by the condition "$\mathsf{pkr}_j^{\mathsf{ska}_{i,1}} \in p_{3,i}$".

Game G_0**:** In this game, a challenger $\mathcal{C}$ simulates the case $b = 0$.

Game G_1**:** Identical to G_0, except that $\mathcal{C}$ executes the protocol twice and provides the distinguisher with the views from the second execution of the protocol. We have:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_0] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_1]| = 0.$$

Game G_2**:** Same as G_1 except that $\mathcal{C}$ runs the extraction algorithm on the proofs τ_i for all $i \in \mathsf{AA}$. We set $\mathsf{ska}_{i,1}$ as the extracted values. If one extraction fails, or if $\mathsf{pka}_{i,1} \neq g^{\mathsf{ska}_{i,1}}$ for one i, $\mathcal{C}$ aborts the game. Moreover, for all $i \in [N]$ and $j \in [M]$, $\mathcal{C}$ replaces each computation of $\mathsf{pka}_{i,1}^{\mathsf{skr}_j}$ performed by an honest party by the computation of $\mathsf{pkr}_j^{\mathsf{ska}_{i,1}}$. We have:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_1] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_2]\Big| \leq M\epsilon^{\mathsf{Ext}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

Game G_3**:** Same as G_2 except that $\mathcal{C}$ runs the extraction algorithm on the proofs/signatures of knowledge generated by the dishonest parties. If one extraction fails, or returns a witness which is not related to the corresponding statement, $\mathcal{C}$ aborts the game. There is at most $2N + MN(M+4) + M < 3MN(M+4)$ proofs/signatures of knowledge generated by the dishonest parties during the protocol, we deduce that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_2] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_3]\Big| \leq 3MN(M+4)\epsilon^{\mathsf{Ext}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

Game G_4**:** Same as G_3 except that $\mathcal{C}$ simulates each proof/signature of knowledge generated by the honest parties. Since all the proofs/signatures of knowledge we used are perfectly zero-knowledge, we have:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_3] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_4]| = 0.$$

Game G_5**:** Same as G_4 except that each time that $\mathcal{C}$ simulates the submission of a paper by an honest author, it generates p_1 by committing $\perp$ instead of the authors description: $p_1 \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_3, \perp)$. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_4] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_5]\Big| \leq N\epsilon^{\mathsf{Hiding}}(\lambda)$$

where $\epsilon^{\mathsf{Hiding}}(\lambda)$ is the advantage on the Hiding security of the commitment scheme. We prove this claim by an hybrid argument. We set $G_{5,0} = G_4$, and we define the game $G_{5,i}$ as follows for $0 < i \leq N$:

Game $G_{5,i}$**:** Same as $G_{5,i-1}$ except that when $\mathcal{C}$ simulates the submission of the paper p_i by the author A_i, if $i \notin \mathsf{AA}$, it generates p_1 by committing $\perp$ instead of the authors description: $p_{i,1} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,3}, \perp)$. We claim that:

$$big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i}]\Big| \leq \epsilon^{\mathsf{Hiding}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AA}$. If $i \notin \mathsf{AA}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i}]\Big| > \epsilon^{\mathsf{Hiding}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon_{\mathcal{B}}^{\mathsf{Hiding}}(\lambda) > \epsilon^{\mathsf{Hiding}}(\lambda)$.

$\mathcal{B}$ simulates the game $G_{5,i-1}$ to $\mathcal{D}$, except that when it simulates the submission of the paper p_i, it does not choose the key $\mathsf{ska}_{i,3}$, sets $m_0 = \mathtt{alist}_i$ and $m_1 = \perp$, sends (m_0, m_1) to its challenger, receives the commitment c, and sets $p_{i,1} = c$. Note that this commitment is never opened during the experiment. At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}_{0,\mathsf{C},\mathcal{D}}^{\mathsf{hiding}}(\lambda)$, then $p_{i,1}$ is the commitment of the author list as in the game $G_{5,i-1}$, so $\Pr\Big[\mathbf{Exp}_{0,\mathsf{C},\mathcal{D}}^{\mathsf{hiding}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}_{1,\mathsf{C},\mathcal{D}}^{\mathsf{hiding}}(\lambda)$, then $p_{i,1}$ is the commitment of $\perp$ as in the game $G_{5,i}$, so $\Pr\Big[\mathbf{Exp}_{1,\mathsf{C},\mathcal{D}}^{\mathsf{hiding}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i}]$.

The proof of the claim follows.

Game G_6**:** Same as G_5 except that each time that $\mathcal{C}$ simulates the bidding on a paper p_i by a reviewer R_j such that $j \notin \mathsf{AR}$, it chooses the element pk at random: $\mathsf{pk} \xleftarrow{\$} \mathbb{G}$. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_6] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_5]\Big| \leq NM\epsilon^{\mathsf{DDH}}(\lambda).$$

We prove this claim by an hybrid argument. We set $G_{6,1,0} = G_5$, and we define the game $G_{6,i,j}$ as follows for $0 < i \leq N$ and $0 < j \leq M$:

Game $G_{6,i,j}$**:** If $i > 1$ and $j = 1$, same as $G_{6,i-1,M}$, else same as $G_{6,i,j-1}$, except that when $\mathcal{C}$ simulates the bidding algorithm of the reviewer R_j on the paper p_i, if $j \notin \mathsf{AR}$, then it picks $\mathsf{pk} \xleftarrow{\$} \mathbb{G}$. We claim that, if

$i > 1$ and $j = 1$:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{6,i,0}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i-1,M}]\big| \leq \epsilon^{\mathsf{DDH}}(\lambda)$$

else:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{6,i,j}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{5,i,j-1}]\big| \leq \epsilon^{\mathsf{DDH}}(\lambda).$$

This is clearly verified when $j \in \mathsf{AR}$. If $i \notin \mathsf{AR}$, we prove the claim by reduction.

Case $i = 1$ **or** $j > 1$**:** Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j-1}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{13,i,j}]\big| > \epsilon^{\mathsf{DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon^{\mathsf{DDH}}_{\mathcal{B}}(\lambda) > \epsilon^{\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a DDH triplet (X, Y, Z), and sets $\mathsf{skr}_j \leftarrow X$. It simulates the game $G_{6,i,j-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the random oracle H as follows: when it receives a new input input, it picks a random element q, sets $\mathsf{output} \leftarrow g^q$, and returns output. It stores the tuple $(\mathsf{input}, \mathsf{output}, q)$ so that if it is asked to hash input again, it can return output again.
- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_i, it sets $h \leftarrow Y$ and $\mathsf{pk} \leftarrow Z$. It generates γ by computing $\gamma \leftarrow X^q$, where q is the element in the tuple $(p_{i_*}, H(p_{i_*}), q)$ stored by the random oracle H.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then $\mathsf{pk} = Z$ is the DDH of skr_j and the h used by R_j to bid on the paper p_i, as in the game $G_{6,i,j-1}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{6,i,j-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then Z is a random group element, as in the game $G_{6,i,j}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{6,i,j}]$.

The proof of the claim follows for the case $i = 1$ or $j > 1$. The other case can be proven in the same way.

Game G_7**:** Same as G_6 except that each time that $\mathcal{C}$ simulates the bidding on a paper p_i by a reviewer R_j such that $j \notin \mathsf{AR}$, it chooses the element γ at random: $\gamma \xleftarrow{\$} \mathbb{G}$. Let q_H the number of calls to the random oracle during the experiment. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_7] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_6]\big| \leq M\epsilon^{\mathsf{q_H-DDH}}(\lambda).$$

We prove this claim by an hybrid argument. We set $G_{7,0} = G_6$, and we define the game $G_{7,i}$ as follows for $0 < i \leq M$:

Game $G_{7,i}$**:** Same as $G_{7,i-1}$, except that when $\mathcal{C}$ simulates the bidding algorithm of the reviewer R_i on a paper p_j, if $i \notin \mathsf{AR}$, then it picks $\gamma \xleftarrow{\$} \mathbb{G}$. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i-1}]\big| \leq \epsilon^{q_H-\mathsf{DDH}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AR}$. If $i \notin \mathsf{AR}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i-1}]\big| > \epsilon^{\mathsf{q_H-DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon^{q_H-\mathsf{DDH}}_{\mathcal{B}}(\lambda) > \epsilon^{q_H-\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a q_H-DDH triplet $(X, (Y_k, Z_k)_{q \in [q_H]})$, and sets $\mathsf{skr}_i \leftarrow X$. It simulates the game $G_{7,i-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the q^{th} query to the random oracle H as follows: when it receives a new input input_q, it sets $\mathsf{output}_q \leftarrow Y_q$, and returns output_q. It stores the tuple $(\mathsf{input}_q, \mathsf{output}_q)$ so that if it is asked to hash input again, it can return output_q again.
- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_i, it asks $H(p_i)$ to the random oracle, then it generates γ by setting $\gamma \leftarrow Z_q$, where q is the index of the couple of input/output stored by the random oracle such that $(\mathsf{input}_q, \mathsf{output}_q) = (p_i, Y_q)$.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then for each bid of R_j, $\gamma = Z_q$ is the DDH of $\mathsf{skr}_j = X$ and $H(p_i) = Y_q$, as in the game $G_{7,i-1}$, so $\Pr\Big[\mathbf{Exp}^{q_H-\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then for each bid of R_j, $\gamma = Z_q$ is a random element, as in the game $G_{7,i}$, so $\Pr\Big[\mathbf{Exp}^{q_H-\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i}]$.

The proof of the claim follows.

Game G_8**:** Same as G_7, except that instead of rerunning the protocol a second time, we run the simulator (note that the simulator does not depends on the leaks), as in the case $b = 1$ of the privacy definition. Note that the challenger no longer needs to know the secret keys skr_j of honest reviewers and the identity $\mathtt{alist}_i$ of authors to simulate G_7. Furthermore, the way it simulates the second iteration of the protocol is the same as for the simulator we defined at the beginning of the proof. It is therefore not possible to distinguish this game from the previous one. We can deduce that:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_8] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_7]| \leq 0.$$

Conclusion: From this sequence of game, we deduce the advantage $\epsilon(\lambda)$ of a collusion of dishonest reviewers, authors and chair:

$$\epsilon(\lambda) \leq 4MN(M+4)\epsilon^{\mathsf{Ext}}(\lambda) + N\epsilon^{\mathsf{Hiding}}(\lambda) + (N + q_H)M\epsilon^{\mathsf{DDH}}(\lambda).$$

C. Anonymity in the presence of dishonest attackers excluding the chair.

Here, we consider that $\mathsf{cleak} = (\mathtt{content}_i, \mathtt{RC}_i)_{i\in[N]}$ even if the chair is honest.

$\mathsf{Sim}((\mathsf{pkr}_i, \rho_i)_{i\in[M]}, \mathsf{pkc}, \mathsf{dalgo}, \mathsf{cleak}, (\mathsf{leak}_i)_{i\in\mathsf{AR}})$**:**

Sim simulates correctly the real protocol except for the following cases:

- Simulates each proof/signature of knowledge generated by the honest parties.
- Runs the extraction algorithms $\mathcal{E}$ on the proofs ρ_i (resp. τ_i) for all $i \in \mathsf{AR}$ (resp. $i \in \mathsf{AA}$). We set skr_i (resp. $\mathsf{ska}_{i,1}$) as the extracted values. If one extraction fails, Sim aborts.
- Each time that Sim verifies a proof or a signature of knowledge generated by one of the dishonest parties (i.e., the parties A_i for $i \in \mathsf{AA}$ and R_i for $i \in \mathsf{AR}$), Sim runs the extraction algorithm on this proof and verifies that the statement is indeed related to the extracted secret. If the extraction or verification of the relation fails, Sim aborts.
- For all $i \in [N]\backslash\mathsf{AA}$, when it simulates the submission for the honest author i, Sim generates the element $p_{i,1}$ by computing $p_{i,1} \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_{i,3}, \perp)$.
- When Sim simulates the distribution for the review R_j, it decrypts each hash-ElGamal ciphertext $(p_{i,4}, p_{i,5})$ as follows:
  - if $i \in \mathsf{AA}, \mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i \leftarrow H(p_{i,5}^{r_i}) \oplus p_{i,5}$ (where r_i is the witness extracted from $p_{i,6}$);
  - else, it does not decrypt $(p_{i,4}, p_{i,5})$ and uses the same values as in the simulation of the i^{th} paper submission for retrieving the corresponding plaintext $\mathsf{ska}_{i,1}||\mathsf{ska}_{i,4}||\mathtt{content}_i$;
- For each $j \in [N]\backslash\mathsf{AR}$, when it simulates the binding on the paper p_i for the honest reviewer R_j, Sim picks pk and γ at random in $\mathbb{Z}_q^*$, and it replaces the condition "$\mathsf{pka}_{i,1}^{\mathsf{skr}_j} \in p_{3,i}$" by the condition "$\mathsf{pkr}_j^{\mathsf{ska}_{i,1}} \in p_{3,i}$".

Game G_0**:** In this game, a challenger $\mathcal{C}$ simulates the case $b = 0$.

Game G_1**:** Identical to G_0, except that $\mathcal{C}$ executes the protocol twice and provides the distinguisher with the views from the second execution of the protocol. We have:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_0] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_1]| = 0.$$

Game G_2**:** Same as G_1 except that $\mathcal{C}$ runs the extraction algorithm on the proofs τ_i for all $i \in \mathsf{AA}$. We set $\mathsf{ska}_{i,1}$ as the extracted values. If one extraction fails, or if $\mathsf{pka}_{i,1} \neq g^{\mathsf{ska}_{i,1}}$ for one i, $\mathcal{C}$ aborts the game. Moreover, for all $i \in [N]$ and $j \in [M]$, $\mathcal{C}$ replaces each computation of $\mathsf{pka}_{i,1}^{\mathsf{skr}_j}$ performed by a honest party by the computation of $\mathsf{pkr}_j^{\mathsf{ska}_{i,1}}$. We have:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_1] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_2]\Big| \leq M\epsilon^{\mathsf{Ext}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

Game G_3**:** Same as G_2 except that $\mathcal{C}$ runs the extraction algorithm on the proofs/signatures of knowledge generated by the dishonest parties. If one extraction fails, or returns a witness which is not related to the corresponding statement, $\mathcal{C}$ aborts the game. There is at most $2N + MN(M+4) + M < 3MN(M+4)$ proofs/signatures of knowledge generated by the dishonest parties during the protocol, we deduce that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_2] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_3]\Big| \leq 3MN(M+4)\epsilon^{\mathsf{Ext}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

Game G_4**:** Same as G_3 except that $\mathcal{C}$ simulates each proof/signature of knowledge generated by the honest parties. Since all the proofs/signatures of knowledge we used are perfectly zero-knowledge, we have:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_3] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_4]| = 0.$$

Game G_5**:** Same as G_4 except that each time that $\mathcal{C}$ simulates the submission of a paper by an honest author, it generates p_1 by committing $\perp$ instead of the authors description: $p_1 \leftarrow \mathsf{Com}_{(g,\tilde{g})}(\mathsf{ska}_3, \perp)$. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_4] - \Pr[\mathcal{D} \text{ returns 1 on } G_5]\big| \leq N\epsilon^{\mathsf{Hiding}}(\lambda)$$

where $\epsilon^{\mathsf{Hiding}}(\lambda)$ is the advantage on the Hiding security of the commitment scheme. We prove this claim by an hybrid argument. We set $G_{5,0} = G_4$, and we define the game $G_{5,i}$ as follows for $0 < i \leq N$:

Game $G_{5,i}$**:** Same as $G_{5,i-1}$ except that when $\mathcal{C}$ simulates the submission of the paper p_i by the author A_i, if $i \notin \mathsf{AA}$, it generates p_1 by committing $\perp$ instead of the authors description: $p_{i,1} \leftarrow \mathsf{Com}_{(g,\tilde{g})}(\mathsf{ska}_{i,3}, \perp)$. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_{5,i-1}] - \Pr[\mathcal{D} \text{ returns 1 on } G_{5,i}]\big| \leq \epsilon^{\mathsf{Hiding}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AA}$. If $i \notin \mathsf{AA}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_{5,i-1}] - \Pr[\mathcal{D} \text{ returns 1 on } G_{5,i}]\big| > \epsilon^{\mathsf{Hiding}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon_{\mathcal{B}}^{\mathsf{Hiding}}(\lambda) > \epsilon^{\mathsf{Hiding}}(\lambda)$.

$\mathcal{B}$ simulates the game $G_{5,i-1}$ to $\mathcal{D}$, except that when it simulates the submission of the paper p_i, it does not choose the key $\mathsf{ska}_{i,3}$, sets $m_0 = \mathtt{alist}_i$ and $m_1 = \perp$, sends (m_0, m_1) to its challenger, receives the commitment c, and sets $p_{i,1} = c$. Note that this commitment is never opened during the experiment. At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{hiding}}_{0,\mathsf{C},\mathcal{D}}(\lambda)$, then $p_{i,1}$ is the commitment of the author list as in the game $G_{5,i-1}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{hiding}}_{0,\mathsf{C},\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns 1 on } G_{5,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{hiding}}_{1,\mathsf{C},\mathcal{D}}(\lambda)$, then $p_{i,1}$ is the commitment of $\perp$ as in the game $G_{5,i}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{hiding}}_{1,\mathsf{C},\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns 1 on } G_{5,i}]$.

The proof of the claim follows.

Game G_6**:** Same as G_5 except that at the begining of the game, $\mathcal{C}$ instantiates a dictionary D, then each time $\mathcal{C}$ encrypts a message m using pkc in a ciphertext $c = (p_{i,4}, p_{j,5})$ using hash-Elgamal during the simulation of the sumbission for an author A_i such that $i \notin \mathsf{AA}$, it sets $\mathsf{D}[c] \leftarrow m$. Moreover, each time it has to decrypt a ciphertext $c = (p_{i,4}, p_{i,5})$ during the distribution algorithm:

- if $i \notin \mathsf{AA}$, it does not decrypt c and uses $\mathsf{D}[c]$ as the decryption of c instead.
- if $i \in \mathsf{AA}$, it uses the witness r_i extracted from $p_{i,6}$ and decrypts $c = (p_{i,4}, p_{i,5})$ by computing $H(\mathsf{pkc}^{r_i}) \oplus p_{i,5}$.

Since hash-ElGamal is correct and the all the extractions returned the correct witnesses, this does not change the distribution of views for $\mathcal{D}$, so:

$$|\Pr[\mathcal{D} \text{ returns 1 on } G_6] - \Pr[\mathcal{D} \text{ returns 1 on } G_5]| = 0.$$

Game G_7**:** Same as G_6 except that each time that $\mathcal{C}$ simulates the bidding on a paper p_i by a reviewer R_j such that $j \notin \mathsf{AR}$, it chooses the element pk at random: $\mathsf{pk} \xleftarrow{\$} \mathbb{G}$. We claim that:

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_7] - \Pr[\mathcal{D} \text{ returns 1 on } G_6]\big| \leq NM\epsilon^{\mathsf{DDH}}(\lambda).$$

We prove this claim by an hybrid argument. We set $G_{7,1,0} = G_6$, and we define the game $G_{7,i,j}$ as follows for $0 < i \leq N$ and $0 < j \leq M$:

Game $G_{7,i,j}$**:** If $i > 1$ and $j = 1$, same then as $G_{7,i-1,M}$, else same of $G_{7,i,j-1}$, except that when $\mathcal{C}$ simulates the bidding algorithm of the reviewer R_j on the paper p_i, if $j \notin \mathsf{AR}$, then it picks $\mathsf{pk} \xleftarrow{\$} \mathbb{G}$. We claim that, if $i > 1$ and $j = 1$:

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_{7,i,0}] - \Pr[\mathcal{D} \text{ returns 1 on } G_{6,i-1,M}]\big| \leq \epsilon^{\mathsf{DDH}}(\lambda)$$

else:

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_{7,i,j}] - \Pr[\mathcal{D} \text{ returns 1 on } G_{7,i,j-1}]\big| \leq \epsilon^{\mathsf{DDH}}(\lambda).$$

This is clearly verified when $j \in \mathsf{AR}$. If $i \notin \mathsf{AR}$, we prove the claim by reduction.

Case $i = 1$ **or** $j > 1$**:** Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\big|\Pr[\mathcal{D} \text{ returns 1 on } G_{7,i,j-1}] - \Pr[\mathcal{D} \text{ returns 1 on } G_{7,i,j}]\big| > \epsilon^{\mathsf{DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon_{\mathcal{B}}^{\mathsf{DDH}}(\lambda) > \epsilon^{\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a DDH triplet (X, Y, Z), and sets $\mathsf{skr}_j \leftarrow X$. It simulates the game $G_{7,i,j-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the random oracle H as follows: when it receives a new input input, it picks a random element q, sets $\mathsf{output} \leftarrow g^q$, and returns output. It stores the tuple $(\mathsf{input}, \mathsf{output}, q)$ so that if it is asked to hash input again, it can return output again.

- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_i, it sets $h \leftarrow Y$ and $\mathsf{pk} \leftarrow Z$. It generates γ by computing $\gamma \leftarrow X^q$, where q is the element in the tuple $(p_{i_*}, H(p_{i_*}), q)$ stored by the random oracle H.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then $\mathsf{pk} = Z$ is the DDH of skr_j and the h used by R_j to bid on the paper p_i, as in the game $G_{7,i,j-1}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i,j-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then Z is a random group element, as in the game $G_{7,i,j}$, so $\Pr\Big[\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{7,i,j}]$.

The proof of the claim follows for the case $i = 1$ or $j > 1$. The other case can be proven in the same way.

Game G_8**:** Same as G_7 except that each time that $\mathcal{C}$ simulates the bidding on a paper p_i by a reviewer R_j such that $j \notin \mathsf{AR}$, it chooses the element γ at random: $\gamma \xleftarrow{\$} \mathbb{G}$. Let q_H the number of call to the random oracle during the experiment. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_8] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_7]\Big| \leq M\epsilon^{\mathsf{q_H-DDH}}(\lambda).$$

We prove this claim by an hybrid argument. We set $G_{8,0} = G_7$, and we define the game $G_{8,i}$ as follows for $0 < i \leq M$:

Game $G_{8,i}$**:** Same as $G_{8,i-1}$, except that when $\mathcal{C}$ simulates the bidding algorithm of the reviewer R_i on a paper p_j, if $i \notin \mathsf{AR}$, then it picks $\gamma \xleftarrow{\$} \mathbb{G}$. We claim that:

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i-1}]\Big| \leq \epsilon^{q_H-\mathsf{DDH}}(\lambda).$$

This is clearly verified when $i \in \mathsf{AR}$. If $i \notin \mathsf{AR}$, we prove the claim by reduction.

Assume that there exists $\mathcal{D} \in \mathsf{poly}(\lambda)$ such that

$$\Big|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i}] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i-1}]\Big| > \epsilon^{\mathsf{q_H-DDH}}(\lambda).$$

We build the following algorithm $\mathcal{B} \in \mathsf{poly}(\lambda)$ such that $\epsilon^{q_H-\mathsf{DDH}}_{\mathcal{B}}(\lambda) > \epsilon^{q_H-\mathsf{DDH}}(\lambda)$.

$\mathcal{B}$ receives a q_H-DDH triplet $(X, (Y_k, Z_k)_{q\in[q_H]}$, and sets $\mathsf{skr}_i \leftarrow X$. It simulates the game $G_{8,i-1}$ to $\mathcal{D}$, except that:

- $\mathcal{B}$ simulates the q^{th} query to the random oracle H as follows: when it receives a new input input_q, it sets $\mathsf{output}_q \leftarrow Y_q$, and returns output_q. It stores the tuple $(\mathsf{input}_q, \mathsf{output}_q)$ so that if it is asked to hash input again, it can return output_q again.
- when $\mathcal{B}$ simulates the bidding algorithm for R_j on a paper p_i, it asks $H(p_i)$ to the random oracle, then it generates γ by setting $\gamma \leftarrow Z_q$, where q is the index of the couple of input/output stored by the random oracle such that $(\mathsf{input}_q, \mathsf{output}_q) = (p_i, Y_q)$.

At the end of the experiment, $\mathcal{B}$ returns the same bit as $\mathcal{D}$. We have the two following cases:

- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{0,\mathcal{D}}(\lambda)$, then for each bid of R_j, $\gamma = Z_q$ is the DDH of $\mathsf{skr}_j = X$ and $H(p_i) = Y_q$, as in the game $G_{8,i-1}$, so $\Pr\Big[\mathbf{Exp}^{q_H-\mathsf{DDH}}_{0,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i-1}]$.
- If $\mathcal{B}$ plays the experiment $\mathbf{Exp}^{\mathsf{DDH}}_{1,\mathcal{D}}(\lambda)$, then for each bid of R_j, $\gamma = Z_q$ is a random element, as in the game $G_{8,i}$, so $\Pr\Big[\mathbf{Exp}^{q_H-\mathsf{DDH}}_{1,\mathcal{D}}(\lambda) = 1\Big] = \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_{8,i}]$.

The proof of the claim follows.

Game G_9**:** Same as G_8, except that instead of rerunning the protocol a second time, we run the simulator (note that the simulator does not depends on the leaks), as in the case $b = 1$ of the privacy definition. Note that the challenger no longer needs to know the secret keys skr_j of honest reviewers and the identity $\texttt{alist}_i$ of authors to simulate G_8. Furthermore, the way it simulates the second iteration of the protocol is the same as for the simulator we defined at the beginning of the proof. It is therefore not possible to distinguish this game from the previous one. We can deduce that:

$$|\Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_9] - \Pr[\mathcal{D} \text{ returns } 1 \text{ on } G_8]| \leq 0.$$

Conclusion: From this sequence of game, we deduce the advantage $\epsilon(\lambda)$ of a collusion of dishonest reviewers, authors with honest chair:

$$\epsilon(\lambda) \leq 4MN(M+4)\epsilon^{\mathsf{Ext}}(\lambda) + N\epsilon^{\mathsf{Hiding}}(\lambda) + (N + q_H)M\epsilon^{\mathsf{DDH}}(\lambda)$$

This concludes the proof of Theorem 1.

Appendix D
Verifiability Proofs

We now demonstrate the result in Theorem 3 (i.e., that BlindReview satisfies end-to-end verifiability) via a series of Lemmata (Lemmata 1–5).

Lemma 1. *If* $(\mathsf{Sig}_g, \mathsf{Ver}_g)$ *satisfies EUF-CMA security then BlindReview satisfies submission authenticity.*

Proof. The adversary in the submission authenticity experiment succeeds if they output a submission $\mathbf{p}_*$ on behalf of an honest author pka_* that is not generated by the challenger such that algorithm $\mathsf{VerSubmit}$ returns 1. Recall, that this occurs if the signature $p_{7,*}$ and the non-interactive proof $p_{6,*}$ included in the submission verify. Note that an adversary can trivially generate a proof $p_{6,*}$ that will verify. Thus, it is clear that the submission authenticity property reduces to the unforgeability of the signature scheme used to sign the submission.

More formally, let $\mathcal{A}$ be an adversary in the submission authenticity experiment that has access to $(\mathsf{pkc}, \{(\mathsf{pkr}_i, \rho_i)\}_{i\in[M]}, \{\mathsf{pka}_j, \rho_j\}_{j\in[N]})$. Following the submission phase of the experiment, $\mathcal{A}$ will also have a set of corrupt author secret keys $\{\mathsf{ska}_k\}_{k\in[|A_c|]}$ and a set of paper submissions $\{\mathbf{p}_j\}_{j\in[N]}$. $\mathcal{A}$ then outputs a target author pka_* and target paper $\mathbf{p}_*$ such that $(\mathsf{pka}_*, \mathbf{p}_*) \notin \mathcal{S}$. That is to say, the target paper is not generated by the challenger on behalf of uncorrupted author pka_*.

We show that if $\mathcal{A}$ succeeds in the submission authenticity experiment then we can construct an adversary $\mathcal{A}'$ that succeeds in the $\mathsf{EUF-CMA}$ experiment (Definition 9) for the Schnorr signature scheme. We define adversary $\mathcal{A}'$ in Figure 24. It is clear that the inputs to $\mathcal{A}$ are distributed identically to the submission authenticity experiment. Indeed, all key pairs and submissions are generated identically. Moreover, if $\mathcal{A}$ succeeds in the submission authenticity experiment then $(\mathsf{pka}_*, (p_1, p_2, p_3, p_4, p_5, p_6), p_7)$ is a valid key, message, signature tuple and $\mathsf{SIG.Verify}$ will return 1. We also assume that $(\mathsf{pka}_{2,*}, (p_1, p_2, p_3, p_4, p_5, p_6)) \notin \mathsf{Qsign}$ and $\mathsf{pka}_{2,*} \notin \mathsf{Qcorrupt}$. Therefore, $\mathcal{A}'$ succeeds in the $\mathsf{EUF-CMA}$ experiment. Then, by contradiction, the result holds. □

Lemma 2. *If the proof systems used in the bid phase satisfy extractability and zero-knowledge then BlindReview satisfies bid eligibility under the discrete logarithm assumption.*

Proof. We use the following sequence of games:

Game G_0**:** This game is the experiment $\mathsf{Exp}^{\mathsf{BE}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda, \ell)$

Game G_1**:** Same as G_0 but, rather than verifying the proofs generated by the adversary up to the bid phase, $\mathcal{C}$ runs the extraction algorithm on all proofs and stores the corresponding secret witnesses. If one extraction fails, or returns a witness which is not related to the corresponding statement, $\mathcal{C}$ aborts the game. There is at most $2N + M + NM(2 + M)$ proofs generated by the adversary during phases up to and including paper bidding. Hence, we deduce that:

$$\begin{aligned}\big|\Pr[\mathcal{A} \text{ wins } G_1] - \Pr[\mathcal{A} \text{ wins } G_0]\big| \\ \leq (2N + M + NM(2 + M))\epsilon^{\mathsf{Ext}}(\lambda)\end{aligned}$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

At this point, since all proofs are verified in $\mathsf{VerBidPool}$, the extraction of proofs of the form π guarantees that, if the adversary wins, the bids they produced (especially those of the set B_c) are well-formed, use a key pkr_j from a registered reviewer, and use different reviewers keys for the paper i_* (by checking that each $\gamma = H(p_{i_*})^{\mathsf{skr}_j}$ is unique). The extraction of proofs of the form $\hat{\pi}_j$ guarantees that, if $m \neq 0$, then the author of the paper, if honest (which is the case for the bids of the set B_c), has not declared a conflict with the owner of the key pkr_j.

Game G_2**:** Same as G_1 except that $\mathcal{C}$ simulates each proof generated by the honest entities. Since all the proofs we used are perfectly zero-knowledge, we have:

$$|\Pr[\mathcal{A} \text{ wins } G_2] - \Pr[\mathcal{A} \text{ wins } G_1]| = 0.$$

Game G_3**:** Same as G_2 except that $\mathcal{C}$ picks an index $j_* \xleftarrow{\$} [M]$ at random, then if $\mathcal{C}$ extracts the secret key skr_j of an honest uncorrupted reviewer from one proof/signature of knowledge generated by the adversary such that $j \neq j_*$, $\mathcal{C}$ aborts the games and returns 0. We have that:

$$\Pr[\mathcal{A} \text{ wins } G_3] \leq \frac{1}{M} \Pr[\mathcal{A} \text{ wins } G_2].$$

Game G_4**:** Same as G_3 except that if $\mathcal{C}$ extracts the secret key skr_{j_*} from one proof/signature of knowledge generated by the adversary and the reviewer j_* is not corrupted at the time of the extraction, then it sets $\mathsf{abort} = 1$, it aborts the games, and it returns 0. We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_4] - \Pr[\mathcal{A} \text{ wins } G_3]| \leq \epsilon^{\mathsf{DL}}(\lambda).$$

We prove this claim by reduction. We define $\mathcal{B}(y)$ as an adversary against the discrete logarithm in a group $\langle g\rangle$ of prime order p as follows. $\mathcal{B}$ setups the BlindReview protocol with this group, and plays the game G_3 with $\mathcal{A}$, except that it sets $\mathsf{pkr}_{j_*} \leftarrow y$ and aborts the game is $\mathcal{A}$ corrupts the reviewer j_*. To simulate the behaviour of reviewer j_* when bidding without knowing their secret key skr_{j_*}, $\mathcal{B}$ acts as follows: if it produces a bid on a paper generated by an honest author of index i, $\mathcal{B}$ can simulate zero-knowledge proofs in a manner consistent with the conflicts $\mathtt{RC}_i$. If it produces a bid on a paper

```
$\mathcal{A}'^{\mathsf{Osign},\mathsf{Ocorrupt}}(\{\mathsf{pk}_{2,j}\}_{j\in[N]})$
$(\mathsf{pkc},\mathsf{skc}) \leftarrow \mathsf{C.KGen}()$
for $i = 1,\ldots,M$ : $(\mathsf{pkr}_i,\mathsf{skr}_i,\rho_i) \leftarrow \mathsf{R.KGen}()$
for $j = 1,\ldots,N$
    $(\mathsf{ska}_{1,j},\mathsf{ska}_{3,j},\mathsf{ska}_{4,j}) \leftarrow (\mathbb{Z}_q^*)^3$
    $\mathsf{ska}_j \leftarrow (\mathsf{ska}_{1,j},\bot,\mathsf{ska}_{3,j},\mathsf{ska}_{4,j})$
    $\mathsf{pka}_{1,j} \leftarrow g^{\mathsf{ska}_{1,j}}$
    $\tau_j \leftarrow \mathsf{ZK.Prove}(\mathsf{pka}_{1,j},\mathsf{ska}_{1,j})$
    $\mathsf{pka}_j \leftarrow (\mathsf{pka}_{1,j},\mathsf{pka}_{2,j})$
return $(\mathsf{pkc},\{(\mathsf{pkr}_i,\rho_i)\}_{i\in[M]},\{(\mathsf{pka}_j,\tau_j)\}_{j\in[N]})$
```

```
$\mathcal{A}'^{\mathsf{Osign},\mathsf{Ocorrupt}}(\mathsf{pka}^*,\mathbf{p}^*)$
parse $\mathsf{pka}^*$ as $(\mathsf{pka}_1^*,\mathsf{pka}_2^*)$
parse $\mathbf{p}^*$ as $(p_1^*,p_2^*,p_3^*,p_4^*,p_5^*,p_6^*,p_7^*)$
return $(\mathsf{pka}_2^*,(p_1^*,p_2^*,p_3^*,p_4^*,p_5^*,p_6^*),p_7^*)$
```

```
$\mathcal{A}'^{\mathsf{Osign},\mathsf{Ocorrupt}}(\mathtt{alist},\mathtt{content},\mathtt{RC})$//if $\mathsf{pka} \in A_u$
parse $\mathsf{ska}$ as $(\mathsf{ska}_1,\bot,\mathsf{ska}_3,\mathsf{ska}_4)$
$p_1 \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_3,\mathtt{alist})$
$p_2 \leftarrow \mathsf{Com}_{(g,\bar{g})}(\mathsf{ska}_4,\mathtt{content})$
for $j \in [M]$
    if $j \in \mathtt{RC}$ : $x_j \leftarrow \mathsf{pkr}_j^{\mathsf{ska}_1}$
    else $x_j \xleftarrow{\$} \mathbb{G}$
$p_3 \leftarrow \mathsf{Shuffle}(\{x_j\}_{1\le j\le M})$
$r \xleftarrow{\$} \mathbb{Z}_q^*$
$p_4 \leftarrow g^r$
$p_5 \leftarrow H(\mathsf{pkc}^r) \oplus [\mathsf{ska}_1||\mathsf{ska}_4||\mathtt{content}]$
$p_6 \leftarrow \mathsf{ZK}\{r : g^r = p_4\}$
$p_7 \leftarrow \mathsf{Osign}(\mathsf{pka}_2,(p_1||p_2||p_3||p_4||p_5||p_6))$
return $\mathbf{p} = (p_1,p_2,p_3,p_4,p_5,p_6,p_7)$
```

```
$\mathcal{A}'^{\mathsf{Osign},\mathsf{Ocorrupt}}(\mathsf{pka})$//if $\mathsf{pka} \in A_c$
$\mathsf{ska}_2 \leftarrow \mathsf{Ocorrupt}(\mathsf{pka}_2)$
return $\mathsf{ska} = (\mathsf{ska}_1,\mathsf{ska}_2,\mathsf{ska}_3,\mathsf{ska}_4)$
```

Figure 24: Adversary $\mathcal{A}'$ that breaks the $\mathsf{EUF-CMA}$ security of signature scheme SIG in the proof of Lemma 1.

generated by a corrupted author of index i, then $\mathcal{B}$ uses the secret key ska_i that it retrieved using the extractor on τ_i and checks whether $y^{\mathsf{ska}_i} \in p_{i,3}$ to see if has conflicts instead of $\mathsf{pka}_i^{\mathsf{skr}_{j_*}} \in p_{i,3}$ in order to simulate the zero-knowledge proofs in a manner consistent with the conflicts. If $\mathcal{B}$ extracts the secret key skr_{j_*}, then it aborts the game and returns it. Until the abort event $\mathsf{abort} = 1$, G_4 is perfectly simulated, so we have that:

$$\Pr[\mathsf{abort} = 1] = \Pr[\mathcal{B} \text{ wins}] \le \epsilon^{\mathsf{DL}}(\lambda),$$

which concludes the proof of the claim.

At this step, if the adversary wins, then they not produced a bid by themselves using the key of an honest, uncorrupted reviewer.

In G_4, if $\mathsf{VerBidPool}(\{\mathsf{pka}_i,\mathbf{p}_i,\{b_{i,j}\}_{j\in[M]}\}_{i\in[N]}, \{\mathsf{pkr}_i\}_{i\in[M]})$ returns true, then all the bids in B_c use secret keys of registered reviewers (extracted by the challenger from the bids), use a different secret key for each bid, and use keys of corrupted reviewers (i.e. reviewers j where $j \in R_c$), which implies that $|B_c| \le |R_c|$. Moreover, if in addition $\mathsf{pka}_{i_*} \in A_u$, since the opponent cannot produce a verified bid with a mark $m \neq 0$ if it has conflict with the (honest) author of the paper, they will produce at most as many bids with non-null mark as they control corrupt reviewers who have no conflict with the paper, which implies that $|\{j|b_{i_*,j} \in \mathsf{B}_c \wedge m_{i_*,j} \neq 0\}| \le |R_c \backslash \mathtt{RC}_{i_*}|$. Finally, we have that:

$$\Pr[\mathcal{A} \text{ wins } G_4] = 0.$$

Conclusion: Finally, we deduce that:

$$\Pr\Big[\mathsf{Exp}^{\mathsf{BE}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda,\ell) = 1\Big] \le (2N + M + NM(2+M))\epsilon^{\mathsf{Ext}}(\lambda) + M\epsilon^{\mathsf{DL}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol, which concludes the proof. □

Lemma 3. *If the proof systems used until the assign phase are sound and zero-knowledge then BlindReview satisfies fair assign under the discrete logarithm assumption.*

Proof. We use the following sequence of games:

Game G_0: This game is the experiment $\mathsf{Exp}^{\mathsf{FA}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda,\ell)$

Game G_1: Same as G_0 except that instead of verifying the proofs/signatures of knowledge generated by the adversary until the assignment phase, $\mathcal{C}$ runs the extraction algorithm on it and stores the corresponding secret witnesses. If one extraction fails, or returns a witness which is not related to the corresponding statement, $\mathcal{C}$ aborts the game. There is at most $2N + M + 5NM + NM^2 + (NM)^2$ proofs/signatures of knowledge generated by the dishonest parties during these phases, we deduce that:

$$|\Pr[\mathcal{A} \text{ wins } G_1] - \Pr[\mathcal{A} \text{ wins } G_0]| \\ \leq (2N + M + 7NM + NM^2)\epsilon_{\mathsf{Ext}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

At this point, since all the proofs/signatures of the bids and assignation acceptance are verified, the extraction of the proofs π in the bids guarantee that if the adversary wins, the bids they produced are well-formed, use a key pkr_j from a registered reviewer, and use different reviewers keys for the paper i_* (by checking in VerDecision that each $\gamma = H(p_{i_*})^{\mathsf{skr}_j}$ used for the reviews is different from each other). The extraction of proofs ψ guarantees that if the adversary rejects a assignment for a bid, then the reviewer key skr_j that has been used for this bid has already been used for at least ℓ bids that have been assigned and accepted before.

Game G_2: Same as G_1 except that $\mathcal{C}$ simulates each proof/signature of knowledge generated by the honest parties. Since all the proofs/signatures of knowledge we used are perfectly zero-knowledge, we have:

$$|\Pr[\mathcal{A} \text{ wins } G_2] - \Pr[\mathcal{A} \text{ wins } G_1]| = 0.$$

Game G_3: Same as G_2 except that $\mathcal{C}$ picks an index $j_* \xleftarrow{\$} [M]$ at random, then if $\mathcal{C}$ extracts the secret key skr_j of an honest uncorrupted reviewer from one proof/signature of knowledge generated by the adversary such that $j \neq j_*$, $\mathcal{C}$ aborts the games and returns 0. We have that:

$$\Pr[\mathcal{A} \text{ wins } G_3] \leq \frac{1}{M} \Pr[\mathcal{A} \text{ wins } G_2].$$

Game G_4: Same as G_3 except that if $\mathcal{C}$ extracts the secret key skr_{j_*} from one proof/signature of knowledge generated by the adversary and the reviewer j_* is not corrupted at the time of the extraction, then it sets $\mathsf{abort}_4 = 1$, it aborts the games, and it returns 0. We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_4] - \Pr[\mathcal{A} \text{ wins } G_3]| \leq \epsilon^{\mathsf{DL}}(\lambda).$$

We prove this claim by reduction. We define $\mathcal{B}(y)$ as an adversary against the discrete logarithm in a group $\langle g \rangle$ of prime order p as follows. $\mathcal{B}$ setups the BlindReview protocol with this group, and plays the game G_3 with $\mathcal{A}$, except that it sets $\mathsf{pkr}_{j_*} \leftarrow y$ and aborts the game is $\mathcal{A}$ corrupts the reviewer j_*. To simulate the behaviour of reviewer j_* when bidding without knowing their secret key skr_{j_*}, $\mathcal{B}$ acts as follows: if it produces a bid on a paper generated by an honest author of index i, $\mathcal{B}$ can simulate zero-knowledge proofs in a manner consistent with the conflicts $\mathtt{RC}_i$. If it produces a bid on a paper generated by a corrupted author of index i, then $\mathcal{B}$ uses the secret key ska_i that it retrieved using the extractor on τ_i and checks whether $y^{\mathsf{ska}_i} \in p_{i,3}$ to see if has conflicts instead of $\mathsf{pka}_i^{\mathsf{skr}_{j_*}} \in p_{i,3}$ in order to simulate the zero-knowledge proofs in a manner consistent with the conflicts. If $\mathcal{B}$ extracts the secret key skr_{j_*}, then it aborts the game and returns it. Until the abort event $\mathsf{abort}_4 = 1$, G_4 is perfectly simulated, so we have that:

$$\Pr[\mathsf{abort}_4 = 1] = \Pr[\mathcal{B} \text{ wins}] \leq \epsilon^{\mathsf{DL}}(\lambda),$$

which concludes the proof of the claim.

At this step, if the adversary wins, then they not produced an assignation response by themselves using the key of an honest, uncorrupted reviewer, which ensures that there is no $(a_k, \mathbf{p}'_k, b_k, \mathtt{response}_k, \alpha_k)$ in $\mathcal{L}_*$ such that b_k has been produced by an honest uncorrupted reviewer.

In G_4, if each bid is verified, then each bid produced by the adversary uses secret keys of a dishonest or corrupted registred reviewers (extracted by the challenger from the bids), and use a different secret key for each paper. Thus, if the adversary rejects an assignment, they have already accepted ℓ assignments for different papers for the corresponding key (since they cannot produce two bids for the same paper with the same key). This ensures that$|(a_k, \mathbf{p}'_k, b_k, \mathtt{response}_k, \alpha_k) \in \mathcal{L}_* | \mathbf{p}'_k = \mathbf{p}_{j_*} \wedge \mathtt{response}_k = 0\}| < i_* + 1$

Game G_5: Same as G_4 except that if

a_{k_*} has been generated by the challenger, then $\mathcal{C}$ sets $\mathsf{abort}_5 = 1$, aborts the games, and returns 0. We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_5] - \Pr[\mathcal{A} \text{ wins } G_4]| \leq \epsilon^{\mathsf{DL}}(\lambda).$$

We prove this claim by reduction. We define $\mathcal{B}(y)$ as an adversary against the discrete logarithm in a group $\langle g \rangle$ of prime order p as follows. $\mathcal{B}$ setups the BlindReview protocol with this group, and plays the game G_4 with $\mathcal{A}$, except that it sets $\mathsf{pkc} \leftarrow y$. Note that since each signature of the chair is simulated, $\mathcal{B}$ can produce them without knowing the key skc, and therefore perfectly simulate the behaviour of the chair except for the distribution. $\mathcal{B}$ simulates $\mathsf{Distribute}$ as in G_4 except that it computes $H(\mathsf{pkc}^r) \oplus p_{i,5}$ instead of $H(p_{i,4}^{\mathsf{skc}}) \oplus p_{i,5}$, where $r = \log_g(p_{i,4})$. In the case where p_i was generated honestly, $\mathcal{B}$ itself chose r when the paper was submitted, and in the case where p_i was produced by the adversary, r was extracted from the proof $p_{i,6}$, so in all cases, if the game has not been aborted before, $\mathcal{B}$ knows the value of r. If $\mathcal{B}$ extracts the secret key skc, then it aborts the game and returns it. Until the abort event $\mathsf{abort}_5 = 1$, G_4 is perfectly simulated, so we have that:

$$\Pr[\mathsf{abort}_5 = 1] = \Pr[\mathcal{B} \text{ wins}] \leq \epsilon^{\mathsf{DL}}(\lambda),$$

which concludes the proof of the claim.

At this step, since a_{k_*} can not be generated by the challenger, we have that:

$$\Pr[\mathcal{A} \text{ wins } G_5] = 0.$$

Conclusion: Finally, we deduce that:

$$\begin{aligned}\Pr\Big[\mathsf{Exp}^{\mathsf{FA}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda,\ell) = 1\Big] \\ \leq (2N + M + 5NM + NM^2 + (NM)^2)\epsilon^{\mathsf{Ext}}(\lambda) \\ + 2M\epsilon^{\mathsf{DL}}(\lambda)\end{aligned}$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol, which concludes the proof.

□

Lemma 4. *If the proof systems used in the bid phase are sound and zero-knowledge then BlindReview satisfies reviewer and decision authenticity under the discrete logarithm assumption.*

Proof. We use the following sequence of games:

Game G_0**:** This game is the experiment $\mathsf{Exp}^{\mathsf{RDA}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda,\ell)$

Game G_1**:** Same as G_0 except that instead of verifying the proofs/signatures of knowledge generated by the adversary until the bid phase and during the reviewing/decision phase, $\mathcal{C}$ runs the extraction algorithm on it and stores the corresponding secret witnesses. If one extraction fails, or returns a witness which is not related to the corresponding statement, $\mathcal{C}$ aborts the game. There is at most $2N + M + NM(2 + M) + 4N$ proofs/signatures of knowledge generated by the dishonest parties during these phases, we deduce that:

$$\begin{aligned}&|\Pr[\mathcal{A} \text{ wins } G_1] - \Pr[\mathcal{A} \text{ wins } G_0]| \\ &\quad \leq (2N + M + NM(2 + M) + 4N)\epsilon^{\mathsf{Ext}}(\lambda)\end{aligned}$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol.

At this point, since all th proofs/signatures of the bids and reviews are verified in VerBid and $\mathsf{VerDecision}$, the extraction of the proofs π guarantees that if the adversary wins, the bids they produced (especially those of the set B_c) are well-formed, use a key pkr_j from a registered reviewer, and use different reviewers keys for the paper i_* (by checking in $\mathsf{VerDecision}$ that each $\gamma = H(p_{i_*})^{\mathsf{skr}_j}$ used for the reviews is different from each other). The extraction of proofs $\hat{\pi}_j$ guarantees that if $m \neq 0$, then the author of the paper, if honest (which is the case for the bids of the set B_c), has not declared a conflict with the owner of the key pkr_j.

Game G_2**:** Same as G_1 except that $\mathcal{C}$ simulates each proof/signature of knowledge generated by the honest parties. Since all the proofs/signatures of knowledge we used are perfectly zero-knowledge, we have:

$$|\Pr[\mathcal{A} \text{ wins } G_2] - \Pr[\mathcal{A} \text{ wins } G_1]| = 0.$$

Game G_3**:** Same as G_2 except that $\mathcal{C}$ picks an index $j_* \xleftarrow{\$} [M]$ at random, then if $\mathcal{C}$ extracts the secret key skr_j of an honest uncorrupted reviewer from one proof/signature of knowledge generated by the adversary such that $j \neq j_*$, $\mathcal{C}$ aborts the games and returns 0. We have that:

$$\Pr[\mathcal{A} \text{ wins } G_3] \leq \frac{1}{M} \Pr[\mathcal{A} \text{ wins } G_2].$$

Game G_4**:** Same as G_3 except that if $\mathcal{C}$ extracts the secret key skr_{j_*} from one proof/signature of knowledge generated by the adversary and the reviewer j_* is not corrupted at the time of the extraction, then it sets $\mathsf{abort}_4 = 1$, it aborts the games, and it returns 0. We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_4] - \Pr[\mathcal{A} \text{ wins } G_3]| \leq \epsilon^{\mathsf{DL}}(\lambda).$$

We prove this claim by reduction. We define $\mathcal{B}(y)$ as an adversary against the discrete logarithm in a group $\langle g \rangle$ of prime order p as follows. $\mathcal{B}$ setups the BlindReview protocol with this group, and plays the game G_3 with $\mathcal{A}$, except that it sets $\mathsf{pkr}_{j_*} \leftarrow y$ and aborts the game is $\mathcal{A}$ corrupts the reviewer j_*. To simulate the behaviour of reviewer j_* when bidding without knowing their secret key skr_{j_*}, $\mathcal{B}$ acts as follows: if it produces a bid on a paper generated by an honest author of index i, $\mathcal{B}$ can simulate zero-knowledge proofs in a manner consistent with the conflicts $\mathtt{RC}_i$. If it produces a bid on a paper generated by a corrupted author of index i, then $\mathcal{B}$ uses the secret key ska_i that it retrieved using the extractor on τ_i and checks whether $y^{\mathsf{ska}_i} \in p_{i,3}$ to see if has conflicts instead of $\mathsf{pka}_i^{\mathsf{skr}_{j_*}} \in p_{i,3}$ in order to simulate the zero-knowledge proofs in a manner consistent with the conflicts. If $\mathcal{B}$ extracts the secret key skr_{j_*}, then it aborts the game and returns it. Until the abort event $\mathsf{abort}_4 = 1$, G_4 is perfectly simulated, so we have that:

$$\Pr[\mathsf{abort}_4 = 1] = \Pr[\mathcal{B} \text{ wins}] \leq \epsilon^{\mathsf{DL}}(\lambda),$$

which concludes the proof of the claim.

At this step, if the adversary wins, then they not produced a bid or review by themselves using the key of an honest, uncorrupted reviewer. In G_4, if each bid/review in $(b_j, m_j, w_j, z_j)_{j\in[3]}$ and the décision $(\tilde{w}, \tilde{z})$ are verified, then all the reviews in S_c use secret keys of a dishonest or corrupted registred reviewers (extracted by the challenger from the bids), and use a different secret key for each review. Moreover, if in addition $\mathsf{pka}_{i_*} \in A_u$, since the adversary cannot produce a verified bid with a mark $m \neq 0$ if it has conflict with the (honest) author of the paper, they will produce at most as many bids with non-null mark as they control corrupt reviewers who have no conflict with the paper, which implies that $|R_c \backslash \mathtt{RC}_{i_*}| \geq |S_c|$.

Game G_5**:** Same as G_4 except that if $(\tilde{w}, \tilde{z})$ was not generated by $\mathcal{C}$, then $\mathcal{C}$ sets $\mathsf{abort}_5 = 1$, aborts the games, and returns 0. We claim that:

$$|\Pr[\mathcal{A} \text{ wins } G_5] - \Pr[\mathcal{A} \text{ wins } G_4]| \leq \epsilon^{\mathsf{DL}}(\lambda).$$

We prove this claim by reduction. We define $\mathcal{B}(y)$ as an adversary against the discrete logarithm in a group $\langle g\rangle$ of prime order p as follows. $\mathcal{B}$ setups the BlindReview protocol with this group, and plays the game G_4 with $\mathcal{A}$, except that it sets $\mathsf{pkc} \leftarrow y$. Note that since each signature of the chair is simulated, $\mathcal{B}$ can produce them without knowing the key skc, and therefore perfectly simulate the behaviour of the chair except for the distribution. $\mathcal{B}$ simulates $\mathsf{Distribute}$ as in G_4 except that it computes $H(\mathsf{pkc}^r) \oplus p_{i,5}$ instead of $H(p_{i,4}^{\mathsf{skc}}) \oplus p_{i,5}$, where $r = \log_g(p_{i,4})$. In the case where p_i was generated honestly, $\mathcal{B}$ itself chose r when the paper was submitted, and in the case where p_i was produced by the adversary, r was extracted from the proof $p_{i,6}$, so in all cases, if the game has not been aborted before, $\mathcal{B}$ knows the value of r. If $\mathcal{B}$ extracts the secret key skc, then it aborts the game and returns it. Until the abort event $\mathsf{abort}_5 = 1$, G_4 is perfectly simulated, so we have that:

$$\Pr[\mathsf{abort}_5 = 1] = \Pr[\mathcal{B} \text{ wins}] \leq \epsilon^{\mathsf{DL}}(\lambda),$$

which concludes the proof of the claim.

At this step, we have that:

$$\Pr[\mathcal{A} \text{ wins } G_5] = 0.$$

Conclusion: Finally, we deduce that:

$$\Pr\left[\mathsf{Exp}^{\mathsf{RDA}}_{\mathcal{A},\text{BlindReview},M,N}(\lambda, \ell) = 1\right] \\ \leq (2N + M + NM(2 + M) + 4N)\epsilon^{\mathsf{Ext}}(\lambda) \\ + 2M\epsilon^{\mathsf{DL}}(\lambda)$$

where $\epsilon_{\mathsf{Ext}}(\lambda)$ is the maximum extraction failure probability over all extractors of the proofs/signatures of knowledge used in the protocol, which concludes the proof. □

Lemma 5. *If (Sig_g, Ver_g) satisfies* $\mathsf{EUF-CMA}$ *security then BlindReview satisfies camera ready authenticity.*

Proof. Camera ready authenticity holds if the adversary can output a submission $\mathbf{p}_*$ and a camera ready version of the submission t_*, accompanied with the original content $\mathtt{content}_*$ and updated content $\mathtt{CRcontent}_*$ on behalf of an honest author pka_* such that algorithm $\mathsf{VerCameraReady}$ returns 1, and the challenge updated content was not produced by the challenger in the camera ready authenticity experiment. This requires that the adversary can produce a signature on behalf of an author, without access to the author's secret signing key. Thus, camera ready authenticity reduces to the unforgeability of the signature scheme used to sign the updated content of the paper.

More specifically, let $\mathcal{A}$ be an adversary in the camera ready authenticity experiment who, at the end of the experiment run, finally outputs a target author pka_* in $\mathcal{A}_u$ and values $\mathbf{p}_*, t_*, \mathtt{alist}_*, \mathtt{content}_*$ and $\mathtt{CRcontent}_*$.

If $\mathcal{A}$ succeeds in the camera ready authenticity experiment then we can construct, similarly to Figure 24, an adversary $\mathcal{A}'$ that succeeds in the $\mathsf{EUF-CMA}$ experiment (Definition 9) for $(\mathsf{Sig}_g, \mathsf{Ver}_g)$.

Adversary $\mathcal{A}'$, having access to its own corruption and signing oracles, can generate all inputs required for $\mathcal{A}$ with the identical distribution to the camera ready experiment. Therefore, if $\mathcal{A}$ is a winning adversary then $(\mathsf{pka}_{*2}, [\mathbf{p}_*||\mathtt{alist}_*||\mathtt{content}_*||\mathtt{CRcontent}_*], t_{*3})$ is a valid key, message, signature tuple and $\mathsf{SIG.Verify}$ will return 1. We also assume that $(\mathsf{pka}_{*2}, [\mathbf{p}_*||\mathtt{alist}_*||\mathtt{content}_*||\mathtt{CRcontent}_*]) \notin \mathsf{Qsign}$ and $\mathsf{pka}_{*2} \notin \mathsf{Qcorrupt}$. We therefore have that $\mathcal{A}'$ succeeds in the $\mathsf{EUF-CMA}$ experiment, and that, by contradiction, the result holds. □