The Abe-Okamoto Partially Blind Signature Scheme Revisited

Julia Kastner[1]*, Julian Loss[2]**, and Jiayu Xu[3]***

[1] Department of Computer Science, ETH Zurich, Zurich, Switzerland
julia.kastner@inf.ethz.ch

[2] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
loss@cispa.de

[3] School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA xujiay@oregonstate.edu

Abstract. Partially blind signatures, an extension of ordinary blind signatures, are a primitive with wide applications in e-cash and electronic voting. One of the most efficient schemes to date is the one by Abe and Okamoto (CRYPTO 2000), whose underlying idea — the OR-proof technique — has served as the basis for several works.
We point out several subtle flaws in the original proof of security, and provide a new detailed and rigorous proof, achieving similar bounds as the original work. We believe our insights on the proof strategy will find useful in the security analyses of other OR-proof-based schemes.

1 Introduction

Blind signatures, first introduced by Chaum [11], are a fundamental cryptographic primitive. They allow two parties, a *signer* who holds the secret key and a *user* who holds the message, to jointly generate a signature. Roughly speaking, security requires that the signer learns nothing about the message nor the signature (*blindness*), and the user cannot forge a signature that does not result from its interaction with the signer (*one-more unforgeability*). Blind signatures have found extensive applications in settings where anonymity is of great concern, such as e-cash [11, 13, 19, 37] and electronic voting [12, 18].

However, in a blind signature scheme, the signer has absolutely no control over the message it signs. This leads to various shortcomings in practice. First, in an e-cash system where a bank uses blind signatures to issue its coins, to avoid the double spending problem, the bank has to keep record of all coins that have been spent; to prevent the ledger from growing unlimitedly, old coins need to expire after a period of time, so that the corresponding entries in the ledger can be deleted. Second, there is no way to inscribe the value or expiration date of a coin. Thus, the bank has to use a different public key for each value/expiration

* Work done while supported by ERC Project PREP-CRYPTO 724307
** Work done while at University of Maryland
*** Work done while at George Mason University

date, and anyone who spends or receives these coins has to maintain a list of all public keys, which has to evolve over time when old coins expire and are replaced by new ones. Similarly, in electronic voting, voters have to download a new public key for each election.

To address these issues, Abe & Fujisaki [2] proposed an extension called *partially* blind signatures, which allow a signer to explicitly include some common information (called the *tag*) in the signature. The tag is agreed upon by the signer and the user in advance and remains unblinded throughout the signing procedure; for example, it can be the date of issue or the value of the electronic coin. Setting the tag to the empty string yields an ordinary blind signature scheme. Informally, a partially blind signature scheme is secure if it satisfies (1) *partial blindness*: for multiple signatures that use the same tag, an adversarial signer cannot link these signatures to the signing sessions they originate from; and (2) *one-more-unforgeability*, or *OMUF security*: an adversarial user that interacts with the signer in at most ℓ many sessions, cannot output more than ℓ valid message-signature pairs.

Despite 25 years of research, there have been very few partially blind signature schemes ever proposed. The most efficient scheme up to date is the one proposed by Abe and Okamoto (AO) [4], which involves only 2 group (multi-) exponentiations for the signer and 4 (multi-)exponentiations for the user. The scheme is based on the classical OR-proof technique for obtaining witness indistinguishable protocols by Cramer *et al.* [15], and its security proof involves an intricate rewinding argument. The ideas behind both the scheme and its security proof repeatedly appear in blind signatures [1, 5, 6, 35].

Unfortunately, close scrutiny shows that there are a number of critical issues with the proof of one-more-unforgeability in AO and in some other subsequent works. In particular, the analysis of the reduction's success probability is based on a problematic counting argument. In this paper, we revisit the AO partially blind signature scheme and present a new comprehensive analysis of its one-more-unforgeability, which addresses *all issues in the original security proof.* (The proof of partial blindness in AO is correct and is not the focus of this paper.) The contributions of this paper are two-fold. First, we identify the flaws in the proof of AO, which we elaborate on in Section 1.1. Second, we overcome these issues by resorting to a more involved and rigorous counting argument. Our insights lead to new proof techniques and a much better understanding of AO's ideas. While we focus on the AO partially blind signature scheme, we believe that our techniques are applicable to other blind signature schemes based on the OR-proof technique.

1.1 Technical Overview

In this section we provide an overview of our security proof of the AO partially blind signature scheme, and explain the issues in the original work [4]. Similar to AO, our proof is done in two steps. First we consider the simplified case where there is only a single tag. This is the most technically involved part of the entire security analysis, and contains essential modifications to the proof in AO. Then we generalize it to the multi-tag case. This part of the proof is straightforward and mostly follows [4]. For simplicity, we only discuss the case of a single tag in this technical overview.

Forking: A Recap. The reduction in our security proof uses the forking technique to rewind the adversary and solve the discrete logarithm problem [31]. As is standard in a forking argument, we first define what we call a *deterministic wrapper* which provides a simplified, non-interactive interface to the reduction. More precisely, the wrapper takes as input an *instance* $\mathbf{I}$ (containing a public key and the internal values used to generate the signer's first messages of all signing sessions), a *random tape* rand (containing the random tape of the actual adversary), and a random *hash vector* $\overrightarrow{h}$ (to be used as outputs of random oracle queries). The reduction forks the wrapper instead of forking the adversary directly. In more concrete terms, this means that the reduction runs the wrapper once on inputs $\mathbf{I}, \mathsf{rand}, \overrightarrow{h}$ and obtains an output which implicitly defines an index $J \in [|\overrightarrow{h}|]$. It then generates a vector $\overrightarrow{h}'$ by resampling the vector $\overrightarrow{h}$ uniformly at random from position J, and keeping the first $J-1$ entries the same. It reruns the wrapper on inputs $\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'$, which will generate a run that is identical up the point where the reduction answers the J-th random oracle query. In particular, the input to this query remains identical in both runs. The goal of the reduction is to infer some equality from these relations so as to solve a discrete logarithm instance that it suitably embeds in its interaction with the adversary (see below).

Dealing with OR-Proofs in Forking. The AO scheme uses the classical OR-proof strategy of [15] to combine two Schnorr-style signatures into one. The witness for one branch of the proof is the actual secret key x of the scheme; the other branch corresponds to the *tag key* z which is obtained through hashing the tag info. On the signer's side, the protocol is a witness indistinguishable (WI) proof of knowledge of at least one witness, either the secret key x or the discrete logarithm of the tag key $\operatorname{dlog} \mathsf{z}$. This gives rise to the following proof strategy, which was also used in [6]: The reduction can choose these tag keys such that it knows a witness and sign without knowing the secret key (so it can embed a discrete logarithm challenge in the public key), or it can embed its discrete logarithm challenge in a tag key and sign using the actual secret key. The intuitive idea here is that for each run of the protocol, the witness used internally by the reduction is perfectly hidden from the adversary (due to WI). Thus, the probability that the reduction is able to extract the "opposing" witness (i.e., the one it is not using itself for answering signing queries) from two forking runs of the adversary should be high.

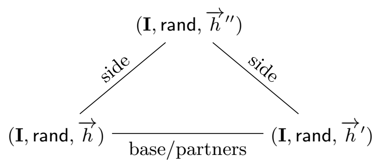
Unfortunately, this intuition proves incorrect upon closer inspection. While WI perfectly hides the witness during any *single* run of the protocol, the transcripts of two executions of the protocol with the adversary (as performed by the reduction) can depend on the witness internally used by the reduction. Therefore, arguing that the reduction indeed extracts the opposing witness from two runs of the adversary turns out to be highly non-trivial.

Partnering Runs. We now describe the general idea for proving that the reduction has a significant probability to extract the witness it needs. For now we fix an instance $\mathbf{I}$ and a random tape rand, and consider the hash vector $\overrightarrow{h}$ as the only varying parameter of the reduction. Using a simple counting argument, one can show that for a significant portion of pairs $\mathbf{I}, \mathsf{rand}$, there must exist two hash vectors $\overrightarrow{h}, \overrightarrow{h}'$ that lead to the same transcript between the wrapper and the adversary when the wrapper is run on $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ or $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, respectively. Borrowing the terminology from [4], we refer to such triples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ as *partners*. The key observation is that the witness extracted from partnering runs is independent of which witness was used by the reduction as part of the instance $\mathbf{I}$, and thus the reduction has a significant probability of extracting the desired witness (i.e., the witness not used by the reduction).[4] Unfortunately, given $\mathbf{I}, \mathsf{rand}$, finding a pair of partners $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ might not be efficiently possible, as in general, only few of them may exist. Hence it requires an additional argument to ensure that the reduction produces forks from which the desired witness can be efficiently extracted.

From Partners to Triangles. The next step in our chain of reasoning is to apply the strategy of AO for "amplifying" the number of forking runs from which the desired witness can be extracted. Thus, analogous to AO, we define *triangles* as follows. The corners of a triangle will be three triples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$, which produce successful runs for the wrapper. In addition, $\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}''$ all share a common prefix of some $i-1$ entries and start to fork from each other at the i-th entry. The most important property of a triangle, however, is that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ be partnering runs, i.e., produce the same transcript for the wrapper. (AO refer to the pair of partnering runs as the "triangle base" and to the remaining pairs of triples as the "triangle sides".) We illustrate this in fig. 1. As observed by AO, if the forked runs corresponding to $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ yield the desired witness (i.e., the one not stored inside $\mathbf{I}$), then either of the forked runs $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ or $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ yield the same witness. Their key insight is that the number of triangles should be far greater than the number of triangle bases formed by partnering runs $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. Intuitively, this is the case because a *single pair* of triples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ can serve as the base in *many different* triangles.

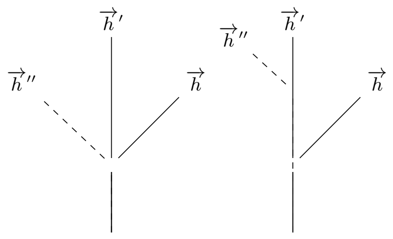
A Gap in AO. The next step in the analysis of AO is to count the number of triangles for which at least one side yields the desired witness. (We call such

[4] Due to the WI property of the scheme, for any $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, there exists a corresponding triple $(\mathbf{I}', \mathsf{rand}, \overrightarrow{h})$ that contains the other witness and produces the same transcript as $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. This means that the same witness w could have been extracted from a pair of partnering runs $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, or from $(\mathbf{I}', \mathsf{rand}, \overrightarrow{h})$, $(\mathbf{I}', \mathsf{rand}, \overrightarrow{h}')$, where one of $\mathbf{I}$ and $\mathbf{I}'$ contains w, and the other instance does not.



$(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$
side
side
$(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$
base/partners
$(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$

(a) A triangle consists of a pair of partners (the base) and one additional tuple (the top). A pair consisting of the top and one of the base corners is called a side.



(b) Left: forking as in a triangle (solid lines are the base, dashed lines are the top); right: not a triangle (forking at wrong point).

Fig. 1: Triangles

triangle sides "successful".) Recall that we keep $\mathbf{I}, \mathsf{rand}$ fixed throughout this counting argument, and argue only about the number of successful hash vectors associated with runs using $\mathbf{I}, \mathsf{rand}$. If we can show that there are enough of triangles with a successful side, we might hope that when sampling a random pair $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ during forking, the reduction will hit a successful triangle side, from which the desired witness can be extracted.

This is the point where our analysis diverges significantly from [4]. As noted above, many triangles may share a base; that is, for any given base, there exist many possible triangle tops. This makes it possible to "amplify" the extractability of the desired witness from a single base to extracting it from many possible triangle sides which are adjacent to this base in some triangle. (Recall that if a triangle base is successful, then at least one of the two sides must also be successful.) However, we observe that *many triangles may also share a side.* If many triangles overlap on successful sides (but not on unsuccessful sides), it might happen that the total number of successful sides is *much smaller* than the total number of unsuccessful sides.[5]

Indeed, this is where the most crucial gap occurs in [4]. First, for each triangle base corner $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, they assign this corner a partner $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ using the mapping Prt (so $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') = Prt(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ forms a triangle base together with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$; see [4, p. 284]). It is, however, unclear if this is intended to be

[5] We stress that simply replacing a triple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ with an indistinguishable triple $(\mathbf{I}', \mathsf{rand}, \overrightarrow{h})$ is not sufficient to solve this problem. Indeed, one might hope that since the adversary can not detect this change, an unsuccessful side may become successful when switching from $\mathbf{I}$ to $\mathbf{I}'$, as the desired witness would flip. However, a successful forking pair $((\mathbf{I}', \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}', \mathsf{rand}, \overrightarrow{h}'))$ need only exist if $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'))$ is a base. The same is not true, in general, for sides, as their endpoints may not (and generally do not) yield the same transcript. Because of this, an unsuccessful side $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'))$ might not even be part of a triangle side when switching witnesses from $\mathbf{I}$ to $\mathbf{I}'$.

an injective assignment (i.e., no two base corners can share the same partner). If so, there is a gap as to why this assignment is possible, i.e., why there are enough such partners for each of the base corners to find a *different* partner. In fact, we provide an argument in our analysis for why such partners — which we call *opposing base corners* — are *not much fewer* than original base corners, but they do not necessarily need to be equal in size.

On the other hand, if the assignment *Prt* is not injective, then different triangles may share the same side. This is also problematic, as we explain now. [4] proceeds to claim that if (for a fixed pair $\mathbf{I}, \mathsf{rand}$) at least $\frac{4}{5}$ of triangle sides are unsuccessful, then at least $\frac{3}{5}$ of triangle bases are also unsuccessful, i.e., they yield the undesired witness that is used by the reduction. (See the proof of the last claim on [4, p. 284].) Although this claim is not explicitly argued, the underlying reasoning seems to be as follows: since every triangle has two sides and one base, if $\frac{4}{5}$ of all sides are unsuccessful, then at least $\frac{4}{5}+\frac{4}{5}-1=\frac{3}{5}$ fraction of triangles have two unsuccessful sides, which implies that their bases must also be unsuccessful. However, this argument *implicitly assumes that no triangles ever share a base or a side*, which, as we have mentioned, is not necessarily the case.

Concrete Counterexamples and Additional Issues. We now provide concrete counterexamples to show why the claim above is false if triangles may share sides, or even just bases. For triangles sharing sides, consider the example in the middle of fig. 2, where 8 out of the 10 triangle sides are unsuccessful, yet only 2 out of the 6 triangle bases are unsuccessful. For triangles sharing only bases (recall that in this case there is already a gap as to why there exists an assignment *Prt* such that triangles do not share sides), the claim is also untrue: see the rightmost part of fig. 2 for an example where 5 out of the 6 triangle sides are unsuccessful, yet only 1 out of the 2 triangle bases are unsuccessful.

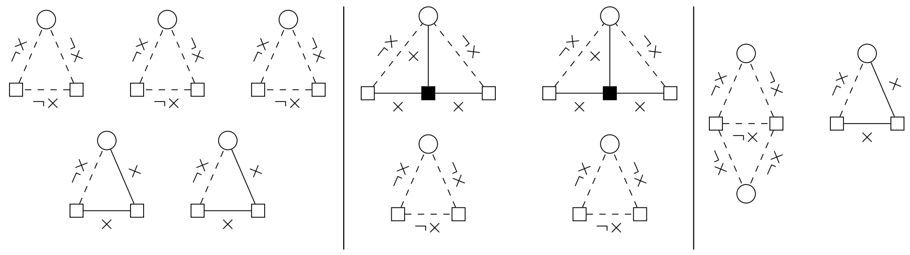


Fig. 2: Claim in [4] that if at least $\frac{4}{5}$ of triangle sides are unsuccessful (i.e., yield the undesirable witness $\neg\times$), then at least $\frac{3}{5}$ of bases (incident to two square nodes) also yield this witness. This holds for non-overlapping triangles (left), but not for triangles overlapping in sides (middle) or in bases (right).

We further note that there are some relatively minor gaps that are easier to fix. In particular, it is incorrect to assert that the probability to extract either witness from a triangle base is close to $\frac{1}{2}$ — we refer here to the last sentence in the proof of the last claim on [4, p. 284]:

> *Since the information of a base,* $(\overrightarrow{\varepsilon}, \overrightarrow{\varepsilon}')$*[*$(\overrightarrow{h}, \overrightarrow{h}')$*], is independent of the witness the simulator already has as a part of* Ω *[*$\mathbf{I}, \mathsf{rand}$*], this contradicts that a biased result should occur with probability (over* Ω *[*$\mathbf{I}, \mathsf{rand}$*]) less than* $1/2 + 1/\mathsf{poly}(n)$ *for any polynomial* poly.

(The expressions in brackets are a translation to our notation.)

To see why this claim is incorrect, imagine a computationally unbounded adversary that finds the secret key $x = \operatorname{dlog} \mathbf{y}$ by brute force, and wins the OMUF game by running the real signer's code. Then the signatures produced by this adversary — including pairs of signatures obtained from triangle bases — will always yield the same witness (the secret key x), rather than yielding either witness with probability close to $\frac{1}{2}$. Our approach to deal with this issue is to define a "majority witness" $\times$ which can be extracted from many triangle bases (for a suitable definition of many). We then show that it is possible to extract $\times$, using a suitable counting argument.

Resolving the issues from earlier works. We now provide an overview of our strategy to bridge the gaps in [4], achieving the same result. We first recall that for any triple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, there is a corresponding instance $\mathbf{I}'$ that contains the other branch of witness, such that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}', \mathsf{rand}, \overrightarrow{h})$ yield the same transcript. This naturally leads to the concept of *both-sided triangle bases*, namely triangle bases $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'))$ that are also the base of some triangle when $\mathbf{I}$ is replaced by $\mathbf{I}'$. Using several counting arguments, we show that the set of both-sided base corners must be large. While our counting arguments are more detailed and rigorous, they are in the same spirit as those of [4].

We now bridge the gap in [4], by showing that there cannot be too large of an overlap between triangle sides such that the absolute amount of successful triangle sides would get small. We define *good base corners* as triples that are incident to many successful both-sided triangle bases, as well as many successful triangle sides. We further require that these triangle sides and bases must exist at the good base corner's *maximum branching index* — the index at which the largest number of partners fork from it. Similarly, we define *good opposing corners* that are incident to a successful both-sided triangle base and many successful sides, but the triangle base and sides are located at the maximum branching index of the triple *at the other end of the base.*

Our crucial observation is that *if there are not too many good base corners, then there must be many good opposing corners.* To see this, consider a base corner $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ that is not good, and consider all triples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ that are partners of $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. Let i denote the maximum branching index of $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$; by definition, a significant portion of these partners $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ fork with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index i. Recall that if a triangle base is successful, then at least one of its sides must also be successful. Since most of the triangle sides involving $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are unsuccessful at index i, this means that many of the *other* triangle sides, i.e., those involving the partners $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, are successful at index i. In other words, a significant portion of a non-good base corner $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$'s partners, are good opposing corners. (In the formal proof, we will also rule out the possibility that different non-good base corners' corresponding good opposing corners overlap too much.)

The above conclusion means that, when the reduction samples the triple for the first forking run, with significant probability the triple is either a good base corner or a good opposing corner. Then, due to the definitions of these good triangle corners, it is not hard to show that with significant probability the reduction hits a successful triangle side while sampling the second triple — that is, the desired witness can be extracted from the two forking runs.

Finally, we remark that our reduction guesses in advance which hash values the adversary will actually use to produce its signatures. This introduces a loss of $\binom{Q_h}{\ell+1}$ in the reduction's advantage, where Q_h is the number of the adversary's hash queries and ℓ is the number of signing sessions closed. This step is necessary in our analysis as we need all possible forking indices to have a signature attached to them in order to lower-bound the set of good opposing base corners. (See Remark 2 in Section 4.4.) We notice that a loss in this order of magnitude seems inherent due to the recent polynomial-time ROS-attack [7], and that we achieve comparable bounds to the original work of Abe & Okamoto [4].[6]

1.2 Related Work

Partially blind signatures were introduced in [2], which also presented a scheme based on a non-standard RSA-type assumption. Cao *et al.* [10] proposed another construction based on the RSA assumption, but their scheme was cryptoanalyzed in [27]. Zhang *et al.* [38], as well as Chow *et al.* [14], proposed schemes based on bilinear pairings, and Papachristoudis *et al.* [29] proposed a scheme based on lattice assumptions. Okamoto [28] proposed a theoretical construction that does not rely on the random oracle model. Finally, Maitland & Boyd [26] considered a *restrictive* partially blind signature scheme, where the user's choice of messages must follow certain rules.

There is a rich literature on (ordinary) blind signatures and their applications. Its security notion was formalized by Pointcheval & Stern [30] and Juels *et al.* [22], and later strengthened by Schröder & Unruh [34]. Fischlin [16] and Abe & Ohkubo [3] considered security definitions in the universal composability (UC) framework. Camenisch *et al.* [8] and Fischlin & Schröder [17] considered a stronger notion of blindness called *selective-failure blindness*. There are a large number of blind signature schemes based on various assumptions and in various models; a very incomplete list includes [1, 9, 20, 21, 25, 28, 30, 32, 33].

We notice that the security analyses of (partially) blind signature schemes are usually extremely involved, with the original security proofs sometimes be-

[6] See the top of [4, p. 285], where the reduction's advantage includes a term η_1^2, where $\eta_1 = \eta/2Q_h^{\ell+1}$ and η is the adversary's advantage.

ing flawed. Apart from the schemes already discussed, we give two additional examples here. The security of the Schnorr blind signature [33] relies on the hardness of the ROS problem, which was recently shown to be easy [7]; a new security proof in the weaker sequential setting appears in [23]. For partially blind signature schemes, the aforementioned Zhang *et al.* scheme [38] has an issue in its security proof, and the full analysis came much later [36]. This paper can be seen as yet another attempt of spotting and fixing issues in previous works; however, we stress that the underlying OR-proof technique of the Abe-Okamoto scheme is widely used in blind signatures, and we believe that our techniques will find applications in the security analyses of other schemes as well.

2 Preliminaries

2.1 Notation

We denote by $[\ell] \coloneqq \{1, \ldots, \ell\}$. For a vector $\overrightarrow{h}$, its i-th entry is denoted by h_i, and the vector of its first i entries is denoted by $\overrightarrow{h}_{[i]}$. We denote by $x \xleftarrow{\$} X$ that x is sampled uniformly at random from set X. For a vector $\overrightarrow{x} \in X^n$, we denote by $\overrightarrow{x}' \xleftarrow{\$} X^n_{|\overrightarrow{x}_{[i]}}$ that $\overrightarrow{x}'$ is sampled uniformly at random from $\{\overrightarrow{x}' \in X^n | \overrightarrow{x}'_{[i]} = \overrightarrow{x}_{[i]}\}$. For an algorithm A, we use t_{A} to denote its running time.

2.2 Computational Problems

Definition 1 (Discrete Logarithm Problem). *For public parameters* $\mathsf{pp} = (\mathbb{G}, q, \mathbf{g})$ *for a group* $\mathbb{G}$ *with order* q *and generator* $\mathbf{g}$*, we describe the discrete logarithm game* $\mathbf{DLOG}_{\mathbb{G}}$ *with adversary* A *as follows:*

Setup. *Sample* $x \xleftarrow{\$} \mathbb{Z}_q$ *and set* $\mathbf{y} := \mathbf{g}^x$*. Output* $(\mathsf{pp}, \mathbf{y})$ *to* A*.*

Output Determination. *When* A *outputs* $x' \in \mathbb{Z}_q$*, return* 1 *if* $g^{x'} = \mathbf{y}$ *and* 0 *otherwise.*

We define the advantage of A *as*

$$\mathrm{adv}_{\mathsf{A}}^{\mathbf{DLOG}_{\mathbb{G}}} := \Pr[\mathbf{DLOG}_{\mathbb{G}}^{\mathsf{A}} = 1]$$

where the probability goes over the randomness of the game as well as the randomness of the adversary A*. We say that the discrete logarithm problem is* (t, ϵ)-hard *in* $\mathbb{G}$ *if for any adversary* A *that runs in time at most* t*, it holds that*

$$\mathrm{adv}_{\mathsf{A}}^{\mathbf{DLOG}_{\mathbb{G}}} \leq \epsilon.$$

(When it is clear from context, we may omit $\mathbb{G}$ *and only write* $\mathbf{DLOG}$ *for the game.)*

2.3 Partially Blind Signatures

The definitions in this section mostly follow [4].

Definition 2 (Partially Blind Signature scheme). *A three-move* partially blind signature scheme $\mathsf{PBS} = (\mathsf{KeyGen}, \mathsf{Sign} = (\mathsf{Sign}_1, \mathsf{Sign}_2), \mathsf{User} = (\mathsf{User}_1, \mathsf{User}_2), \mathsf{Verify})$ *consists of the following* PPT *algorithms:*

Key Generation. *On input public parameters* pp*, the probabilistic algorithm* KeyGen *outputs a public key* pk *and a secret key* sk*. Henceforth we assume that* pp *is provided to all parties (including the adversary) as an input, and do not explicitly write it.*

Signer: *The interactive signer* $\mathsf{Sign} = (\mathsf{Sign}_1, \mathsf{Sign}_2)$ *has two phases:*

- Sign_1**:** *On input a tag* info *and a secret key* sk*, the probabilistic algorithm* Sign_1 *outputs an internal signer state* $\mathsf{st}_{\mathsf{Sign}}$*, and a response* R*.*
- Sign_2**:** *On input the secret key* sk*, a challenge value* e*, and the corresponding internal state* $\mathsf{st}_{\mathsf{Sign}}$*, the deterministic algorithm* Sign_2 *outputs a response* S*.*

User. *The interactive user* $\mathsf{User} = (\mathsf{User}_1, \mathsf{User}_2)$ *has two phases:*

- User_1**:** *On input a public key* pk*, a tag* info*, a message* m*, and a* Sign_1 *response* R*, the probabilistic algorithm* User_1 *outputs a challenge value* e *and an internal user state* $\mathsf{st}_{\mathsf{User}}$*.*
- User_2**:** *On input a public key* pk*, a* Sign_2 *response* S*, and the corresponding internal user state* $\mathsf{st}_{\mathsf{User}}$*, the deterministic algorithm* User_2 *outputs a signature* sig *on message* m *along with the tag* info*.*

Verification. *On input a public key* pk*, a message* m*, a signature* sig*, and a tag* info*, the deterministic algorithm* Verify *outputs either* 1 *or* 0*, where* 1 *indicates that the signature is valid, and* 0 *that it is not.*

We say a partially blind signature scheme PBS *is* (perfectly) correct *if for all* $\mathsf{pk}, m, \mathsf{sig}, \mathsf{info}$ *that result from an honest interaction between signer and user,* $\mathsf{Verify}(\mathsf{pk}, m, \mathsf{sig}, \mathsf{info}) = 1$*.*

We now define the one-more-unforgeability of a partially blind signature scheme. We do not focus on partial blindness in this paper; we include the definition in the full version[24] for completeness, and for a proof that the Abe-Okamoto scheme is partially blind, see the original paper [4].

Definition 3 (One-more-unforgeability). *For a three-move partially blind signature scheme* PBS*, we define the* ℓ*-one more unforgeability (*ℓ*-OMUF) game* ℓ-$\mathbf{OMUF}_{\mathsf{PBS}}$ *with an adversary* U *(in the role of the user) as follows:*

Setup. *Sample a pair of keys* $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{PBS.KeyGen}(\mathsf{pp})$*. Initialize* $\ell_{\mathtt{closed}} := 0$ *and run* U *on input* pk*.*

Online Phase. U *is given access to oracles* sign_1 *and* sign_2*, which behave as follows.*

- **Oracle** sign_1**:** *On input* info*, the oracle samples a fresh session identifier* sid*. It sets* $\mathsf{open}_{\mathsf{sid}} := \mathtt{true}$ *and generates* $(R_{\mathsf{sid}}, \mathsf{st}_{\mathsf{sid}}) \xleftarrow{\$} \mathsf{PBS.Sign}_1(\mathsf{sk}, \mathsf{info})$*. Then it returns the response* R_{sid} *together with* sid *to* U*.*

Oracle sign_2**:** *If* $\ell_{\texttt{closed}} < \ell$*, the oracle takes as input a challenge* e *and a session identifier* sid*. If* $\mathsf{open}_{\mathsf{sid}} = \texttt{false}$*, it returns* $\perp$*. Otherwise, it sets* $\ell_{\texttt{closed}}{+}{+}$ *and* $\mathsf{open}_{\mathsf{sid}} := \texttt{false}$*. Then it computes the response* $S \xleftarrow{\$} \mathsf{PBS.Sign}_2(\mathsf{sk}, \mathsf{st}_{\mathsf{sid}}, e)$ *and returns* S *to* U*.*

Output Determination. *When* U *outputs distinct tuples* $(m_1, \mathsf{sig}_1, \mathsf{info}_1), \ldots, (m_k, \mathsf{sig}_k, \mathsf{info}_k)$*, return* 1 *if* $k \geq \ell_{\texttt{closed}} + 1$ *and for all* $i \in [k] : \mathsf{PBS.Verify}(\mathsf{pk}, \sigma_i, m_i, \mathsf{info}_i) = 1$*. Otherwise, return* 0*.*

We define the advantage of U *as*

$$\mathrm{adv}_{\mathsf{U}}^{\ell\text{-}\mathbf{OMUF}_{\mathsf{PBS}}} = \Pr\left[\ell\text{-}\mathbf{OMUF}_{\mathsf{PBS}}^{\mathsf{U}} = 1\right]$$

where the probability goes over the randomness of the game as well as the randomness of the adversary U*. We say the scheme* PBS *is* (t, ϵ, ℓ)-one-more unforgeable *if for any adversary* U *that runs in time at most* t *and makes at most* ℓ *queries to* sign_2*,*

$$\mathrm{adv}_{\mathsf{U}}^{\ell\text{-}\mathbf{OMUF}_{\mathsf{PBS}}} \leq \epsilon.$$

If U *always queries the same tag to oracle* sign_1*, we denote the game as* ℓ-1-info-$\mathbf{OMUF}_{\mathsf{PBS}}$ *and say that* PBS *is* (t, ϵ, ℓ)-single-tag one-more unforgeable*.*

3 The Abe-Okamoto Partially Blind Signature Scheme

In this section we describe the partially blind signature scheme by Abe & Okamoto [4]. The idea of the scheme relies on the OR-Proof technique by Cramer *et al.* [15]. It runs a proof of knowledge that the signer knows either the secret key x or the discrete logarithm of the so-called *tag key* $\mathbf{z}$, which is obtained through hashing the tag info. In this way we obtain a *witness indistinguishable* scheme: an honest signer does not know $\operatorname{dlog} \mathbf{z}$ and is forced to use x for issuing signatures; while the reduction may program the random oracle so that it knows the $\operatorname{dlog} \mathbf{z}$ and can then simulate the signer without knowing the secret key x.

Key Generation. On input public parameters $\mathsf{pp} = (\mathbb{G}, \mathbf{g}, q, \mathsf{H}^*, \mathsf{H})$ (where H^* and H are random oracles with ranges $\mathbb{G}$ and $\mathbb{Z}_q$, respectively), KeyGen samples $x \xleftarrow{\$} \mathbb{Z}_q$ and sets $\mathbf{y} := \mathbf{g}^x$. It then outputs $(\mathsf{pk}, \mathsf{sk}) := (\mathbf{y}, x)$.

Signer. $\mathsf{Sign} = (\mathsf{Sign}_1, \mathsf{Sign}_2)$ behaves as follows:

Sign_1**:** On input info and sk, Sign_1 computes the tag key $\mathbf{z} := \mathsf{H}^*(\mathsf{info})$ and samples $u, s, d \xleftarrow{\$} \mathbb{Z}_q$. It then computes the *commitments* $\mathbf{a} := \mathbf{g}^u, \mathbf{b} := \mathbf{g}^s \cdot \mathbf{z}^d$. It outputs the response $(\mathbf{a}, \mathbf{b})$ to the user and an internal state $\mathsf{st}_{\mathsf{Sign}} := (u, s, d)$.

Sign_2**:** On input $e \in \mathbb{Z}_q, \mathsf{st}_{\mathsf{Sign}} = (u, s, d), \mathsf{sk} = x$, Sign_2 computes $c := e - d$ and $r := u - cx$. It outputs the response (r, c, s, d) to the user.

User. $\mathsf{User} = (\mathsf{User}_1, \mathsf{User}_2)$ behaves as follows:

User_1**:** On input $\mathsf{pk}, m, \mathsf{info}, \mathbf{a}, \mathbf{b}$, User_1 computes the tag key $\mathbf{z} := \mathsf{H}^*(\mathsf{info})$ and samples $t_1, t_2, t_3, t_4 \xleftarrow{\$} \mathbb{Z}_q$. It then computes $\boldsymbol{\alpha} := \mathbf{g}^{t_1} \cdot \mathbf{y}^{t_2} \cdot \mathbf{a}$ and $\boldsymbol{\beta} := \mathbf{g}^{t_3} \cdot \mathbf{z}^{t_4} \cdot \mathbf{b}$, queries $h := \mathsf{H}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{z}, m)$ for the message m it wants to sign, and computes the blinded challenge $e := h - t_2 - t_4$. It outputs e to the signer and an internal state $\mathsf{st}_{\mathsf{User}} := (t_1, t_2, t_3, t_4)$.

User_2**:** On input $\mathsf{pk}, (r, c, s, d), \mathsf{st}_{\mathsf{User}} = (t_1, t_2, t_3, t_4)$, User_1 computes $\rho := r + t_1$, $\omega := c + t_2$, $\sigma := s + t_3$, and $\delta := d + t_4$. It then verifies that $\omega + \delta = \mathsf{H}(\mathbf{g}^\rho \cdot \mathbf{y}^\omega, \mathbf{g}^\sigma \cdot \mathbf{z}^\delta, \mathbf{z}, m)$; if so, it outputs the signature $(\rho, \omega, \sigma, \delta)$. (Otherwise, it outputs $\perp$.)

Verification. On input $\mathbf{y}, m, \mathsf{info}, (\rho, \omega, \sigma, \delta)$, Verify computes $\mathbf{z} := \mathsf{H}^*(\mathsf{info})$. It outputs 1 if $\omega + \delta = \mathsf{H}(\mathbf{g}^\rho \cdot \mathbf{y}^\omega, \mathbf{g}^\sigma \cdot \mathbf{z}^\delta, \mathbf{z}, m)$ and 0 otherwise.

For a graphic illustration of the scheme, see the full version[24].

4 Computing the probability for extracting the "good" witness

As mentioned in the introduction, our analysis of the Abe-Okamoto scheme is done in two steps. In this section, we deal with the case that the adversary U only uses a *single* tag, i.e., U plays the ℓ-1-info-$\mathbf{OMUF}_{\mathsf{AO}}$ game.

4.1 The Deterministic OMUF Wrapper

Restricting the adversary to making $\ell + 1$ hash queries. Suppose that the adversary U makes ℓ queries to sign_2 (henceforth "signing queries") and Q_h queries to H (henceforth "hash queries"), and uses a single tag $\overline{\mathsf{info}}$. Below we assume w.l.o.g. that U never makes the same query to H twice.

We say that a message-signature pair $(m, (\rho, \omega, \sigma, \delta))$ *corresponds to* an index $i \in [Q_h]$, or corresponds to the adversary U's i-th hash query, if this query was $\mathsf{H}(\mathbf{y}^\omega \mathbf{g}^\rho, \mathbf{z}^\delta \mathbf{g}^\sigma, \mathbf{z}, m)$. (When the message m is clear from context, we may say that the signature $(\rho, \omega, \sigma, \delta)$ corresponds to index i.) We remark that we can further assume w.l.o.g. that there exist $\ell + 1$ hash queries of U, each of which corresponds to a distinct message-signature pair in the output of U (in particular, $Q_h \geq \ell + 1$). This is because otherwise one of the following must hold (assuming that U succeeds):

- There exists a pair $(m, (\omega, \rho, \delta, \sigma))$ that does not correspond to any hash query, i.e., $\mathsf{H}(\mathbf{y}^\omega \mathbf{g}^\rho, \mathbf{z}^\delta \mathbf{g}^\sigma, \mathbf{z}, m)$ has never been queried. In this case, U can be turned into another adversary U' that runs the code of U and additionally makes such a hash query; obviously U and U' have the same advantage.
- There exist two distinct pairs $(m_1, (\omega_1, \rho_1, \delta_1, \sigma_1)), (m_2, (\omega_2, \rho_2, \delta_2, \sigma_2))$ that correspond to the same hash query. In this case, we have that $m_1 = m_2$, $\mathbf{y}^{\omega_1} \mathbf{g}^{\rho_1} = \mathbf{y}^{\omega_2} \mathbf{g}^{\rho_2}$, and $\mathbf{z}^{\delta_1} \mathbf{g}^{\sigma_1} = \mathbf{z}^{\delta_2} \mathbf{g}^{\sigma_2}$. Then a reduction to the discrete logarithm problem can easily compute both x and w as $x = (\omega_1 - \omega_2)^{-1} \cdot (\rho_2 - \rho_2)$ and $w = (\delta_1 - \delta_2)^{-1} \cdot (\sigma_2 - \sigma_1)$.

It is not hard to see that any adversary U can be turned into another adversary that makes *exactly* $\ell + 1$ hash queries, with a factor of $\binom{Q_h}{\ell+1}$ loss in advantage. Formally, we define an adversary $\mathsf{M} := \mathsf{M}^{\mathsf{U}}$ that works as follows. M, on input of a public key pk, chooses a random subset I of $[Q_h]$ with $|I| = \ell + 1$, and invokes $\mathsf{U}(\mathsf{pk})$. For U's i-th query to H, if $i \notin I$, M responds with a random integer in $\mathbb{Z}_q$. For any other query (including queries to signing oracles, queries to H^*, and the i-th query to H for $i \in I$), M forwards it to the corresponding oracle of M's own challenger, and forwards the response back to U. When U outputs a set of $\ell+1$ message-signature pairs, M checks if every pair $(m, (\rho, \omega, \sigma, \delta))$ corresponds to some index $i \in I$, that is, U's i-th hash query was $\mathsf{H}(\mathbf{y}^{\omega}\mathbf{g}^{\rho}, \mathbf{z}^{\delta}\mathbf{g}^{\sigma}, \mathbf{z}, m)$. If so, M copies U's output (and outputs $\perp$ otherwise).

Lemma 1. *For* M *described above, we have that*

$$\mathrm{adv}_{\mathsf{M}}^{\ell\text{-}1\text{-}\mathsf{info}\text{-}\mathbf{OMUF}_{\mathsf{AO}}} \geq \frac{\mathrm{adv}_{\mathsf{U}}^{\ell\text{-}1\text{-}\mathsf{info}\text{-}\mathbf{OMUF}_{\mathsf{AO}}}}{\binom{Q_h}{\ell+1}}.$$

Proof. It is straightforward that M simulates the OMUF game to U perfectly. Assume that U succeeds. By our assumption on U, there is a set of indices $I^* \subset [Q_h]$ corresponding to the message-signature pairs in U's output, with $|I^*| = \ell + 1$. If $I^* = I$, then M also succeeds. Since I is a random subset of size $\ell + 1$ of $[Q_h]$, the probability that $I^* = I$ is $\frac{1}{\binom{Q_h}{\ell+1}}$. The lemma follows. □

The lemma above implies that it is sufficient to consider an adversary that makes exactly $\ell + 1$ (distinct) hash queries, since an upper bound of the adversary's advantage in this specific case immediately translates to such an upper bound in the general case. Below we simply assume that the adversary makes $\ell + 1$ hash queries.

The deterministic wrapper. For any adversary M that makes exactly $\ell + 1$ distinct hash queries, we define a deterministic *wrapper* A that, given the witness and random coin tosses for one side, simulates the view of M. The wrapper uses either the $\mathbf{y}$-side witness (i.e., the secret key) x or the $\mathbf{z}$-side witness $w = \operatorname{dlog} \mathbf{z}$ to respond to sign_2 queries, and simulates the other side of the OR-proof using fixed values. We begin with the formal definition of an instance:

Definition 4 (Instances). *For the deterministic wrapper simulating the OMUF-game to the adversary we define two types of* instances $\mathbf{I}$. *A* $\mathbf{y}$-side *(a.k.a. honest)* instance *consists of the following components:*

$b = 0$**:** *bit indicating that the secret key x will be used for simulation*
x**:** *the secret key, also referred to as the $\mathbf{y}$-side witness*
$\mathbf{z}$**:** *the tag key, to be returned by oracle* H^* *for requested* $\overline{\mathsf{info}}$
d_i, s_i**:** *simulator choices for $\mathbf{z}$-side part corresponding to the i-th signing session*
u_i**:** *discrete logarithm of the $\mathbf{y}$-side commitment $\mathbf{a}_i$ in the i-th signing session*

A $\mathbf{z}$-side instance *consists of the following components:*

$b = 1$**:** *bit indicating that the tag witness w will be used for simulation*
$\mathbf{y}$**:** *the public key*
w**:** *the discrete logarithm of the tag key $\mathbf{z}$ as above*
c_i, r_i**:** *simulator choices for $\mathbf{y}$-side part corresponding to the i-th signing session*

v_i**:** *discrete logarithm of the* $\mathbf{z}$*-side commitment* $\mathbf{b}_i$ *in the* i*-th signing session*

Let $\overrightarrow{h}$ be the vector of responses returned by random oracle H (so $\left|\overrightarrow{h}\right| = \ell+1$), rand be the randomness used by the adversary M, and $\overline{\mathsf{info}}$ be the tag used in the OMUF game. We define a deterministic wrapper $\mathsf{A} := \mathsf{A}^{\mathsf{M}}_{\overline{\mathsf{info}}}$ that runs on $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ as shown in Figure 3. The wrapper allows us to argue about which $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ tuples cause the adversary to succeed.

A has two simulation modes. For $b = 0$, it runs the honest signer's algorithm to simulate both sign_1 and sign_2 oracle queries; for H^* queries, it responds with $\mathbf{z}$ if the input is $\overline{\mathsf{info}}$ and $\perp$ for all other inputs. In mode $b = 1$, A knows w and not x. It therefore runs the so-called $\mathbf{z}$-side signer (see the full version[24]), which is the honest signer's algorithm except that w is treated as the secret key. A responds to queries to H^* with $\mathbf{g}^w$ for $\overline{\mathsf{info}}$ and $\perp$ otherwise. In both modes, A responds to queries to H using entries in the hash vector $\overrightarrow{h}$. Finally, upon receiving M's output message-signature pairs, A checks if they are all valid, and if so, A copies M's output (and outputs $\perp$ otherwise).

It is easy to see that

$$t_{\mathsf{A}} = t_{\mathsf{M}} + \mathrm{O}(\ell) = t_{\mathsf{U}} + \mathrm{O}(\ell) + \mathrm{O}({Q_h}^2) = t_{\mathsf{U}} + \mathrm{O}(\ell) + \mathrm{O}({Q_h}^2),$$

where the term $\mathrm{O}(\ell)$ comes from verifying $\ell + 1$ signatures, and $\mathrm{O}({Q_h}^2)$ comes from identifying the hash indices that correspond to signatures.

```
A(I, rand, $\overrightarrow{h}$)
parse $b$ from $\mathbf{I}$
if $b = 0$
    parse $(b, x, \mathbf{z}, \overrightarrow{d}, \overrightarrow{s}, \overrightarrow{u}) := \mathbf{I}$
    $\mathsf{pk} := \mathbf{g}^x$
else
    parse $(b, \mathbf{y}, w, \overrightarrow{c}, \overrightarrow{r}, \overrightarrow{v}) := \mathbf{I}$
    $\mathsf{pk} := \mathbf{y}$
$\mathsf{sid} := 0$
$j := 0$
$(m_i, (\rho_i, \omega_i, \sigma_i, \delta_i))_{i=1}^{\ell+1} := \mathsf{M}^{\mathsf{sign}_1, \mathsf{sign}_2, \mathsf{H}, \mathsf{H}^*}(\mathsf{pk}; \mathsf{rand})$
if $\forall i$: $\mathsf{Verify}(\mathsf{pk}, m_i, (\rho_i, \omega_i, \sigma_i, \delta_i))$
    return $(m_i, (\rho_i, \omega_i, \sigma_i, \delta_i))_{i=1}^{\ell+1}$
else
    return $\perp$

H($\xi$)
$j$++
return $h_j$

H*(info)
if $\mathsf{info} = \overline{\mathsf{info}}$
    if $b = 0$ return $\mathbf{z}$
    else return $\mathbf{g}^w$
else return $\perp$

sign_1(info)
if $\mathsf{info} = \overline{\mathsf{info}}$
    sid++
    $\mathsf{open}(\mathsf{sid}) :=$ true
    if $b = 0$
        $\mathbf{a}_{\mathsf{sid}} := \mathbf{g}^{u_{\mathsf{sid}}}$
        $\mathbf{b}_{\mathsf{sid}} := \mathbf{g}^{s_{\mathsf{sid}}} \cdot \mathbf{z}^{d_{\mathsf{sid}}}$
    else
        $\mathbf{a}_{\mathsf{sid}} := \mathbf{g}^{r_{\mathsf{sid}}} \cdot \mathbf{y}^{c_{\mathsf{sid}}}$
        $\mathbf{b}_{\mathsf{sid}} := \mathbf{g}^{v_{\mathsf{sid}}}$
    return $(\mathsf{sid}, \mathbf{a}_{\mathsf{sid}}, \mathbf{b}_{\mathsf{sid}})$
else return $\perp$

sign_2(sid, $e_{\mathsf{sid}}$)
if $\mathsf{open}(\mathsf{sid})$
    if $b = 0$
        $c_{\mathsf{sid}} := e_{\mathsf{sid}} - d_{\mathsf{sid}}$
        $r_{\mathsf{sid}} := u_{\mathsf{sid}} - c_{\mathsf{sid}} \cdot x$
    else
        $d_{\mathsf{sid}} := e_{\mathsf{sid}} - c_{\mathsf{sid}}$
        $s_{\mathsf{sid}} := v_{\mathsf{sid}} - d_{\mathsf{sid}} \cdot w$
else
    return $\perp$
$\mathsf{open}(\mathsf{sid}) :=$ false
return $(c_{\mathsf{sid}}, r_{\mathsf{sid}}, d_{\mathsf{sid}}, s_{\mathsf{sid}})$
```

Fig. 3: Wrapper A that simulates the OMUF game to the adversary M

The set of successful tuples. Let

$$\mathsf{Succ} := \{(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) | \mathsf{A}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \neq \bot\}$$

be the set of all "successful" input tuples to the wrapper A. For a pair of instance and randomness $\mathbf{I}, \mathsf{rand}$, it is also useful to define $\mathsf{Succ}_{\mathbf{I},\mathsf{rand}}$ as the set of successful input tuples with instance $\mathbf{I}$ and randomness rand, i.e.,

$$\mathsf{Succ}_{\mathbf{I},\mathsf{rand}} := \left\{ (\mathbf{I}', \mathsf{rand}', \overrightarrow{h}) \in \mathsf{Succ} \middle| \begin{matrix} \mathbf{I}' = \mathbf{I} \\ \mathsf{rand}' = \mathsf{rand} \end{matrix} \right\}.$$

In the following we further denote by $\mathcal{I}$ the set of all possible instances, by $\mathcal{R}$ the set of all possible randomness of A, and by ϵ the success probability of A, i.e.,

$$\epsilon := \frac{|\mathsf{Succ}|}{\left|\mathcal{I} \times \mathcal{R} \times \mathbb{Z}_q^{\ell+1}\right|}$$

We show in Lemma 2 below (in Section 4.3) that the simulation using the $\mathbf{z}$-side witness is perfectly indistinguishable from the real execution where the $\mathbf{y}$-side witness is used (this is called the *witness indistinguishability* of the scheme), i.e., A simulates the OMUF game to M perfectly. Furthermore, if M succeeds, then so does A, since A copies M's output in this case (see lines 10–11 of Figure 3). Therefore,

$$\epsilon = \mathrm{adv}_{\mathsf{M}}^{\ell\text{-}1\text{-}\mathsf{info}\text{-}\mathbf{OMUF}_{\mathsf{AO}}}.$$

4.2 Basic Definitions

We first define some concepts related to the wrapper A's input tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, that will be used throughout the security proof.

Transcripts. We begin with the definition of the *query transcript*, which consists of the adversary's signing queries:

Definition 5 (Query Transcript). *Consider the wrapper* A *running on input tuple* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$*. The* query transcript*, denoted* $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$*, is the vector of queries* e_{sid} *made to the* sign_2 *oracle (simulated by* A*) by the adversary* M*, ordered by* sid*.*

Next, we define (full) interaction *transcripts* between adversary M and wrapper A. These contain, in addition to $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, also M's sign_1 queries and the signatures from the output of M. This will be useful to argue about A's behavior on different inputs $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. Looking ahead, we will see that it is possible to deterministically transform $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ into a dual input $\Phi_{\mathsf{rand},\overrightarrow{h}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ that results in the same behavior as $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ (i.e., produces the same full transcript as $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$), but inverts the type of the witness $\mathbf{I}$ from $\mathbf{y}$-side to $\mathbf{z}$-side (or vice-versa).

Definition 6 (Full Transcripts). *Consider the wrapper* A *running on input tuple* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. *We denote by* $\mathsf{tr}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *the transcript produced between* A *and the adversary* M*, i.e., all messages sent between the user (played by* M*) and the signer (played by* A*). Concretely,*

$$\mathsf{tr}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \Big(\mathsf{info}, (\overrightarrow{\mathbf{a}}, \overrightarrow{\mathbf{b}}), \overrightarrow{e}, (\overrightarrow{c}, \overrightarrow{r}, \overrightarrow{d}, \overrightarrow{s}), \mathsf{sig}_1, \ldots \mathsf{sig}_{\ell+1}\Big),$$

where $\mathsf{sig}_1, \ldots, \mathsf{sig}_{\ell+1}$ *are the signatures output by* M. *(If* M *aborts at any point during the protocol or outputs fewer than* $\ell + 1$ *signatures, we consider any undefined entry to be* $\perp$*.)*

Forking, partners, and triangles. We next define what it means for two input tuples to *fork* successfully — this corresponds to all cases where the reduction would be able to compute at least one of the two witnesses from the resulting signatures. However, without further work, the witness that can be computed might be the one that the reduction already knows.

Definition 7 (Successful forking). *We say two successful input tuples* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in \mathsf{Succ}$ fork *from each other at index* $i \in [\ell+1]$ *if* $\overrightarrow{h}_{[i-1]} = \overrightarrow{h}'_{[i-1]}$ *but* $h_i \neq h_i$. *We denote the set of hash vector pairs* $(\overrightarrow{h}, \overrightarrow{h}')$ *such that* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$*,* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *fork at index* i *as* $F_i(\mathbf{I}, \mathsf{rand})$.

We now define *partners*, which will play a key role in our analysis. Informally, two tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are partners at some index i if they fork from this index and produce the same query transcript (but not necessarily the same full transcript).

Definition 8 (Partners). *We say two (successful) tuples* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *are* partners *at index* $i \in [\ell + 1]$ *if the followings hold:*

- $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *and* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *fork at index* i
- $\overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$

We denote the set of $(\overrightarrow{h}, \overrightarrow{h}')$ *such that* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *and* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *are partners at index* i *by* $\mathrm{prt}_i(\mathbf{I}, \mathsf{rand})$. *We further denote by* $P_{\mathbf{I},\mathsf{rand}}$ *the following set:*

$$P_{\mathbf{I},\mathsf{rand}} = \Big\{(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}_{\mathbf{I},\mathsf{rand}} \Big| \exists \overrightarrow{h}', i \in [\ell + 1] \colon (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\mathbf{I}, \mathsf{rand})\Big\}$$

We define *triangles* in order to extend the nice properties of partners to more general forking tuples. Informally, a triangle consists of three vectors $\overrightarrow{h}, \overrightarrow{h'}, \overrightarrow{h''}$ which all fork from each other at the same index, and also have the property that $\overrightarrow{h}$ and $\overrightarrow{h'}$ are partners at this index. This way, it is natural to view these vectors as corners of the triangle and any pair of two vectors as the sides.

Definition 9 (Triangles). *A* triangle *at index $i \in [\ell+1]$ with respect to* $\mathbf{I}, \mathsf{rand}$ *is a tuple of three (successful) tuples in the following set:*

$$\triangle_i(\mathbf{I}, \mathsf{rand}) = \left\{ \begin{array}{l|l} ((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), & (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\mathbf{I}, \mathsf{rand}) \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), & (\overrightarrow{h}, \overrightarrow{h}'') \in F_i(\mathbf{I}, \mathsf{rand}) \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')) & (\overrightarrow{h}', \overrightarrow{h}'') \in F_i(\mathbf{I}, \mathsf{rand}) \end{array} \right\}$$

For a triangle $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')) \in \triangle_i(\mathbf{I}, \mathsf{rand})$*, we call the pair of tuples* $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'))$ *the* base*, and* $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}''))$ *and* $((\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}''))$ *the* sides*. We further refer to the tuples* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$*,* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$*,* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ *as* corners*, where the two corners incident to the base are called* base corners*, and the third corner is called the* top*. We will sometimes write* $(\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand})$ *for compactness.*

Maximum branching index and set. In the following we define two important characteristics of partner tuples. We begin by defining the *maximum branching index*, which is the index at which a partner tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in P_{\mathbf{I},\mathsf{rand}}$ has *the most partners.*

Definition 10 (Maximum Branching Index). *Fix a pair* $\mathbf{I}, \mathsf{rand}$*. The* maximum branching index *of a partner tuple* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in P_{\mathbf{I},\mathsf{rand}}$ *is the index at which* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *has the most partners, i.e.,*

$$\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \mathrm{argmax}_{i \in [\ell+1]} \left| \left\{ \overrightarrow{h}' \middle| (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\mathbf{I}, \mathsf{rand}) \right\} \right| .$$

In case of ties, we pick the lowest such index.

The maximum branching index naturally defines a partition of any non-empty set of partnered tuples $P_{\mathbf{I},\mathsf{rand}}$, where the i-th set of the partition contains all tuples with maximum branching index i. We define the *maximum branching set* as the largest part of this partition, i.e., the largest subset of tuples that share a common maximum branching index.

Definition 11 (Maximum Branching Set). *For a pair* $\mathbf{I}, \mathsf{rand}$*, consider the partition of partner tuples according to their maximal branching indices:*

$$B_i(\mathbf{I}, \mathsf{rand}) = \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \middle| \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = i \right\} .$$

The maximum branching set *of* $\mathbf{I}, \mathsf{rand}$ *is defined as the largest set among them, i.e.,*

$$B_{max}(\mathbf{I}, \mathsf{rand}) = B_{i_{max}(\mathbf{I}, \mathsf{rand})}(\mathbf{I}, \mathsf{rand}),$$

where

$$i_{max}(\mathbf{I}, \mathsf{rand}) = \mathrm{argmax}_{i \in [\ell+1]} \left| B_i(\mathbf{I}, \mathsf{rand}) \right| .$$

In case of ties, we pick the lowest such index.

Note in particular that $B_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}(\mathbf{I}, \mathsf{rand})$ (henceforth $B_{\mathrm{Br}_{\max}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ for simplicity) is the set of all tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ which have the same maximum branching index as $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ (so $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in B_{\mathrm{Br}_{\max}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$).

4.3 The Mapping Φ

For any successful tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, we now define the mapping $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ and prove its transcript preserving properties in Lemma 2. We remark that this mapping is not efficiently computable and will merely serve as a technical tool in our analysis.

Definition 12 (Mapping instances via transcript). *For* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$*, we define* $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I})$ *as follows. For a* $\mathbf{y}$*-side instance* $\mathbf{I} = (1, w, \mathbf{y}, \overrightarrow{c}, \overrightarrow{r}, \overrightarrow{u})$*,* $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I})$ *is a* $\mathbf{z}$*-side instance that consists of*

$$b = 0 \qquad x = \mathrm{dlog}\, \mathbf{y} \qquad \mathbf{z} = \mathbf{g}^w \qquad \forall i \in [\ell]\colon d_i = e_i - c_i$$
$$\forall i \in [\ell]\colon s_i = u_i - d_i \cdot w \qquad \forall i \in [\ell]\colon v_i = c_i \cdot x + r_i$$

For a $\mathbf{z}$*-side instance* $\mathbf{I} = (0, x, \mathbf{z}, d, s, v)$*,* $\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I})$ *is a* $\mathbf{y}$*-side instance that consists of*

$$b = 1 \qquad w = \mathrm{dlog}\, \mathbf{z} \qquad \mathbf{y} = \mathbf{g}^x \qquad \forall i \in [\ell]\colon c_i = e_i - d_i$$
$$\forall i \in [\ell]\colon r_i = v_i - c_i \cdot x \qquad \forall i \in [\ell]\colon u_i = d_i \cdot w + s_i$$

(where $\overrightarrow{e}$ *is the query vector produced by* $\mathsf{rand}, \overrightarrow{h}$ *using instance* $\mathbf{I}$*). We will sometimes use the notation* $\Phi_{\overrightarrow{e}}$ *instead of* $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ *for a given* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$*. We also define* $\Phi(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = (\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$*.*

Lemma 2 ($\Phi_{\mathsf{rand}, \overrightarrow{h}}$ is a bijection that preserves transcripts). *Fix* $\mathsf{rand}, \overrightarrow{h}$*. For all tuples* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$*,* $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ *is a self-inverse bijection and*

$$\mathsf{tr}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \mathsf{tr}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$$

The proof is deferred to the full version[24].

The lemma above shows that the Abe-Okamoto scheme is *witness indistinguishable*, i.e., a simulator that uses the $\mathbf{z}$-side witness to sign (see the full version[24]) creates a view identical to the real view to the adversary. In particular, this implies that the wrapper A simulates the ℓ-OMUF game to the adversary M perfectly.

Corollary 1. $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ} \Leftrightarrow (\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h}) \in \mathsf{Succ}$.

We look into the effect of the transcript mapping function on partner tuples. We have proven that $\Phi_{\mathsf{rand}, \overrightarrow{h}}$ preserves the transcript (and hence success) of $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. However, note that this does not (by itself) imply that partnering tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ result in partnering tuples $(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$ and $(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h}')$, or $(\Phi_{\mathsf{rand}, \overrightarrow{h}'}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$ and $(\Phi_{\mathsf{rand}, \overrightarrow{h}'}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h}')$, respectively. Lemma 3 asserts that this is indeed the case.

Lemma 3 (Partners stay partners through Φ). *For all* $\mathbf{I}, \mathsf{rand}$*, and vectors* $\overrightarrow{h}, \overrightarrow{h}'$*,*

$$\begin{aligned}(\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\mathbf{I}, \mathsf{rand}) &\Leftrightarrow (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}) \\ &\Leftrightarrow (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_i(\Phi_{\mathsf{rand}, \overrightarrow{h}'}(\mathbf{I}), \mathsf{rand})\end{aligned}$$

We refer the reader to the full version for the proof.

Corollary 2. $\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \mathrm{Br}_{\max}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$.

4.4 Extracting a Witness from a Fork

Witness Extraction. We briefly recall how the reduction can compute a witness from two signatures from forking runs of the wrapper A. We say a signature $(\rho, \omega, \sigma, \delta)$ on a message m in the output of A on input $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *corresponds to* a hash value h_i, if $\mathsf{H}(\mathbf{g}^{\rho}\mathbf{y}^{\omega}, \mathbf{g}^{\sigma}\mathbf{z}^{\delta}, \mathbf{z}, m)$ was the i-th hash query made to the random oracle H in this run of A. Informally we say that a witness *can be extracted* from $\mathbf{I}, \mathsf{rand}$, and a pair of forking hash vectors $(\overrightarrow{h}, \overrightarrow{h}') \in F_i(\mathbf{I}, \mathsf{rand})$, if it can be efficiently computed from the two signatures corresponding to h_i and h'_i. We make this formal in the following definition.

Definition 13 (Witness Extraction). *Fix* $\mathbf{I}, \mathsf{rand}$ *and let* $(\overrightarrow{h}, \overrightarrow{h}') \in F_i(\mathbf{I}, \mathsf{rand})$ *for some* $i \in [\ell + 1]$*. Moreover, denote* $\mathsf{sig}_i, \mathsf{sig}'_i$ *the signatures that correspond to* h_i *and* h'_i*, respectively. Consider the two witness extraction algorithms* $\mathsf{E}_{\mathbf{y}}, \mathsf{E}_{\mathbf{z}}$ *as described in Figure 4. For* $\times \in \{\mathbf{y}, \mathbf{z}\}$*, we say that the* $\times$*-side witness* can be extracted from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *at index* i *if* $\mathsf{E}_{\times}$ *on input* $(\mathsf{sig}_i, \mathsf{sig}'_i)$ *does not return* $\perp$.

Lemma 4. *Let* $\mathbf{I}, \mathsf{rand}, i, (\overrightarrow{h}, \overrightarrow{h}') \in F_i(\mathbf{I}, \mathsf{rand}), \mathsf{sig}_i, \mathsf{sig}'_i$*, and algorithms* $\mathsf{E}_{\mathbf{y}}, \mathsf{E}_{\mathbf{z}}$ *be as in Definition 13. Then at least one of* $\mathsf{E}_{\mathbf{y}}$ *and* $\mathsf{E}_{\mathbf{z}}$ *outputs a correct witness on input the two signatures* $\mathsf{sig}_i = (\rho_i, \omega_i, \sigma_i, \delta_i)$ *and* $\mathsf{sig}'_i = (\rho'_i, \omega'_i, \sigma'_i, \delta'_i)$ *corresponding to* h_i *and* h'_i*. More specifically,* $\mathsf{E}_{\mathbf{y}}$ *outputs the* $\mathbf{y}$*-side witness if and only if* $\omega_i \neq \omega'_i$*, otherwise* $\mathsf{E}_{\mathbf{z}}$ *outputs the* $\mathbf{z}$*-side witness.*

The proof is a standard forking argument and is deferred to the full version[24].

Remark 1. We note that the witness may be contained in the instance $\mathbf{I}$, in which case the witness can be trivially extracted. For the purposes of the lemma we only consider the more interesting case that the witness can be computed from the two signatures directly, regardless of which witness was used for simulating the signing oracles.

```
$\mathsf{E}_{\mathbf{y}}((\rho_i, \omega_i, \sigma_i, \delta_i), (\rho'_i, \omega'_i, \sigma'_i, \delta'_i))$
if $(\omega_i \neq \omega'_i)$
    return $x := \frac{\rho_i - \rho'_i}{\omega'_i - \omega_i}$
else
    return $\perp$
```

```
$\mathsf{E}_{\mathbf{z}}((\rho_i, \omega_i, \sigma_i, \delta_i), (\rho'_i, \omega'_i, \sigma'_i, \delta'_i))$
if $(\delta_i \neq \delta'_i)$
    return $w := \frac{\sigma_i - \sigma'_i}{\delta'_i - \delta_i}$
else
    return $\perp$
```

Fig. 4: The two witness extraction algorithms from Definition 13

Witnesses in triangles. We now show that if a witness can be extracted from the base of a triangle, it can also be extracted from at least one of the sides. This was previously shown in [4].

Corollary 3. *Fix* $\mathbf{I}, \mathsf{rand}$ *and let* $(\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand})$ *for some* $i \in [\ell + 1]$. *Moreover, suppose that the* $\mathbf{y}$*-side witness can be extracted from the base* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ *of the triangle at index* i. *Then the* $\mathbf{y}$*-side witness can also be extracted from at least one of the sides* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ *or* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ *at index* i. *An analogous statement holds for the* $\mathbf{z}$*-side witness.*

Proof. Toward a contradiction, suppose that the $\mathbf{y}$-side witness can be extracted from the base $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index i, but can not be extracted at index i for either of the sides $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ or $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$. Then, by Lemma 4, $\omega_i = \omega''_i$ and $\omega'_i = \omega''_i$, so $\omega_i = \omega'_i$. By Lemma 4 again, the $\mathbf{y}$-side witness can not be extracted from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, a contradiction. An analogous argument can be made for the $\mathbf{z}$-side. □

We now define *both-sided triangle base corners* as triangle base corners $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ which remain base corners of some triangle at their maximal branching index when mapped via $\Phi_{\mathsf{rand}, \overrightarrow{h}}$. (Recall that by Corollary 2, the maximal branching index is preserved under Φ.) On top of this, if $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ is a both-sided triangle base corner, and forms a triangle base with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index $\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, then $(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}, \overrightarrow{h})$ and $(\Phi_{\mathsf{rand}, \overrightarrow{h}}, \mathsf{rand}, \overrightarrow{h}')$ also form a triangle base.

For every such tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$, we further define the set $D_i^{\mathbf{y}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ of tuples that form a both-sided triangle base with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ at index i from which the $\mathbf{y}$-side witness can be extracted, and an analogous set $D_i^{\mathbf{z}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ for the $\mathbf{z}$-side witness. This allows us to then define sets $B_T^{\mathbf{y}}$ and $B_T^{\mathbf{z}}$ that contain tuples where the majority of both-sided triangle bases incident to the tuple allow for extraction of the $\mathbf{y}$-side or $\mathbf{z}$-side witness, respectively.

Definition 14 (Both-sided Triangle Base Corners). *We call elements of the set*

$$B_T := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \;\middle|\; \begin{array}{c} \exists \overrightarrow{h}', \\ \overrightarrow{h}'', \overrightarrow{h}''' \end{array} : \begin{array}{c} (\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}(\mathbf{I}, \mathsf{rand}) \\ (\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}''') \in \triangle_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}) \end{array} \right\}$$

both-sided triangle base corners*. For any index $i \in [\ell + 1]$, we define sets*

$$D_i^{\mathbf{y}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \;\middle|\; \begin{array}{c} \exists \overrightarrow{h}'', \\ \overrightarrow{h}''' \end{array} : \begin{array}{c} (\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand}) \\ (\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}''') \in \triangle_i(\Phi_{\mathsf{rand}, \overrightarrow{h}}(\mathbf{I}), \mathsf{rand}) \\ \text{The } \mathbf{y}\text{-side witness can be} \\ \text{extracted from } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \text{ at index } i \end{array} \right\}$$

and $B_T^{\mathbf{y}} \subset B_T$ *as*

$$B_T^{\mathbf{y}} := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \;\middle|\; \begin{array}{l} D^{\mathbf{y}}_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \neq \emptyset \\ \left| D^{\mathbf{y}}_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \right| \\ \geq \left| D^{\mathbf{z}}_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \right| \end{array} \right\}$$

We define sets $D_i^{\mathbf{z}}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ *and* $B_T^{\mathbf{z}}$ *analogously.*

Lemma 5 (Both-sided triangle bases produce the same witness on both sides).

1. $\Phi(B_T^{\mathbf{y}}) = B_T^{\mathbf{y}}$ *and* $\Phi(B_T^{\mathbf{z}}) = B_T^{\mathbf{z}}$*;*
2. $B_T^{\mathbf{y}} \cup B_T^{\mathbf{z}} = B_T$.

We defer the proof to the full version[24].

We define $B_T^{\times}$ as the larger set of $B_T^{\mathbf{y}}$ and $B_T^{\mathbf{z}}$. By the second item of Lemma 5, $\left|B_T^{\times}\right| \geq \frac{1}{2}\left|B_T\right|$.

Let $B_{T,\mathbf{y}}^{\times}$ (resp. $B_{T,\mathbf{z}}^{\times}$) be the subset of $B_T^{\times}$ with $\mathbf{y}$-side instances (resp. $\mathbf{z}$-side instances). We stress that $B_T^{\mathbf{y}}$ and $B_{T,\mathbf{y}}^{\times}$ are two different sets: $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in B_T^{\mathbf{y}}$ means that more both-sided triangle bases (with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ as one of its corners) allow for extracting the $\mathbf{y}$-side witness than the $\mathbf{z}$-side witness; whereas $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in B_{T,\mathbf{y}}^{\times}$ means that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in B_T^{\times}$ and $\mathbf{I}$ is a $\mathbf{y}$-side witness.

Lemma 6. $\left|B_{T,\mathbf{y}}^{\times}\right| = \left|B_{T,\mathbf{z}}^{\times}\right| = \frac{1}{2}\left|B_T^{\times}\right|$.

Proof. By the first item of Lemma 5, Φ is a bijection within $B_T^{\times}$, and since Φ maps a tuple with a $\mathbf{y}$-side instance to a tuple with a $\mathbf{z}$-side instance (and vice versa), we know that Φ is a bijection between $B_{T,\mathbf{y}}^{\times}$ and $B_{T,\mathbf{z}}^{\times}$; therefore, $\left|B_{T,\mathbf{y}}^{\times}\right| = \left|B_{T,\mathbf{z}}^{\times}\right|$. Since $B_{T,\mathbf{y}}^{\times}$ and $B_{T,\mathbf{z}}^{\times}$ form a partition of $B_T^{\times}$, we know that $\left|B_{T,\mathbf{y}}^{\times}\right| + \left|B_{T,\mathbf{z}}^{\times}\right| = \left|B_T^{\times}\right|$, and the lemma follows. □

We now give a lower bound of the size of $B_T^\times$. Let $\epsilon_{B_T^\times}$ be the probability of getting a tuple in $B_T^\times$ while sampling uniformly at random, i.e.,

$$\epsilon_{B_T^\times} := \frac{\left|B_T^\times\right|}{\left|\mathcal{I} \times \mathcal{R} \times \mathbb{Z}_q{}^{\ell+1}\right|}.$$

Lemma 7 (Lowerbounding the size of $B_T^\times$). *Assume* $\epsilon \geq \frac{432\left(1-\frac{1}{(\ell+1)^2}\right)}{q}$. *Then*

$$\epsilon_{B_T^\times} \geq \frac{\epsilon}{96}.$$

We defer the proof to the full version[24].

Finding triangle tops. In order for our security proof to go through, a key step is to compute the probability that the reduction hits a triangle side from which the $\times$-side witness can be extracted when forking the wrapper, independently of the witness that is being used by the reduction. This event is crucial in our proof because, assuming that the reduction samples one of these sides, it is likely that it did so with the witness opposite of $\times$, meaning that it extracts the witness $\times$ it *does not already know* with significant probability, hence solving the discrete logarithm problem. In order to lower bound the probability of the event above, we first define *relevant triangle tops* for a both-sided triangle base corner $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in B_T^\times$. These are all the tuples $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$ such that $(\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'')$ forms triangles at index i (where $\overrightarrow{h}'$ is as in the definition of both-sided triangle tops (Definition 14)).

Definition 15 (Relevant triangle tops). *For a tuple* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$*, define its* relevant triangle tops *at index i as tuples in the following set:*

$$T_{T,i}^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'') \middle| \exists \overrightarrow{h}' : \begin{array}{c} (\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand}) \\ \text{The } \times \text{-side witness} \\ \text{can be extracted from } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \text{ at } i \end{array} \right\}$$

We will mostly consider relevant triangle tops at the maximum branching index $\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ and we thus define $T_T^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) := T_{T,\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})}^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$.

What remains to be shown is that many elements of $B_T^\times$ actually have many relevant triangle tops, regardless of whether they reside in $B_{T,\mathbf{y}}^\times$ or $B_{T,\mathbf{z}}^\times$, i.e., independently of the witness that they store. This ensures that when the reduction samples and then (partially) resamples the vectors during the forking process, it will hit a side from which the desired witness can be extracted with significant probability, as explained above.

Lemma 8 (There are enough relevant triangle tops). *There exists a subset $G_{\mathbf{y}} \subset B^{\times}_{T,\mathbf{y}}$ with $|G_{\mathbf{y}}| \geq \frac{3}{8}\left|B^{\times}_{T,\mathbf{y}}\right|$ such that for each $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in G_{\mathbf{y}}$,*

$$\left|T^{\times}_{T}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| \geq \frac{\epsilon_{B^{\times}_{T}}}{16(\ell+1)} \cdot q^{\ell-\mathrm{Br}_{\max}(\mathbf{I},\mathsf{rand},\overrightarrow{h})+2} - 2q^{\ell-\mathrm{Br}_{\max}(\mathbf{I},\mathsf{rand},\overrightarrow{h})+1}.$$

An analogous statement holds for $B^{\times}_{T,\mathbf{z}}$.

The proof is deferred to the full version[24].

Corollary 4. *Let $G_{\mathbf{y}}$ be as in Lemma 8. Then*

$$\Pr_{\substack{(\mathbf{I},\mathsf{rand},\overrightarrow{h}) \xleftarrow{\$} \mathcal{I}\times\mathcal{R}\times\mathbb{Z}_q{}^{\ell+1} \\ i \xleftarrow{\$} [\ell+1],\, \overrightarrow{h}' \xleftarrow{\$} \mathbb{Z}_q{}^{\ell+1}_{|\overrightarrow{h}_{[i-1]}}}} \left[(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in T^{\times}_{T}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \;\middle|\; \begin{array}{c} (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in G_{\mathbf{y}} \\ \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = i \end{array} \right]$$
$$\geq \frac{\epsilon_{B^{\times}_{T}}}{16(\ell+1)} - \frac{2}{q}.$$

An analogous statement holds for $G_{\mathbf{z}}$.

Proof. Suppose $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in G_{\mathbf{y}}$ and $\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = i$. Note that $\left|\mathbb{Z}_q{}^{\ell+1}_{|\overrightarrow{h}_{[i-1]}}\right| = q^{\ell-i+2}$. Therefore, the probability of sampling an $\overrightarrow{h}'$ such that $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in T^{\times}_{T}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ is

$$\frac{\left|T^{\times}_{T}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right|}{q^{\ell-i+2}} \geq \frac{\frac{\epsilon_{B^{\times}_{T}}}{16(\ell+1)} \cdot q^{\ell-i+2} - 2q^{\ell-i+1}}{q^{\ell-i+2}} = \frac{\epsilon_{B^{\times}_{T}}}{16(\ell+1)} - \frac{2}{q}.$$

□

Opposing base corners. By Corollary 3 we know that each triangle with a relevant base has at least one relevant side. We now want to consider the probability of finding such a relevant side in the forking proof.

To this end, we consider *opposing base corners* — corners of relevant bases whose partners are in $G_{\mathbf{y}}$ or $G_{\mathbf{z}}$. See the full version[24] for a graphic illustration. (Keep in mind that the sets $G_{\mathbf{y}}$ and $G_{\mathbf{z}}$ are the sets of both sided triangle base corners for which there exist many triangle tops.)

Definition 16 (Opposing base corners).

$$O^{\times}_{T} := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \;\middle|\; \exists \overrightarrow{h}' : \begin{array}{c} (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in G_{\mathbf{y}} \cup G_{\mathbf{z}} \\ (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_{\mathrm{Br}_{\max}(\mathbf{I},\mathsf{rand},\overrightarrow{h}')}(\mathbf{I}, \mathsf{rand}) \\ \text{the } \times \text{-side witness can be} \\ \text{extracted from } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \text{ at } \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \end{array} \right\}$$

Good corners with useful tops. For each tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ in $O_T^\times$ or $B_T^\times$ we define *useful triangle tops* — triangle tops that allow for extraction of the $\times$-side witness when combined with the base corner $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ (see the full version[24] for a graphic illustration):

Definition 17 (Useful triangle tops). *For any* $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in O_T^\times \cup B_T^\times$, *define*

$$A_{T,i}^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) := \left\{ \begin{array}{c|c} (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'') & \text{the } \times \text{-side witness can be} \\ \in T_{T,i}^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) & \text{extracted from } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), \\ & (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'') \text{ at index } i \end{array} \right\}$$

Recall that relevant base corners — those in $G_\mathbf{y}$ or $G_\mathbf{z}$ — are tuples in $B_T^\times$ for which many triangle tops are relevant (i.e., the corresponding $T_T^\times$ set is large). We now consider a subset of these relevant base corners for which a lot of the relevant triangle tops are useful (i.e., the corresponding $A_T^\times$ set is large). We call these base corners *good*.

Definition 18 (Good base corners). *We say that a base corner in* $G_\mathbf{y} \cup G_\mathbf{z}$ *is* good *if it lies within the following set:*

$$\widehat{B_T^\times} := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in G_\mathbf{y} \cup G_\mathbf{z} \;\middle|\; \begin{array}{c} \left|A_T^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| \\ \geq \frac{1}{2} \left|T_T^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| \\ -q^{\ell - \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) + 1} \end{array} \right\}$$

We now want to show that if the set of good base corners is small, then there exist a lot of opposing base corners — which we call *good opposing base corners* — that fulfill a property analogous to good base corners.

Definition 19 (Good opposing base corners).

$$\widehat{O_T^\times} := \left\{ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \;\middle|\; \exists \overrightarrow{h}': \begin{array}{c} (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in G_\mathbf{y} \cup G_\mathbf{z} \\ (\overrightarrow{h}, \overrightarrow{h}') \in \mathrm{prt}_{\mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')}(\mathbf{I}, \mathsf{rand}) \\ \text{the } \times \text{-side witness can be} \\ \text{extracted from } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \text{ at } \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \\ \left|A_{T, \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')}^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| \\ \geq \frac{1}{2} \left|T_{T, \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')}^\times(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| \\ -q^{\ell - \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') + 1} \end{array} \right\}$$

Let $\widehat{B_{T,\mathbf{y}}^\times} \subset \widehat{B_T^\times}$ and $\widehat{O_{T,\mathbf{y}}^\times} \subset \widehat{O_T^\times}$ be analogous to $B_{T,\mathbf{y}}^\times \subset B_T^\times$, i.e., the subset of tuples with $\mathbf{y}$-side instances. We define $\widehat{B_{T,\mathbf{z}}^\times}$ and $\widehat{O_{T,\mathbf{z}}^\times}$ similarly.

Lemma 9. *If* $\left|\widehat{B_{T,\mathbf{y}}^\times}\right| < \frac{1}{2} |G_\mathbf{y}|$, *then* $\left|\widehat{O_{T,\mathbf{y}}^\times}\right| \geq \frac{1}{8(\ell+1)} |G_\mathbf{y}|$. *An analogous statement holds for* $\mathbf{z}$.

Proof. Let $F = G_{\mathbf{y}} \setminus \widehat{B^{\times}_{T,\mathbf{y}}}$ (so $|F| \geq \frac{1}{2}|G_{\mathbf{y}}|$). Consider any $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\in F$, and let $i = \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. Then

$$\left|A^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| < \frac{1}{2}\left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| - q^{\ell-i+1}.$$

By Corollary 3, for any $(\overrightarrow{h}, \overrightarrow{h}', \overrightarrow{h}'') \in \triangle_i(\mathbf{I}, \mathsf{rand})$ such that the $\times$-side witness can be extracted from the base $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, if the $\times$-side witness cannot be extracted from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$, then it can be extracted from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}), (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}'')$. (All extractions mentioned above are at index i.) Therefore,

$$\left|A^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| + \left|A^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| \geq \left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right|.$$

We note that all but $q^{\ell-i+1}$ elements of $T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ are also elements of $T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. This is because $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}^*) \in T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \setminus T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ implies that $(\overrightarrow{h}, \overrightarrow{h}^*) \in F_i(\mathbf{I}, \mathsf{rand})$ but $(\overrightarrow{h}', \overrightarrow{h}^*) \notin F_i(\mathbf{I}, \mathsf{rand})$, which means that $\overrightarrow{h}^*$ must share its first i entries with $\overrightarrow{h}'$ (recall that $\overrightarrow{h}$ and $\overrightarrow{h}'$ share the first $i-1$ entries), so there are at most $q^{\ell-i+1}$ such vectors. We get that

$$\left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| \geq \left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| - q^{\ell-i+1}.$$

Combining all inequalities above, we get

$$\begin{aligned}
\left|A^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| &\geq \left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| - \left|A^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| \\
&> \left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| - \left(\frac{1}{2}\left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| - q^{\ell-i+1}\right) \\
&= \frac{1}{2}\left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| + q^{\ell-i+1} \\
&\geq \frac{1}{2}\left(\left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| - q^{\ell-i+1}\right) + q^{\ell-i+1} \\
&> \frac{1}{2}\left|T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})\right| - q^{\ell-i+1}
\end{aligned}$$

I.e., if $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in F$, then all of its partners $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ at index i with which it forms triangle bases from which the $\times$-side witness can be extracted, are in $\widehat{O^{\times}_{T,\mathbf{y}}}$.

We now lower-bound the number of such partners $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. Define the set of tuples that yield the same query transcript with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ as

$$E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') = \{(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}^{\star}) | \overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}^{\star}) = \overrightarrow{e}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\}.$$

Note that $E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ is the set of partners of $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ at any index. Consider a subset $E_i(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ of all tuples that fork from $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ at index

i, i.e., $E_i(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') = \{(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}^\star) | (\overrightarrow{h}^\star, \overrightarrow{h}') \in \mathrm{prt}_i(\mathbf{I}, \mathsf{rand})\}$. Recall that $i = \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. By the definition of maximum branching index, we have

$$\left|E_i(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| \geq \frac{1}{\ell+1}\left(\left|E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right| - 1\right) \geq \frac{1}{2(\ell+1)}\left|E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')\right|$$

(where the -1 comes from excluding $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ itself). As $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in B_T^\times$, it holds that at least half of the tuples in $E_i(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, together with $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$, allow for the extraction of the $\times$-side witness. This means that at least half of the tuples in $E_i(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are in $\widehat{O^\times_{T,\mathbf{y}}}$.

We have shown that for any $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in F$, at least $\frac{1}{4(\ell+1)}$ of tuples in $E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ are in $\widehat{O^\times_{T,\mathbf{y}}}$. Further note that for any $(\mathbf{I}_1, \mathsf{rand}_1, \overrightarrow{h}_1)$ and $(\mathbf{I}_2, \mathsf{rand}_2, \overrightarrow{h}_2)$, either $E(\mathbf{I}_1, \mathsf{rand}_1, \overrightarrow{h}_1) = E(\mathbf{I}_2, \mathsf{rand}_2, \overrightarrow{h}_2)$ or $E(\mathbf{I}_1, \mathsf{rand}_1, \overrightarrow{h}_1) \cap E(\mathbf{I}_2, \mathsf{rand}_2, \overrightarrow{h}_2) = \emptyset$. [7] Summing over all $E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$ for some $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in F$, we get

$$\begin{aligned}
\left|O_T^\times\right| &\geq \frac{1}{4(\ell+1)} \sum_{\substack{E \text{ s.t. } E = E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \\ \text{for some } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in F}} |E| \geq \frac{1}{4(\ell+1)} \sum_{\substack{E \text{ s.t. } E = E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \\ \text{for some } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in F}} |E \cap F| \\
&= \frac{1}{4(\ell+1)} \left| \bigcup_{\substack{E \text{ s.t. } E = E(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \\ \text{for some } (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') \in F}} (E \cap F) \right| \\
&= \frac{1}{4(\ell+1)} |F| \geq \frac{1}{8(\ell+1)} |G_{\mathbf{y}}|.
\end{aligned}$$

□

Remark 2. We point out that it is at this point that we need to require the adversary to make *exactly* $\ell+1$ hash queries (and thus lose a $\binom{Q_h}{\ell+1}$ factor in advantage). The proof of Lemma 9 would not go through with $Q_h > \ell+1$ hash queries, as hash vectors in this case may fork at arbitrary indices that do not have a corresponding signature. Therefore, not every tuple in an E-set would also be a partner of every other tuple in the same E-set (with the definition of partners adapted to this setting, i.e., two tuples can only be partners if they both have a signature at their forking index).

In the following, we want to avoid the case distinction of whether triangle corners come from the B-sets or the O-sets. We therefore define *good triangle corners*:

[7] This is because $E(\mathbf{I}_1, \mathsf{rand}_1, \overrightarrow{h}_1) \cap E(\mathbf{I}_2, \mathsf{rand}_2, \overrightarrow{h}_2) \neq \emptyset$ implies that $\mathbf{I}_1 = \mathbf{I}_2$, $\mathsf{rand}_1 = \mathsf{rand}_2$, and $\overrightarrow{e}(\mathbf{I}_1, \mathsf{rand}_1, \overrightarrow{h}_1) = \overrightarrow{e}(\mathbf{I}_2, \mathsf{rand}_2, \overrightarrow{h}_2)$, which in turn implies that $E(\mathbf{I}_1, \mathsf{rand}_1, \overrightarrow{h}_1) = E(\mathbf{I}_2, \mathsf{rand}_2, \overrightarrow{h}_2)$.

Definition 20. *Let $\widehat{G_{\mathbf{y}}}$ be the larger set of $\widehat{B^{\times}_{T,\mathbf{y}}}$ and $\widehat{O^{\times}_{T,\mathbf{y}}}$. Furthermore, for a tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \widehat{G_{\mathbf{y}}}$, let $t(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ be an index at which many relevant triangle tops exist, i.e.,*

$$t(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = \begin{cases} \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) & \textit{(if } \widehat{G_{\mathbf{y}}} = \widehat{B^{\times}_{T,\mathbf{y}}}\textit{)} \\ \mathrm{Br}_{\max}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}') & \textit{(if } \widehat{G_{\mathbf{y}}} = \widehat{O^{\times}_{T,\mathbf{y}}}\textit{)} \end{cases}$$

(where $\overrightarrow{h}'$ is as in the definition of $\widehat{O^{\times}_{T,\mathbf{y}}}$). If multiple such $\overrightarrow{h}'$ (and thus multiple choices for t) exist, choose one that results in the smallest value of t. Define set $\widehat{G_{\mathbf{z}}}$ analogously, and for a tuple $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \widehat{G_{\mathbf{z}}}$, define $t(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$ analogously.

It is easy to see that for a good opposing base corner, the number of triangle tops is the same as for the corresponding tuple from $G_{\mathbf{y}} \cup G_{\mathbf{z}}$. We state this as a lemma.

Lemma 10.

$$\Pr_{\substack{b \xleftarrow{\$} \{0,1\} \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \xleftarrow{\$} \mathcal{I}_b \times \mathcal{R} \times \mathbb{Z}_q{}^{\ell+1} \\ i \xleftarrow{\$} [\ell+1], \overrightarrow{h}' \xleftarrow{\$} \mathbb{Z}_q{}^{\ell+1}_{|\overrightarrow{h}_{[i-1]}}}} \left[\overrightarrow{h}' \in T^{\times}_{T,i}(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \,\middle|\, \begin{array}{c} (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \widehat{G_{\mathbf{y}}} \\ t(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) = i \end{array} \right] \geq \frac{\epsilon_{B^{\times}_T}}{16(\ell+1)} - \frac{2}{q}$$

An analogous statement holds for $\widehat{G_{\mathbf{z}}}$.

Proof. If $\widehat{G_{\mathbf{y}}} = \widehat{B^{\times}_{T,\mathbf{y}}}$, then the lower bound is implied by Corollary 4. If $\widehat{G_{\mathbf{y}}} = \widehat{O^{\times}_{T,\mathbf{y}}}$, setting the partner from the proof of Lemma 8 to the triangle corner from $\widehat{O^{\times}_{T,\mathbf{y}}}$ yields this lower bound. □

We furthermore note the following regarding the probability of sampling a tuple in $\widehat{G_{\mathbf{y}}}$ and $\widehat{G_{\mathbf{z}}}$:

Lemma 11.

$$\Pr_{\substack{b \xleftarrow{\$} \{0,1\} \\ (\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \xleftarrow{\$} \mathcal{I}_b \times \mathcal{R} \times \mathbb{Z}_q{}^{\ell+1} \\ i \xleftarrow{\$} [\ell+1], \overrightarrow{h}' \xleftarrow{\$} \mathbb{Z}_q{}^{\ell+1}_{|\overrightarrow{h}_{[i-1]}}}} \Pr\left[(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}) \in \widehat{G_{\mathbf{y}}} \right] \geq \frac{3}{128(\ell+1)} \epsilon_{B^{\times}_T}$$

The same holds for $\widehat{G_{\mathbf{z}}}$.

We defer the proof to the [24]. We will use the sets $\widehat{G_{\mathbf{y}}}$ and $\widehat{G_{\mathbf{z}}}$ for simplicity in the forking proof to avoid case distinctions over whether $\widehat{B^{\times}_{T,\mathbf{y}}}$ or $\widehat{O^{\times}_{T,\mathbf{y}}}$ (or $\widehat{B^{\times}_{T,\mathbf{z}}}$ or $\widehat{O^{\times}_{T,\mathbf{z}}}$) are larger.

4.5 Forking Proof for concurrent OMUF

In this section, we show that the Abe-Okamoto partially blind signature scheme AO is single-tag one-more unforgeable. We extend the proof to multiple tags in Section 4.6.

Theorem 1 (OMUF security for single-tag adversaries). *For all $\ell \in \mathbb{N}$, if there exists an adversary U that makes Q_h hash queries to random oracle H and $(t_{\mathsf{U}}, \epsilon_{\mathsf{U}}, \ell)$-breaks 1-$\mathsf{info}$-$\mathbf{OMUF}_{\mathsf{AO}}$ with $\epsilon_{\mathsf{U}} \geq \frac{432\left(1-\frac{1}{(\ell+1)^2}\right)}{q} \cdot \binom{Q_h}{\ell+1}$, then there exists an algorithm B that $\left(t_{\mathsf{B}} = 2t_{\mathsf{U}} + \mathrm{O}({Q_h}^2), \epsilon_{\mathsf{B}} \approx \frac{3\epsilon_{\mathsf{U}}^2}{75423744\cdot\binom{Q_h}{\ell+1}^2\cdot(\ell+1)^3}\right)$-breaks* **DLOG**.

Proof. We use the wrapper A as described in Figure 3. We now construct a reduction B that plays the **DLOG** game as follows.

After B receives its discrete logarithm challenge $\mathbf{U}$, it samples a bit $b \xleftarrow{\$} \{0,1\}$. It then samples an instance $\mathbf{I}$ of type b where it sets $\mathbf{z} := \mathbf{U}$ if $b = 0$ and $\mathbf{y} := \mathbf{U}$ if $b = 1$, and all other items uniformly at random from $\mathbb{Z}_q$. Furthermore, B samples a random tape rand for A and a random hash vector $\overrightarrow{h}$. After that, B runs A on $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h})$. If A returns a set of $\ell + 1$ valid message-signature pairs, B chooses a random index $i \xleftarrow{\$} [\ell + 1]$. B then re-samples the vector $\overrightarrow{h}' \xleftarrow{\$} {\mathbb{Z}_q}^{\ell+1}_{|\overrightarrow{h}[i-1]}$ and runs A on $(\mathbf{I}, \mathsf{rand}, \overrightarrow{h}')$. If A outputs a second set of $\ell + 1$ valid message-signature pairs, B identifies the signature matching the hash value h_i and h'_i respectively in both pair (it aborts if there exists no such signature for h'_i). Denote the corresponding signature components to the ith hash query by $\rho_i, \rho'_i, \omega_i, \omega'_i, \sigma_i, \sigma'_i, \delta_i, \delta'_i$ (see the full version[24]).

If $\omega_i \neq \omega'_i$ and $b = 1$, B computes

$$x := (\omega_i - \omega'_i)^{-1} \cdot (\rho'_i - \rho_i)$$

as its output; if $\delta_i \neq \delta'_i$ and $b = 0$, B computes

$$w := (\delta_i - \delta'_i)^{-1} \cdot (\sigma'_i - \sigma_i)$$

as its output. Otherwise B aborts. (If A fails to return a set of $\ell + 1$ valid message-signature pairs either time, B also aborts.)

B runs A twice, and performs $\Theta(\ell)$ additional computation (in particular, B verifies up to $2(\ell + 1)$ signatures). Plugging in $t_{\mathsf{A}} = t_{\mathsf{U}} + \mathrm{O}({Q_h}^2)$, we get that

$$t_{\mathsf{B}} = 2t_{\mathsf{U}} + \mathrm{O}({Q_h}^2).$$

We now analyze the advantage of reduction B. Let ϵ_{U} be the advantage of U in the OMUF game, and ϵ be the probability that A outputs $\ell + 1$ valid message-signature pairs. By Lemma 1 and subsequent analysis in Section 4.1,

$$\epsilon \geq \frac{\epsilon_{\mathsf{U}}}{\binom{Q_h}{\ell+1}}.$$

We can see that B internally runs the witness extracting algorithm $\mathsf{E}_{\mathbf{y}}$ or $\mathsf{E}_{\mathbf{z}}$ in Definition 13. Forking over the set $\widehat{G}$ of good base corners yields the theorem statement. We provide a detailed computation of the probability in the full version [24]. □

4.6 Extension to multiple tags

Theorem 2. *Let* U *be an adversary against* ℓ-$\mathbf{OMUF}_{\mathsf{AO}}$ *that runs in time* t_{U}*, closes at most* ℓ_{info} *signing sessions per tag* info*, closes at most* ℓ *signing sessions in total, and queries at most* Q_{info} *tags* info *to oracle* H^**. Let* $\mathrm{adv}^{\mathbf{OMUF}_{\mathsf{AO}}}_{Q_{\mathsf{info}},\ell_{\mathsf{info}},\mathsf{U}}$ *be* U*'s advantage. Then there exists a reduction* B *against* 1-info-$\mathbf{OMUF}_{\mathsf{AO}}$ *that runs in time* $t_{\mathsf{B}} \approx t_{\mathsf{U}}$ *and makes at most* ℓ_{info} *signing queries and has advantage*

$$\mathrm{adv}_{\mathsf{B}}^{\ell_{\mathsf{info}}\text{-}1\text{-}\mathsf{info}\text{-}\mathbf{OMUF}_{\mathsf{AO}}} \geq \frac{\mathrm{adv}^{\ell-\mathbf{OMUF}_{\mathsf{AO}}}_{Q_{\mathsf{info}},\ell_{\mathsf{info}},\mathsf{A}} - \frac{\ell}{q}}{Q_{\mathsf{info}}}.$$

The proof of this theorem mostly follows that in [4]. We provide it in the full version[24] for completeness.

References

1. Abe, M. *A Secure Three-Move Blind Signature Scheme for Polynomially Many Signatures* in *EUROCRYPT 2001* (2001).
2. Abe, M. & Fujisaki, E. *How to Date Blind Signatures* in *ASIACRYPT'96* (1996).
3. Abe, M. & Ohkubo, M. *A Framework for Universally Composable Non-committing Blind Signatures* in *ASIACRYPT 2009* (2009).
4. Abe, M. & Okamoto, T. *Provably Secure Partially Blind Signatures* in *CRYPTO 2000* (2000).
5. Alkadri, N. A., Harasser, P. & Janson, C. *BlindOR: An Efficient Lattice-Based Blind Signature Scheme from OR-Proofs* in *CANS 21* (2021).
6. Baldimtsi, F. & Lysyanskaya, A. *Anonymous credentials light* in *ACM CCS 2013* (2013).
7. Benhamouda, F., Lepoint, T., Loss, J., Orrù, M. & Raykova, M. *On the (in)security of ROS* in *EUROCRYPT 2021, Part I* (2021).
8. Camenisch, J., Neven, G. & shelat, a. *Simulatable Adaptive Oblivious Transfer* in *EUROCRYPT 2007* (2007).
9. Camenisch, J., Piveteau, J.-M. & Stadler, M. *Blind Signatures Based on the Discrete Logarithm Problem (Rump Session)* in *EUROCRYPT'94* (1995).
10. Cao, T., Lin, D. & Xue, R. A randomized RSA-based partially blind signature scheme for electronic cash. *Computers & Security* (2005).
11. Chaum, D. *Blind Signatures for Untraceable Payments* in *CRYPTO'82* (1982).
12. Chaum, D. *Elections with Unconditionally-Secret Ballots and Disruption Equivalent to Breaking RSA* in *EUROCRYPT'88* (1988).
13. Chaum, D., Fiat, A. & Naor, M. *Untraceable Electronic Cash* in *CRYPTO'88* (1990).
14. Chow, S. S. M., Hui, L. C. K., Yiu, S.-M. & Chow, K. P. *Two Improved Partially Blind Signature Schemes from Bilinear Pairings* in *ACISP 05* (2005).

15. Cramer, R., Damgård, I. & Schoenmakers, B. *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols* in *CRYPTO'94* (1994).
16. Fischlin, M. *Round-Optimal Composable Blind Signatures in the Common Reference String Model* in *CRYPTO 2006* (2006).
17. Fischlin, M. & Schröder, D. *Security of Blind Signatures under Aborts* in *PKC 2009* (2009).
18. Fujioka, A., Okamoto, T. & Ohta, K. *A Practical Secret Voting Scheme for Large Scale Elections* in *AUSCRYPT'92* (1993).
19. Hanatani, Y., Komano, Y., Ohta, K. & Kunihiro, N. *Provably Secure Electronic Cash Based on Blind Multisignature Schemes* in *FC 2006* (2006).
20. Hauck, E., Kiltz, E., Loss, J. & Nguyen, N. K. *Lattice-Based Blind Signatures, Revisited* in *CRYPTO 2020, Part II* (2020).
21. Hazay, C., Katz, J., Koo, C.-Y. & Lindell, Y. *Concurrently-Secure Blind Signatures Without Random Oracles or Setup Assumptions* in *TCC 2007* (2007).
22. Juels, A., Luby, M. & Ostrovsky, R. *Security of Blind Digital Signatures (Extended Abstract)* in *CRYPTO'97* (1997).
23. Kastner, J., Loss, J. & Xu, J. *On Pairing-Free Blind Signature Schemes in the Algebraic Group Model* in *PKC 2022* (2022).
24. Kastner, J., Loss, J. & Xu, J. *The Abe-Okamoto Partially Blind Signature Scheme Revisited* Cryptology ePrint Archive, Paper 2022/1232. 2022.
25. Katsumata, S., Nishimaki, R., Yamada, S. & Yamakawa, T. *Round-Optimal Blind Signatures in the Plain Model from Classical and Quantum Standard Assumptions* in *EUROCRYPT 2021, Part I* (2021).
26. Maitland, G. & Boyd, C. *A Provably Secure Restrictive Partially Blind Signature Scheme* in *PKC 2002* (2002).
27. Martinet, G., Poupard, G. & Sola, P. *Cryptanalysis of a Partially Blind Signature Scheme or How to Make $100 Bills with $1 and $2 Ones* in *FC 2006* (2006).
28. Okamoto, T. *Efficient Blind and Partially Blind Signatures Without Random Oracles* in *TCC 2006* (2006).
29. Papachristoudis, D., Hristu-Varsakelis, D., Baldimtsi, F. & Stephanides, G. *Leakage-Resilient Lattice-Based Partially Blind Signatures* 2019.
30. Pointcheval, D. & Stern, J. *Provably Secure Blind Signature Schemes* in *ASIACRYPT'96* (1996).
31. Pointcheval, D. & Stern, J. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology* (2000).
32. Rückert, M. *Lattice-Based Blind Signatures* in *ASIACRYPT 2010* (2010).
33. Schnorr, C.-P. *Security of Blind Discrete Log Signatures against Interactive Attacks* in *ICICS 01* (2001).
34. Schröder, D. & Unruh, D. *Security of Blind Signatures Revisited* in *PKC 2012* (2012).
35. Tessaro, S. & Zhu, C. *Short Pairing-Free Blind Signatures with Exponential Security* Cryptology ePrint Archive, Report 2022/047. 2022.
36. Tyagi, N. *et al. A Fast and Simple Partially Oblivious PRF, with Applications* Cryptology ePrint Archive, Report 2021/864. 2021.
37. Yi, X. & Lam, K.-Y. *A New Blind ECDSA Scheme for Bitcoin Transaction Anonymity* in *ASIACCS 19* (2019).
38. Zhang, F., Safavi-Naini, R. & Susilo, W. *Efficient Verifiably Encrypted Signature and Partially Blind Signature from Bilinear Pairings* in *INDOCRYPT 2003* (2003).