Lattice EPID with Efficient Revocation

Corentin Jeudy[1] and Olivier Sanders[1]
corentin.jeudy@orange.com, olivier.sanders@orange.com

Orange Labs, Applied Crypto Group, Cesson-Sévigné, France

Abstract. Enhanced Privacy Identification (EPID) is one of the anonymous authentication mechanisms that found their way into the industry, being deployed in billions of chips and standardized at ISO. The linchpin of EPID lies in its decentralized revocation procedure that allows to revoke a signer by simply placing one of its signatures on a signature revocation list SRL. Each new signature must then include a proof that it has been generated with a key different from those used to produce the signatures on the SRL. This proof of non-revocation in current post-quantum schemes either relies on general-purpose NIZKs or on regular zero-knowledge proofs (ZKP) but with a witness dimension linear in the size of the SRL, which leads to large size and/or computational complexity.

In this paper, we rethink the standard approach of non-revocation so as to avoid its heavy reliance on ZKP. Our construction indeed combines features from different tools (such as Falcon signatures) that are unusual in this context to pull most elements out of the ZKP, leading to significant performance improvements. Providing all these elements unconcealed creates many security challenges for our construction but we yet manage to address all of them and prove security under well-understood lattice assumptions, and in the strong model of Sanders-Traoré (CT-RSA'21) allowing malicious SRLs.

Keywords: Lattice-Based Cryptography · EPID · Revocation · Privacy

1 Introduction

Electronic authentication often relies on digital certificates issued by an authority to attest the validity of some user's claims. This approach has proven to be extremely efficient and secure but has been criticized for a long time for its lack of privacy. A certificate can indeed leak unnecessary information on its owner and can be traced when presented several times. This has led cryptographers to propose a series of primitives (e.g., [Cha82,Cha85,CvH91]) that differ in the details, but all share the same goal: to propose the private counterpart of the authentication processes we use today.

Over time, these privacy-preserving constructions acquired new capabilities (e.g., proving more complex statement) and gained in efficiency, at least for classical (i.e., non post-quantum) settings, which reduced the gap with non-private authentication methods. There is however one aspect on which private solutions hardly compete with the latter: the ability to easily revoke a user. This is not very surprising as privacy inherently conflicts with the usual approach for revocation, where a list of revoked credentials (or associated identifiers) is published and compared to the presented credential. Cryptographers must then find a way to ensure untraceability of non-revoked users while preventing use of revoked credentials, which is not easy. This probably explains why this problem is usually sidestepped in many papers, and why even mainstream models (e.g., [BMW03,BSZ05,FHS19]) do not consider revocation, although it is a very common process in practice. It may indeed result from the loss or theft of a user's device, from the change of the user's attributes (e.g., address, employer, etc) or simply from non-compliance with the system rules. A privacy-preserving mechanism failing to efficiently support revocation is thus likely to face many hurdles in its deployment.

When looking at the cryptographic literature, we can actually distinguish three main approaches for revoking credentials in anonymous systems. The first one is universal but very cumbersome. It consists in revoking all the keys of the system, and thus all the credentials issued so far. This corresponds to a system reboot which forces every unrevoked user to query new credentials. It is thus an extreme solution that may be justified in critical situations (e.g., in case of a major security breach, leakage of authorities' secret keys, etc) but cannot address the routine revocations needed in practice.

The second approach is the one where some authority can generate a tracing token that can be used to decide whether an (alleged) anonymous signature was produced by a given user. Concretely, a token per revoked user is placed on a so-called verifier-local revocation list (e.g., [BS04,LLNW14]) that allows verifiers to check if the signature were produced by a revoked user. Unfortunately, these tokens can do more than that as they could be used to trace all the signatures produced by a user, even before being revoked. While this could be somehow mitigated by restricting tokens to some epochs (see e.g., [San21]), this does not fully address the problem (the tokens still enable user tracing within an epoch) and weakens the privacy assurances since the authority can still passively trace every user.

The third approach is the one where some authority regularly publishes some elements that, de facto, update the list of allowed (non-revoked) users. This is often done through cryptographic accumulators (see, e.g. [CL02]) that allow users to produce a compact proof that they still belong to the set of allowed users or, alternatively, that they do not belong to the set of revoked ones. However, accumulators usually imply some centralized revocation procedure. Indeed, they are generally built using information, like the whole set of users' public keys or certificate identifiers, that may not be accessible to a mere verifier. Moreover, even if we provide this information to the verifiers, it is unclear how they could handle

revocation on their own. Indeed, let us consider the case of a service provider that would grant access to its service after verifying some user's anonymous credential. If this user misbehaves, the service provider might want to revoke them but how to do it concretely? All it has is an anonymous credential. An opening procedure (as in group signatures) could solve the problem as it enables to trace back the credential to its issuer, but it cannot be directly accessible to the service provider, otherwise it would remove every privacy assurance. One must then entrust some authority with the management of this procedure, hence the centralized process mentioned above.

To address this problem, and thus provide a decentralized revocation mechanism, one can instead implement this third approach as in EPID (Enhanced Privacy ID) systems [BL07]. Concretely, an EPID system defines a Signature Revocation List (SRL) where one can place the signatures[1] produced by the users one wants to revoke. The SRL is then taken as input by all future signers that must prove they did not generate any element on this list. This way, users that have produced the signatures on the SRL are unable to produce new signatures (for this SRL) and are thus revoked. If we come back to our example above, we note that EPID can be readily used by the service provider to revoke misbehaving users. Moreover, the service provider can define its own SRL[2] and so does not need to synchronize with other verifiers that may use different SRLs. We thus get a decentralized revocation mechanism, although we note that it is still compatible with centralized revocation. For example, the group manager could get the elements necessary to revoke users when they register and then place them on a central SRL if it wants to revoke them.

1.1 Existing EPID Design Approaches

Since its introduction, several efficient EPID constructions were proposed in classical groups, e.g., [BL07,BL10,ST21,San22], and standardized at ISO [ISO13]. The situation of post-quantum EPID systems is unfortunately much less satisfying. To our knowledge, the first post-quantum EPID construction was proposed in [BEF19] and only relies on symmetric primitives (hash functions and block ciphers). The signature size is about 200 MB large but can be reduced to a few megabytes in settings where the users need to update their keys after each registration of a new user (which may not be practical in dynamic settings). No timings are provided, but the heavy use of general purpose NIZKs suggests a high computational complexity. A lattice-based construction, LEPID, was proposed in [KFM+19], with signatures of about 850 MB that require hundreds of seconds to be generated and verified. Very recently, Chen et al. [CDK+24] introduced a hash-based construction which improves over [BEF19] but still inherits the advantages (relatively small sizes) and downsides (high computational complexity) of relying on symmetric algorithms and general-purpose NIZKs. Moreover, the

[1] Only part of each signature is required in practice.

[2] Actually, the original EPID security model does not allow that but this question was addressed in [ST21].

complexity is extremely sensitive to the number of users and the size of SRL, leading to signatures of about 11.8 MB generated in close to 90 seconds when the SRL contains 1000 elements.

Before introducing our contribution, we first recall the approach followed by previous EPID constructions (including classical ones) so as to highlight the main challenges in their design and explain how they were overcome in practice (and why we chose to depart from those approaches). An EPID system considers a set of platforms (a.k.a. users) that need to interact with an issuer $\mathcal{I}$ to join a group. Once this is done, they have the ability to produce anonymous signatures on behalf of the group. So far, this is essentially a group signature [CvH91,BMW03]. The main difference with the latter primitive lies in safeguards placed to deter bad behaviour. Concretely, instead of the opening process of group signature that allows some authority to lift anonymity of any signature, EPID supports the SRL lists presented above which allow for revoking platforms. The similarities and differences with group signatures are reflected by their technical component. As in group signatures, a platform must prove it is part of the group which is usually done by proving knowledge of a certificate (generated by $\mathcal{I}$) on its secret key. This part is standard and can be instantiated efficiently, even with lattices (see e.g., [AGJ+24]). The main technical challenge is thus to implement the proof of non-revocation and we now focus on this part in the remainder of this section.

In practice, EPID systems force every platform to include in its signature a tag tag whose only purpose is to simplify (potential) future revocation. To bind it to the platform, the tag must depend on its secret key while, at the same time, preventing traceability. In classical constructions, the usual approach is to generate a fresh basis h and compute tag as h^s where s is the platform secret key. Linking h^s to $(h')^s$ for a different basis h' is equivalent to solving the DDH problem, hence the untraceability of this approach. However, when it comes to proving non-revocation, the situation becomes more complex. Indeed, the SRLs contain pairs $(h_i, \mathsf{tag}_i = h_i^{s_i})$ from revoked signatures and the signer must now prove that its own secret key s would yield different tags. At first sight, one could think that it is sufficient to produce the elements $t_i = h_i^s$, along with a NIZK proof of well-formedness. This way, the verifier would simply check whether $t_i = \mathsf{tag}_i$, or not. Unfortunately, t_i would be common to every signature produced by this platform with a SRL containing $(h_i, h_i^{s_i})$, allowing to trace the platform. One must then design proofs of non-revocation that remain unlinkable even when produced on the same SRL (or on SRLs with non-empty intersection).

The solution proposed in [BL07] is to consistently re-randomize each element, that is, to replace h_i by $h_i^{r_i}$, $h_i^{s_i}$ by $(h_i^{s_i})^{r_i}$, and set t_i as $(h_i^{r_i})^s$ so as to make proofs unlinkable. However, not only does this increase the size of the proof that must include all these elements, but this also requires zero-knowledge proofs that all of them are well-formed, which adds to the complexity. In classical settings, there are alternative approaches (e.g., proofs of inequality of discrete logs [CS03], use of the Dodis-Yampolskiy PRF [DY05] in [San22]) we only mention here but that we will not consider as they do not seem to be transposable in a lattice setting.

The LEPID approach [KFM+19] adapts the idea in [BL07] to lattices and thus inherits its main features. Concretely, a tag is now defined as $\mathsf{tag}_i = \mathbf{a}_i \cdot s_i + \mathbf{e}_i$ where s_i is the platform's secret key, $\mathbf{a}_i$ is a random vector (playing the role of the basis h_i) and $\mathbf{e}_i$ is a small error. Under the LWE assumption, this tag appears to be random. Here again, one might be tempted to prove that its own secret key s has not produced this tag by generating $t_i = \mathbf{a}_i \cdot s + \mathbf{e}'_i$ (which is close to tag_i if and only $s = s_i$), but one would face the same issue as above. Two signatures produced with the same tag_i as input (in the SRL) would respectively yield $\mathbf{a}_i \cdot s + \mathbf{e}'_i$ and $\mathbf{a}_i \cdot s + \mathbf{e}''_i$, whose difference $\mathbf{e}'_i - \mathbf{e}''_i$ is small, enabling to trace the platform. This led the authors of LEPID to resort to the same re-randomization technique, that is, compute, $\mathbf{r}_i^T\mathbf{a}_i + \mathbf{f}_i$, $\mathbf{r}_i^T\mathsf{tag}_i + \mathbf{f}'_i$, $(\mathbf{r}_i^T\mathbf{a}_i + \mathbf{f}_i)s + \mathbf{f}''_i$, etc. and prove that every element is well-formed. In particular, in addition to sending these $3N$ uniform-looking (and thus incompressible) elements (where N is the number of elements on the SRL), one can note that every re-randomization vector ($\mathbf{r}_i$, $\mathbf{f}_i$, $\mathbf{f}'_i$, $\mathbf{f}''_i$, etc) here must be small and concealed in a zero-knowledge proof. The latter then also scales with N, explaining the complexity of the resulting construction.

If we take a step back, we note that both [BL07] and [KFM+19] ask the platform to produce a randomized tag, which we later call token, t_i for a random basis ($h_i^{r_i}$ or $\mathbf{r}_i^T\mathbf{a}_i + \mathbf{f}_i$). This randomization step seems necessary to avoid the traceability issue presented above, unless one is willing to conceal altogether the deterministic elements h_i^s or $\mathbf{a}_i\mathbf{s} + \mathbf{e}$ in the NIZK proofs, which would increase the complexity of the latter. As the platform generates itself t_i in [BL07] and [KFM+19], it must provide a proof that it, and all the intermediary elements, are well formed, which linearly increases the complexity of the NIZK proof.

1.2 Our Contributions

In this work, we propose a lattice-based construction relying on standard assumptions that outperforms the sizes of the latest symmetric construction [CDK+24] while only relying on lattice building blocks (NIZKs proofs for quadratic relations [LNP22], signature with efficient protocols [JRS23,AGJ+24] and signatures à la Falcon [PFH+20]) that have proven to be much faster. Our main contribution is a new approach to handle proofs of non-revocation which drastically reduces the weight of the NIZK proof (which was the main source of inefficiency for LEPID as we explained above) as the signer must only prove well-formedness of a couple of elements in a zero-knowledge way, regardless of the size of the revocation list. All along the paper, we follow the EPID terminology and model but stress that our proofs of non-revocation are actually very modular and so could be used in other primitives (e.g., anonymous credentials), to support decentralized revocation.

Warm-up. Warming up to our construction, let us bridge it with the previous LEPID approach where each signature includes a tag $\mathsf{tag} = \mathbf{a}s + \mathbf{e}$. As we explained, the only purpose of this tag is to be placed on a revocation list SRL in case one wants to revoke the platform that issued this signature. One then ends

up with a SRL containing several tags $\mathsf{tag}_i = \mathbf{a}_i s_i + \mathbf{e}_i$ and the goal of the signer is to (1) re-randomize these tags into $\widetilde{\mathsf{tag}_i}$ (to remain unlinkable, see above) and (2) produce non-revocation tokens t_i that should be close to $\widetilde{\mathsf{tag}_i}$ if, and only if, they involve the same secret key s_i. This approach is abstracted in Figure 1.1.

Public Re-Randomization. Our construction retains this tag structure $\mathsf{tag} = \mathbf{a}s + \mathbf{e}$ but requires a much more elaborate definition of $\mathbf{a}$ that we discuss later. For the moment, our first target towards diminishing the weight of ZK-proofs is to define a *public* re-randomization process. Our first idea is to define $\widetilde{\mathsf{tag}_i}$ as $\boldsymbol{\gamma}_i^T \mathsf{tag}_i$ for some short, public $\boldsymbol{\gamma}_i$ generated as a hash output. As the $\boldsymbol{\gamma}_i$ and tag_i are public, the randomized tags $\widetilde{\mathsf{tag}_i}$ do not have to be computed and sent by the signer, and in turn no well-formedness proof is needed for them. This obviously alleviates the witness of the zero-knowledge proof but this creates new security challenges that could have catastrophic consequences if they are not managed carefully. For example, if one sets $\boldsymbol{\gamma}_i$ as a single polynomial, then two signatures involving the same tag tag_i on the SRL would lead to re-randomized tags $\widetilde{\mathsf{tag}_i} = \boldsymbol{\gamma}_i \mathsf{tag}_i$ and $\widetilde{\mathsf{tag}_i}' = \boldsymbol{\gamma}_i' \mathsf{tag}_i$. As both $\boldsymbol{\gamma}_i$ and $\boldsymbol{\gamma}_i'$ are public, one could link the resulting two signatures as $\boldsymbol{\gamma}_i' \widetilde{\mathsf{tag}_i}$ is likely to be close to $\boldsymbol{\gamma}_i \widetilde{\mathsf{tag}_i}'$. Alternatively, an adversary could reuse the same $\mathbf{a}$ as an honest platform and then place the resulting tag on the SRL. As we explain in this paper, this would allow an adversary to get two elements $\boldsymbol{\gamma}\mathbf{a}s + \mathbf{e}$ and $\boldsymbol{\gamma}\mathbf{a}s + \mathbf{e}'$ for different errors $\mathbf{e}$ and $\mathbf{e}'$, which is likely to leak information on s. These are just examples of the problems one needs to account for in our case when generating the vector $\mathbf{a}$ used to build tag. As more requirements will appear, we still let $\mathbf{a}$ unspecified for the moment.

At this stage, the signer (whose signing key is s) has thus *implicitly*[3] produced re-randomized tags $\widetilde{\mathsf{tag}_i} = \boldsymbol{\gamma}_i^T \mathbf{a}_i s_i + \boldsymbol{\gamma}_i^T \mathbf{e}_i$ and, if we resorted to previous approaches[4], one would produce non-revocation tokens $t_i = \boldsymbol{\gamma}_i^T \mathbf{a}_i s + e_i'$, for short e_i'. Thanks to these tags and tokens, the verifier could test non-revocation as $s_i = s$ implies that $\widetilde{\mathsf{tag}_i}$ is close to t_i. However, there are two main efficiency problems with this approach. First, it requires to send N (the number of elements on the SRL) random-looking tokens t_i that cannot be compressed. Second, it requires to prove well-formedness of *each* t_i and thus prove knowledge of N elements e_i'. With traditional proof systems like that of [LNP22], this makes the proof size scale linearly with N. This could be mitigated by using zk-SNARKs but the latter have their own downsides.

Trading Zero-Knowledge Proofs for Signatures. In our construction, we avoid these two problems by enabling the verifier to compute all the t_i itself. Indeed, in our case, the platform will generate a Falcon public key h, along with the associated trapdoor, and produce a unique element $t = hs + e$, regardless of the number of tags tag_i in the SRL. Thanks to the Falcon trapdoor, it can produce short polynomials $x_{i,1}$ and $x_{i,2}$ such that $x_{i,1} + x_{i,2}h = u_i$, where $u_i =$

[3] By "implicitly", we mean that the signer does not even need to compute them as they can be directly recovered by the verifier from the public $\boldsymbol{\gamma}_i$.

[4] Here again, we refer to Figure 1.1 for an overview of the main steps of the proof of non-revocation.

$\boldsymbol{\gamma}_i^T \mathbf{a}_i$. Note that we made $\boldsymbol{\gamma}_i$ public and so are $\mathbf{a}_i$ and h, so anyone can check the validity of the Falcon "signature" $(x_{i,1}, x_{i,2})$ on u_i. Actually, we only need to transmit the second component $x_{i,2}$ since $x_{i,2}t = (u_i - x_{i,1})s + x_{i,2}e$ is close to t_i and thus sufficient to perform the non-revocation check. The actual token would then be $t_i = x_{i,2}t$, but we insist that the signer never has to compute and send these elements as they can be reconstructed publicly by the verifier that receives t and the $x_{i,2}$'s, as illustrated in Figure 1.1. Putting everything together, we just need to send a unique t, along with a proof of well-formedness, regardless of the size of the SRL, and N short, Gaussian (and hence compressible) $x_{i,2}$ instead of N uniform t_i, which drastically improves performance. This comparison with the LEPID approach [KFM+19] is highlighted in the transcript phase of Figure 1.1.

Proving Security. We have therefore a compact proof of non-revocation, but is it secure? At first sight, and if we let aside the definition of $\mathbf{a}$, it seems so. The element $t = hs + e$ indeed looks like an LWE sample and so should hide s and thus the identity of the signer. Actually, t has exactly the structure we expect from tag so it seems that we could even kill two birds with one stone and set $\mathsf{tag} = t$. However, when we look closer, we note that the situation is a bit more complex as we intrinsically need the trapdoor associated to h whereas an LWE sample would only involve a *perfectly random* h. One could try to define some hybrid NTRU-LWE assumption to address this problem but it would be quite contrived and would depart from our goal to rely on standard assumptions.

Instead, we rather slightly adapt our construction to only rely on the standard Ring LWE (R-LWE) and NTRU assumptions. As a consequence, we will have no choice in our security proof but to replace t by a R-LWE sample, that is, an element that is either random or can be written as $hs + e$ for a *random* h. In all cases, we lose the trapdoor in h and must then find an alternative way to construct the Falcon signatures $(x_{i,1}, x_{i,2})$ satisfying $x_{i,1} + x_{i,2}h = u_i$. To this end, we leverage both the $\mathbf{a}_i$ used to produce the tags tag_i and the randomization vectors $\boldsymbol{\gamma}_i$. Indeed by hiding a trapdoor in $\mathbf{a}_i$ (which can be done in the random oracle model, by generating $\mathbf{a}_i$ as a hash output), one gets the ability to find, for any polynomial v_i, short $\boldsymbol{\alpha}_i$ such that $\boldsymbol{\alpha}_i^T \mathbf{a}_i = v_i$. We can thus shift, in our security reduction, the trapdoor we need from h to $\mathbf{a}_i$ which, combined with a careful definition of $\boldsymbol{\gamma}_i$, provides the ability to perfectly simulate all the elements of the proof of non-revocation, including the $(x_{i,1}, x_{i,2})$.

Note that, in the process, we have identified some requirements on the vector $\mathbf{a}$ used to build $\mathsf{tag} = \mathbf{a}s + \mathbf{e}$. It must be distributed in a way compatible with a trapdoor pre-image sampler and generated as a hash output. It thus remains to define the proper input inp for the hash function. One might be tempted to define inp as a mere seed but, then, what would prevent an adversary from using the same seed for its signature, yielding the same $\mathbf{a}$ and thus leading to the problem mentioned above? We must then ensure that a platform will never produce a signature if the SRL contains an input inp that it has already used before. This is done by designing a detection procedure that allows a platform to recognize the elements inp it used, in which case it aborts. At the same time, we must ensure that this procedure (and the potential abortions it entails)

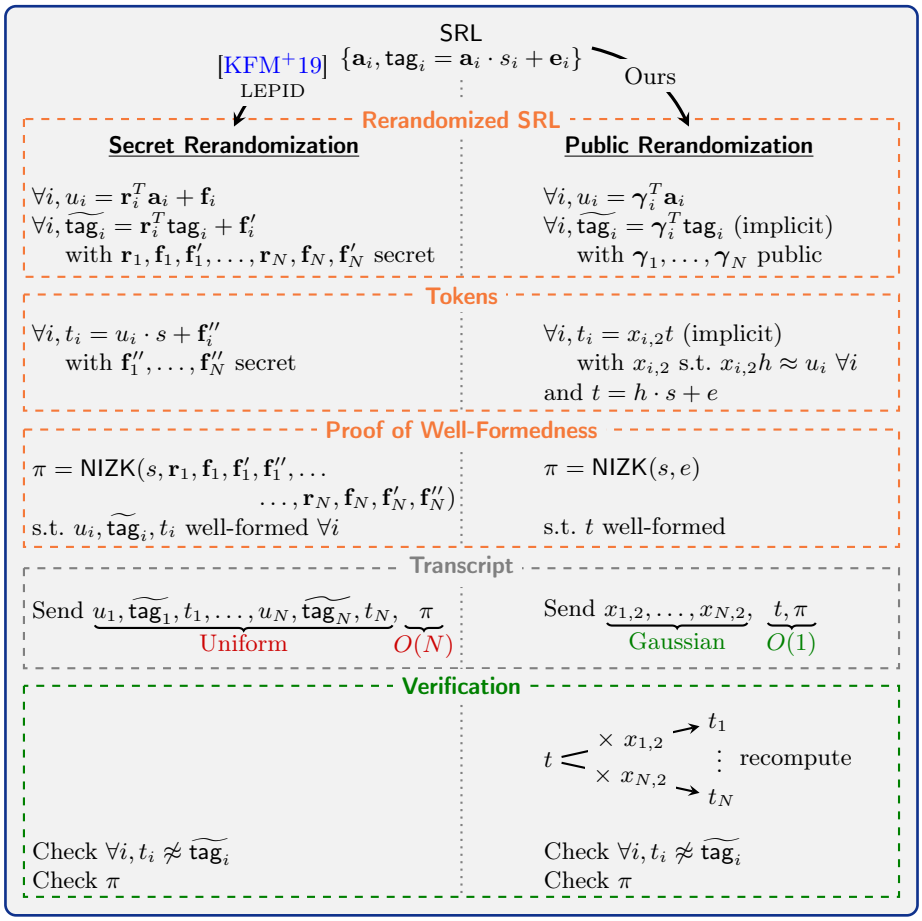

Fig. 1.1. High-level overview of our proof of non-revocation compared to the LEPID approach [KFM+19]. The elements from the transcript marked uniform are incompressible uniform modulo p, while Gaussian refers to compressible Gaussian elements (e.g., with rANS encoding). The $O(N)$ and $O(1)$ labels specify the asymptotic dependency in N. The strength of our solution is that many elements can be publicly generated and thus no longer need to be included (along with a NIZK of well-formedness) in the proof. Such elements are marked as "implicit" because the user does not need to compute them but are still displayed to highlight the conceptual similarities and differences with [KFM+19].

will not be diverted by an adversary to prevent an honest user from producing signatures, thus creating a denial of service. The latter point pleads for deriving, in a verifiable way, inp from the platform secret key s which, combined with the privacy requirement, leads to use a verifiable pseudo-random function. In our case, we define $\mathsf{inp} = (\mathsf{seed}, c)$ for a fresh random seed seed and where c is an element defined by $c = \mathcal{H}(\mathsf{seed})s + e$, but with a few subtleties regarding the

generation of e that we explain in Section 4. In all cases, this only adds a single element regardless of the size of the SRL thus entailing a very minor increase of the complexity. It also does not affect the other elements of the EPID signatures.

Performance. In the end, we get an EPID system whose security is proven on standard lattice assumptions, while having a much more efficient revocation mechanism. As the zero-knowledge proof is not driven by the size of the SRL, our system can be parameterized in a very modular fashion for different values of N (the maximal size of an SRL). Concretely, our EPID signature is roughly 118 KB (which includes seed, h, tag, t and c, and the proof of group membership and well-formedness of tag, c and t) to which is added the size of $x_{1,2}, \ldots, x_{N,2}$. As a result, for $N = 10$, our total EPID signature size is around 170 KB, while for $N = 1000$, it is around 5.3 MB. From these size considerations alone, we are around 160 times more compact than the previous lattice construction [KFM+19], and even 2.2 times more compact than the most efficient hash-based solution [CDK+24]. Regarding timing performances, our reliance on well-known lattice building blocks means our system is also very likely to outperform symmetric constructions relying on general-purpose NIZKs. Even though we leave the implementation of our system for future work, recent work have shown concrete evidence of the efficiency of the building blocks we use [AGJ+24,LSS24,JS25a,JS25b] for the group signature component. The proof of non-revocation then simply consists in generating one Falcon key and N Falcon signatures which would also be very efficient compared to proving knowledge of $O(N)$-dimensional witnesses using [LNP22] or [BS23].

Beyond the sole application of EPID systems, we insist that our revocation strategy, which represents our main result, is very modular and can interface with other anonymous authentication primitives like anonymous credentials [AGJ+24] and more. We also note that our construction is the only post-quantum construction that provably tackles malicious revocation lists, a feature desirable in practical use cases of EPIDs and revocation in general as pointed out in [ST21].

2 Preliminaries

We use $\mathbb{N}, \mathbb{Z}, \mathbb{R}$ to respectively denote the set of natural integers, the ring of integers, and the field of reals. For two integers $a \leq b$, we define $[a, b] = \{a, \ldots, b\}$ and $[b] = [1, b]$. For a positive integer q, we define $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. We may refer to $a \mod^{\pm} q$ as the unique representative of $a \in \mathbb{Z}_q$ that lies in $(-q/2, q/2] \cap \mathbb{Z}$. We also denote by $R^\times$ the unit group of a ring R, including for $\mathbb{N}$ where $\mathbb{N}^\times$ abusively corresponds to $\mathbb{N} \setminus \{0\}$. The uniform distribution over a finite set S is denoted by $U(S)$. Finally, for two distributions $\mathcal{P}_1, \mathcal{P}_2$ over a countable set S, we define the statistical distance between $\mathcal{P}_1$ and $\mathcal{P}_2$ by $\Delta(\mathcal{P}_1, \mathcal{P}_2) = \frac{1}{2} \sum_{x \in S} |\mathcal{P}_1(x) - \mathcal{P}_2(x)|$.

2.1 Lattices

A d-dimensional lattice $\mathcal{L}$ is a discrete subgroup of $(\mathbb{R}^d, +)$. For a lattice $\mathcal{L}$, we define its first minimum as $\lambda_1(\mathcal{L}) = \min_{\mathbf{x} \in \mathcal{L} \setminus \{\mathbf{0}\}} \|\mathbf{x}\|_2$. The dual lattice of a lattice

$\mathcal{L}$ is defined by $\mathcal{L}^* = \{\mathbf{x} \in \mathrm{Span}_{\mathbb{R}}(\mathcal{L}) : \forall \mathbf{y} \in \mathcal{L}, \mathbf{x}^T\mathbf{y} \in \mathbb{Z}\}$. We define specific lattices known as q-ary lattices, each associated to a matrix $\mathbf{A} \in \mathbb{Z}_q^{d\times m}$ and defined by $\mathcal{L}_q^{\perp}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \bmod q\mathbb{Z}\}$. For all $\mathbf{u} \in \mathbb{Z}_q^d$, we also define the lattice coset $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{u} \bmod q\mathbb{Z}\}$.

2.2 Probabilities

For a positive definite matrix $\mathbf{S} \in \mathbb{R}^{d\times d}$, and $\mathbf{c} \in \mathbb{R}^d$, we define the Gaussian function $\rho_{\sqrt{\mathbf{S}},\mathbf{c}}$ by $\rho_{\sqrt{\mathbf{S}},\mathbf{c}}(\mathbf{x}) = \exp(-\pi(\mathbf{x}-\mathbf{c})^T\mathbf{S}^{-1}(\mathbf{x}-\mathbf{c}))$ for all $\mathbf{x} \in \mathbb{R}^d$. Note that the expression only depends on $\mathbf{S}$ and not a specific choice of square root, which is why we index the function by $\sqrt{\mathbf{S}}$. For any d-dimensional lattice $\mathcal{L}$, we define the discrete Gaussian distribution by its probability mass function $\mathcal{D}_{\mathcal{L},\sqrt{\mathbf{S}},\mathbf{c}} : \mathbf{x} \in \mathcal{L} \mapsto \rho_{\sqrt{\mathbf{S}},\mathbf{c}}(\mathbf{x})/\rho_{\sqrt{\mathbf{S}},\mathbf{c}}(\mathcal{L})$. When $\mathbf{c} = \mathbf{0}$, we omit the subscript, and when $\mathbf{S} = s^2\mathbf{I}_d$ for $s > 0$, we replace the subscript $\sqrt{\mathbf{S}}$ by s.

We later use the following tail bound on the Euclidean norm of discrete Gaussians.

Lemma 2.1 ([Ban93, Lem. 1.5]). *Let $\mathcal{L} \subset \mathbb{R}^d$ be a lattice of rank d, and $s > 0$. Then, for all $c > 1/\sqrt{2\pi}$, we have*

$$\mathbb{P}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{L},s}}\left[\|\mathbf{x}\|_2 > cs\sqrt{d}\right] < \left(c\sqrt{2\pi e}e^{-\pi c^2}\right)^d.$$

We use c_d to denote the smallest $c > 1/\sqrt{2\pi}$ such that $(c\sqrt{2\pi e}e^{-\pi c^2})^d \leq 2^{-(\lambda+O(1))}$, where λ is the implicit security parameter. As coined by Micciancio and Regev [MR07], we define the *smoothing parameter* of a lattice $\mathcal{L}$, parameterized by some $\varepsilon > 0$, by $\eta_\varepsilon(\mathcal{L}) = \min\{s > 0 : \rho_{1/s}(\mathcal{L}^*) = 1+\varepsilon\}$. A recent work by Espitau, Wallet and Yu [EWY23] gives an exact expression of $\eta_\varepsilon(\mathcal{L})$ with tight approximations for remarkable lattices.

Lemma 2.2 ([EWY23, Lem. 5]). *Let $\mathcal{L}$ be a lattice and $\varepsilon > 0$. It holds that*

$$\eta_\varepsilon(\mathcal{L}) = \frac{1}{\lambda_1(\mathcal{L}^*)}\sqrt{\frac{1}{\pi}\ln\left(\frac{\kappa(\mathcal{L}^*)}{\varepsilon}(1+o_\varepsilon(1))\right)},$$

where $\kappa(\mathcal{L}^*) = |\{\mathbf{x} \in \mathcal{L}^* : \|\mathbf{x}\|_2 = \lambda_1(\mathcal{L}^*)\}|$ is the kissing number of $\mathcal{L}^*$. In particular, it holds that for any $d \in \mathbb{N}^\times$, $\eta_\varepsilon(\mathbb{Z}^d) \approx \sqrt{\ln(2d/\varepsilon)/\pi}$.

We also need the following straightforward tail bound on uniform elements.

Lemma 2.3. *Let q, n, B, λ be in $\mathbb{N}^\times$. Then, $\mathbb{P}_{\mathbf{a}\sim U(\mathbb{Z}_q^n)}[\left\|\mathbf{a} \bmod^{\pm} q\right\|_\infty \leq B] = ((2B+1)/q)^n$. When $B \leq \lfloor(q2^{-\lambda/n}-1)/2\rfloor$, the probability is bounded by $2^{-\lambda}$.*

Proof. It holds that $\mathbb{P}_{\mathbf{a}\sim U(\mathbb{Z}_q^n)}[\left\|\mathbf{a} \bmod^{\pm} q\right\|_\infty \leq B] = \mathbb{P}_{a\sim U(\mathbb{Z}_q)}[\left|a \bmod^{\pm} q\right| \leq q]^n = ((2B+1)/q)^n$. For the second statement, the definition of B gives $B \leq (q2^{-\lambda/n}-1)/2$. This is equivalent to $((2B+1)/q)^n \leq 2^{-\lambda}$. Combined with the first statement, it yields the claim. □

2.3 Algebraic Number Theory

We now give the necessary notions in algebraic number theory. A number field $\mathcal{K} = \mathbb{Q}(\zeta)$ is a field extension of $\mathbb{Q}$ of finite degree n adjoining an algebraic number ζ. The set of algebraic integers in $\mathcal{K}$ is a ring $\mathcal{R}$ called the ring of integers of $\mathcal{K}$. We also define $\mathcal{K}_{\mathbb{R}} = \mathcal{K} \otimes_{\mathbb{Q}} \mathbb{R}$. For any $q \geq 2$, we define $\mathcal{R}_q = \mathcal{R}/q\mathcal{R}$. A popular choice of number field is the class of power-of-two cyclotomic fields, for which the degree n is a power-of-two and which are isomorphic to $\mathbb{Q}[x]/\langle x^n + 1\rangle$. The ring of integers in this case is identified with $\mathbb{Z}[x]/\langle x^n + 1\rangle$.

The following is stated for $\mathcal{K}_{\mathbb{R}}$ but holds also for $\mathcal{K}$ and $\mathcal{R}$. Elements of $\mathcal{K}_{\mathbb{R}}$ can naturally be embedded into the Euclidean space $\mathbb{R}^n$ by their coefficient vector when seen as a polynomial in ζ or x. We use τ to denote this coefficient embedding, i.e., for $r = \sum_{i=0}^{n-1} r_i \zeta^i$, $\tau(r) = [r_0 | \dots | r_{n-1}]^T$. We also define the conjugate r^* of the element r by being $r^* = r(\zeta^{-1})$. In the power-of-two cyclotomic field of degree n, it holds that $\tau(r^*) = [r_0 | -r_{n-1} | \dots | -r_1]^T$. We then define the multiplication matrix map M_τ defined by the relation $\tau(rs) = M_\tau(r)\tau(s)$ for all pairs of elements r, s. In power-of-two cyclotomic fields, $M_\tau(r)$ corresponds to the nega-circulant matrix with first column $\tau(r)$. We define the usual ℓ^p norms over $\mathcal{K}_{\mathbb{R}}$ with respect to the embedding τ, i.e., $\|r\|_p := \|\tau(r)\|_p$.

These notations extend to vectors and/or matrices in the natural way by concatenation, except that the conjugate of a matrix actually corresponds to the conjugate transpose. For a matrix $\mathbf{A} \in \mathcal{K}_{\mathbb{R}}^{d \times m}$, we define its spectral norm through its embedding to $\mathbb{R}^{nd \times nm}$ via M_τ as $\|\mathbf{A}\|_2 = \|M_\tau(\mathbf{A})\|_2$.

We extend the notations $\mathcal{L}_q^\perp(\mathbf{A})$ and $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A})$ for matrices over $\mathcal{R}_q$ as $\mathcal{L}_q^{\mathbf{u}}(\mathbf{A}) = \{\mathbf{x} \in \mathcal{R}^m : \mathbf{A}\mathbf{x} = \mathbf{u} \bmod q\mathcal{R}\}$ and $\mathcal{L}_q^\perp(\mathbf{A}) = \mathcal{L}_q^{\mathbf{0}}(\mathbf{A})$. Gaussian distributions over the ring are defined through their coefficient embedding, i.e., $\mathcal{D}_{\mathcal{M},\sqrt{\mathbf{S}}} = \tau^{-1}(\mathcal{D}_{\tau(\mathcal{M}),\sqrt{\mathbf{S}}})$ where $\tau(\mathcal{M})$ is an module lattice corresponding to an $\mathcal{R}$-module $\mathcal{M}$. We also define the centered binomial distribution $\mathcal{B}_1$ over $\mathcal{R}$ obtained by sampling each coefficient according to ψ_1, with $\psi_1(-1) = \psi_1(1) = 1/4$ and $\psi_1(0) = 1/2$.

Lemma 2.4. *Let $\mathcal{K}$ be a number field, and $\mathcal{R}$ its ring of integer. Let p be a prime that is unramified in $\mathcal{R}$, and $p\mathcal{R} = \prod_{i\in[\kappa]} \mathfrak{p}_i$ its prime ideal factorization, where $\mathfrak{p}_i$ are distinct prime ideals. It holds that $\mathbb{P}_{u\sim U(\mathcal{R}_p)}[u \in \mathcal{R}_p^\times] = \prod_{i\in[\kappa]}(1 - N(\mathfrak{p}_i)^{-1})$, where $N(\mathfrak{p}_i)$ is the algebraic norm of the ideal $\mathfrak{p}_i$.*

Note that when $\mathcal{K}$ is a cyclotomic field of degree n, the probability of being a unit is $(1 - p^{-n/\kappa})^\kappa \geq 1 - \kappa \cdot p^{-n/\kappa}$.

Proof. It first holds that $p_{inv} := \mathbb{P}_{u\sim U(\mathcal{R}_p)}[u \in \mathcal{R}_p^\times] = \left|\mathcal{R}_p^\times\right| / |\mathcal{R}_p|$. By the chinese remainder theorem for ideals and the prime ideal factorization of $p\mathcal{R}$, it holds that $\mathcal{R}_p$ is isomorphic to the direct product of residue fields $\otimes_{i\in[\kappa]} \mathcal{R}/\mathfrak{p}_i$. It yields $|\mathcal{R}_p| = \prod_{i\in[\kappa]} N(\mathfrak{p}_i)$, where $N(\mathfrak{p}_i) = |R/\mathfrak{p}_i|$ is the algebraic norm. Similarly, $\mathcal{R}_p^\times$ is isomorphic to $\otimes_{i\in[\kappa]}(\mathcal{R}/\mathfrak{p}_i)^\times = \otimes_{i\in[\kappa]}((\mathcal{R}/\mathfrak{p}_i) \setminus \{0\})$, which entails $\left|\mathcal{R}_p^\times\right| = \prod_{i\in[\kappa]} N(\mathfrak{p}_i) - 1$. The ratio thus gives the expression of p_{inv} as desired. □

2.4 Samplers and Signatures

Our EPID construction leverages two different lattice preimage samplers, which we briefly recall here. The first, used to register EPID platforms, is the recent ring truncated sampler from [JS25b]. It is an improvement on the seminal gadget-based sampler from [MP12] which enables gadget truncation for a better efficiency, while being proven secure in the worst-case, a property that is crucial in advanced signature constructions. The sampler however preserves the same structure. We let $\mathbf{A}$ be a matrix in $\mathcal{R}_q^{d\times d}$, $\mathbf{G}_L = [b^0\mathbf{I}_d|\dots|b^{\ell-1}\mathbf{I}_d]$ and $\mathbf{G}_H = [b^\ell\mathbf{I}_d|\dots|b^{k-1}\mathbf{I}_d]$ be the low and high gadget matrices. We also let $\mathbf{R}_1, \mathbf{R}_2 \in \mathcal{R}^{d\times d(k-\ell)}$ be the short matrices constituting the trapdoor, and define $\mathbf{S} = \mathrm{diag}(s_1^2\mathbf{I}_d, s_2^2\mathbf{I}_{d(\ell-1)}, s_3^2\mathbf{I}_d, s_4^2\mathbf{I}_{d(k-\ell)})$. Then, for any tag $\mathfrak{t} \in \mathcal{R}_q^\times$ and syndrome $\mathbf{u} \in \mathcal{R}_q^d$, we define

$$\mathbf{L} = \begin{bmatrix} \mathfrak{t}\mathbf{I}_d & \mathbf{0} & \mathbf{R}_1 \\ \mathbf{0} & \mathfrak{t}\mathbf{I}_{d(\ell-1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{d(k-\ell)} \end{bmatrix},$$

and the following RingTruncatedSampler. It produces short Gaussian preimages $\mathbf{v}$ such that $[\mathbf{I}_d|\mathbf{A}|\mathfrak{t}\mathbf{G}_H - (\mathbf{R}_1 + \mathbf{A}\mathbf{R}_2)]\mathbf{v} = \mathbf{u} \bmod q\mathcal{R}$, for any fixed $\mathbf{u}$. In particular, the vector $\mathbf{v}'$ in Algorithm 2.1 follows a distribution satistically close to $\mathcal{D}_{\mathcal{L}_q^{\mathbf{u}}([\mathbf{I}_d|\mathbf{A}|\mathfrak{t}\mathbf{G}_H-(\mathbf{R}_1+\mathbf{A}\mathbf{R}_2)]),\sqrt{M_\tau(\mathbf{S})}}$ for carefully chosen widths s_1, s_2, s_3, s_4, which thus does not leak information on the trapdoor [JS25b, Thm. 4.1]. We recall here the security result associated to the sampler, and refer to [JS25b] for further details.

Lemma 2.5 ([JS25b, Thm. 4.2] adapted). *Let $\mathcal{R}$ be the ring of integer of a number field of degree n. Let d, q, b, ℓ be positive integers and $\varepsilon \in (0,1)$, such that $b \le \sqrt{q}$, and $\ell < k = \lceil \log_b q \rceil$. Let $\mathbf{R}_1, \mathbf{R}_2$ be in $\mathcal{R}^{d\times d(k-\ell)}$, $\mathbf{A} \in \mathcal{R}_q^{d\times d}$, $\mathbf{u} \in \mathcal{R}_q^d$, and $\mathfrak{t} \in \mathcal{R}_q^\times$. Define $\mathbf{G}_L = [1|b|\dots|b^{\ell-1}] \otimes \mathbf{I}_d$ and $\mathbf{G}_H = [b^\ell|\dots|b^{k-1}] \otimes \mathbf{I}_d$. Finally, let $t \ge \|\mathfrak{t}\mathbf{I}_d\|_2$, $s_{\mathbf{G}} = \eta_\varepsilon(\mathbb{Z}^{ndk})\sqrt{b^2+1}$ and $\mathbf{S} = \mathrm{diag}(s_1^2\mathbf{I}_d, s_2^2\mathbf{I}_{d(\ell-1)}, s_3^2\mathbf{I}_d, s_4^2\mathbf{I}_{d(k-\ell)})$ where*

$$s_1 = \eta_\varepsilon(\mathbb{Z}^{ndk})\left(b+\frac{1}{b}\right)\sqrt{t^2+3\|\mathbf{R}_1\|_2^2}s_2 = \eta_\varepsilon(\mathbb{Z}^{ndk})\left(b+\frac{1}{b}\right)t$$
$$s_3 = \sqrt{3}\eta_\varepsilon(\mathbb{Z}^{ndk})\left(b+\frac{1}{b}\right)\|\mathbf{R}_2\|_2 s_4 = \sqrt{3}\eta_\varepsilon(\mathbb{Z}^{ndk})\left(b+\frac{1}{b}\right)$$

The distribution of RingTruncatedSampler$(\mathbf{R}_1, \mathbf{R}_2, \mathbf{A}, \mathbf{u}, \mathfrak{t}, s_{\mathbf{G}}, s_1, s_2, s_3, s_4)$ is such that $\mathrm{Supp}(\mathcal{P}) = \mathcal{L}_q^{\mathbf{u}}(\mathbf{A}_{\mathbf{T}})$ *and*

$$\forall \mathbf{x} \in \mathcal{L}_q^{\mathbf{u}}(\mathbf{A}_{\mathbf{T}}), \mathcal{P}(\mathbf{x}) \in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^2, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\right] \cdot \mathbf{K}\mathcal{D}_{\mathcal{L}_q^{\mathbf{u}}(\mathbf{A}_{\mathbf{T}}\mathbf{K}),\sqrt{M_\tau(\mathbf{S})}}(\mathbf{x}),$$

with $\mathbf{A}_{\mathbf{T}} = [\mathbf{I}_d|\mathbf{A}|\mathbf{T}\mathbf{G}_H - (\mathbf{R}_1 + \mathbf{A}\mathbf{R}_2)] \bmod q\mathcal{R}$, *and* $\mathbf{K} = \mathrm{diag}(\mathbf{G}_L, \mathbf{I}_d, \mathbf{I}_{d(k-\ell)})$.

We note that the security proof requires at some point to hide a tag guess $\mathfrak{t}^+$ in the public key, making the effective tag $\mathfrak{t} - \mathfrak{t}^+$. This is why our parameter selection described in Algorithm 4.1 uses $t = 2w \geq \|\mathfrak{t}\|_1 + \|\mathfrak{t}^+\|_1 \geq \|\mathfrak{t} - \mathfrak{t}^+\|_1 \geq \|(\mathfrak{t} - \mathfrak{t}^+)\mathbf{I}_d\|_2$.

Algorithm 2.1: RingTruncatedSampler$(\mathbf{R}_1, \mathbf{R}_2, \mathbf{A}, \mathbf{u}, \mathfrak{t}, s_{\mathbf{G}}, s_1, s_2, s_3, s_4)$

Input: Trapdoor $\mathbf{R}_1, \mathbf{R}_2 \in \mathcal{R}^{d \times d(k-\ell)}$, Matrix $\mathbf{A} \in \mathcal{R}_q^{d \times d}$, Syndrome $\mathbf{u} \in R_q^d$, tag $\mathfrak{t} \in \mathcal{R}_q^{\times}$, Gaussian parameters $s_{\mathbf{G}}, s_1, s_2, s_3, s_4$.

1. $\mathbf{p} \hookleftarrow \mathcal{D}_{\mathcal{R}^{d(k+1)}, \sqrt{M_\tau(\mathbf{S}_p)}}$ with $\mathbf{S}_p = \mathbf{S} - s_{\mathbf{G}}^2 \mathbf{L}\mathbf{L}^* \in \mathcal{K}_{\mathbb{R}}^{d(k+1) \times d(k+1)}$.
2. Parse $\mathbf{p} = [\mathbf{p}_L^T | \mathbf{p}_{1,2}^T | \mathbf{p}_2^T]^T$ with $\mathbf{p}_L \in \mathcal{R}^{d\ell}, \mathbf{p}_{1,2} \in \mathcal{R}^d$ and $\mathbf{p}_2 \in \mathcal{R}^{d(k-\ell)}$.
3. $\mathbf{w} \leftarrow \mathfrak{t}^{-1}(\mathbf{u} - \mathbf{G}_L \mathbf{p}_L - \mathbf{A}\mathbf{p}_{1,2} - (\mathfrak{t}\mathbf{G}_H - (\mathbf{R}_1 + \mathbf{A}\mathbf{R}_2))\mathbf{p}_2) \bmod q\mathcal{R}$.
4. $\mathbf{z} \hookleftarrow \mathcal{D}_{\mathcal{L}_q^{\mathbf{w}}([\mathbf{G}_L | \mathbf{G}_H]), s_{\mathbf{G}}}$.
5. $\mathbf{v}' \leftarrow \mathbf{p} + \mathbf{L}\mathbf{z}$.
6. Parse $\mathbf{v}' = [\mathbf{v}_L^T | \mathbf{v}_{1,2}^T | \mathbf{v}_2^T]^T$ with $\mathbf{v}_L \in \mathcal{R}^{d\ell}, \mathbf{v}_{1,2} \in \mathcal{R}^d$ and $\mathbf{v}_2 \in \mathcal{R}^{d(k-\ell)}$.
7. $\mathbf{v}_{1,1} \leftarrow \mathbf{G}_L \mathbf{v}_L$.

Output: $\mathbf{v} = [\mathbf{v}_{1,1}^T | \mathbf{v}_{1,2}^T | \mathbf{v}_2^T]^T \in \mathcal{R}^{d(2+k-\ell)}$

The second type of samplers that we require is taken directly from the future post-quantum signature standard Falcon [PFH+20]. More precisely, we leverage almost all components of the Falcon signature scheme, namely its key generation, its preimage sampler, and parts of its verification algorithm. We thus simply give the description of the components we need and refer to the specifications [PFH+20] for more details.

Concretely, there exists an algorithm Falcon.KeyGen taking as input the description of a power-of-two cyclotomic ring $\mathcal{R}$, and a modulus p. This algorithm samples (f, g) from $\mathcal{D}_{\mathcal{R}^2, \sigma_{f,g}}$ until $f \bmod p\mathcal{R} \in \mathcal{R}_p^{\times}$ and that the final trapdoor reaches the target quality $\alpha = 1.17$. The algorithm outputs $h = gf^{-1} \bmod p\mathcal{R}$ defining the NTRU lattice $\mathcal{L}_q^{\perp}([1|h])$, and a basis $\mathbf{B}_F = \begin{bmatrix} g & G \\ -f & -F \end{bmatrix}$ such that $\|\mathbf{B}_F\|_{GS} \leq \alpha\sqrt{p}$ where $\|\mathbf{B}_F\|_{GS}$ denotes the Gram-Schmidt norm of $M_\tau(\mathbf{B}_F)$. Then, there also exists an algorithm Falcon.SamplePre which takes as input the NTRU lattice basis $\mathbf{B}_F$, the public key (or description of the NTRU lattice) h, a syndrome $u \in \mathcal{R}_p$, and a Gaussian parameter σ_F, and outputs a short Gaussian preimage $(x_1, x_2) \in \mathcal{R}^2$ such that $x_1 + hx_2 = u \bmod p\mathcal{R}$. That is, (x_1, x_2) is close to $\mathcal{D}_{\mathcal{L}_q^u([1|h]), \sigma_F}$ as stated in Lemma 2.6. This sampler is instantiated with the Fast Fourier Orthogonalization (FFO) sampler [DP16] in Falcon [PFH+20]. Since there is no known formal analysis of the latter, we instead use the Klein sampler [Kle00] which was thoroughly analyzed by Prest [Pre17]. We note that the Klein sampler analysis is expected to closely approximate the FFO sampler, which means the following result we use in the analysis of our EPID scheme should not differ significantly when using the FFO sampler.

Lemma 2.6 ([Pre17, Lem. 6] adapted). *Let $\mathcal{R}$ be a ring of algebraic integers of degree n, and p in $\mathbb{N}^{\times}$. Let $\varepsilon \in (0, 1/4)$ and let $\sigma_F \geq 1.17\eta_\varepsilon(\mathbb{Z}^{2n})\sqrt{p}$. Let $(h, \mathbf{B}_F) \leftarrow$ Falcon.KeyGen$(\mathcal{R}, p)$, $u \in \mathcal{R}_p$, and denote by $\mathcal{P}$ the output distribution of* Falcon.SamplePre$(\mathbf{B}_F, h, u, \sigma_F)$. *It holds that $\forall \mathbf{x} \in \mathcal{R}^2$, $\mathcal{P}(\mathbf{x}) \in$*

$[\delta^{-1}, \delta]\mathcal{D}_{\mathcal{L}_q^\perp([1|h]),\sigma_F}(\mathbf{x})$ *with* $\delta = ((1+\varepsilon/2n)/(1-\varepsilon/2n))^{2n}$. *In particular, we have* $\Delta(\mathcal{P}, \mathcal{D}_{\mathcal{L}_q^\perp([1|h]),\sigma_F}) \leq (\delta-1)/2 \approx \varepsilon$.

In our construction, we follow the parameter selection methodology from the Falcon specifications. We set $\sigma_{f,g} = \alpha\sqrt{p\pi/\deg(\mathcal{R})}$ as the Gaussian parameter[5] for f and g. The preimage sampling width is defined by $\sigma_F = \eta_{\varepsilon_F}(\mathbb{Z}^{2\deg(\mathcal{R})})\alpha\sqrt{p}$ so that $\sigma_F \geq \eta_{\varepsilon_F}(\mathcal{R}^2)\|\mathbf{B}_F\|_{GS} \geq \eta_{\varepsilon_F}(\mathcal{L}_p^\perp([1|h]))$ required by the sampler. We also need the following Gaussian regularity lemma from [GPV08], which we slightly adapt to match the context of NTRU lattices.

Lemma 2.7 ([GPV08, Lem. 5.2] adapted). *Let* p *be in* $\mathbb{N}^\times$ *and* $\mathcal{R}$ *a ring of algebraic integers of degree* n. *Let* $h \in \mathcal{R}_p$, $\varepsilon > 0$ *and* $\sigma \geq \eta_\varepsilon(\mathcal{L}_p^\perp([1|h]))$. *We define* $\mathcal{P} = [1|h]\mathcal{D}_{\mathcal{R}^2,\sigma} \bmod p\mathcal{R}$. *It holds that* $\forall r \in \mathcal{R}_p$, $\mathcal{P}(r) \in [(1-\varepsilon)/(1+\varepsilon), 1+\varepsilon]p^{-n}$. *In particular, we have* $\Delta(\mathcal{P}, U(\mathcal{R}_p)) \leq \varepsilon/(1+\varepsilon) \approx \varepsilon$.

2.5 Hardness Assumptions

Our EPID system is designed so that its security relies on well-known lattice assumptions, which we introduce now. More precisely, it relies on the hardness of *ring* and *module* versions of the *Learning With Errors* problem, denoted by R-LWE and M-LWE respectively, as well as the *Short Integer Solution* problem, denoted by R-SIS and M-SIS. These were formalized in [LPR10,LS15]. It also relies on the decisional NTRU problem introduced in [HPS98].

Definition 2.1 (M-LWE & R-LWE). *Let* $\mathcal{R} = \mathbb{Z}[x]/\langle x^n+1\rangle$ *with* n *a power-of-two. Let* d, m, k, q *be in* $\mathbb{N}^\times$ *and* $\mathscr{D}_s, \mathscr{D}_e$ *be two distributions on* $\mathcal{R}$. *The Module Learning With Errors problem* $\text{M-LWE}^k_{n,d,m,q,\mathscr{D}_s,\mathscr{D}_e}$ *asks to distinguish between the following distributions:* $(\mathcal{P}_1) \quad (\mathbf{A}, \mathbf{AS}+\mathbf{E} \bmod q\mathcal{R})$, *where* $\mathbf{A} \sim U(\mathcal{R}_q^{m\times d})$, $\mathbf{S} \sim \mathscr{D}_s^{d\times k}$ *and* $\mathbf{E} \sim \mathscr{D}_e^{m\times k}$, *and* $(\mathcal{P}_2) \quad (\mathbf{A}, \mathbf{B})$, *where* $\mathbf{A} \sim U(\mathcal{R}_q^{m\times d})$ *and* $\mathbf{B} \sim U(\mathcal{R}_q^{m\times k})$. *When* $k=1$, *we omit the superscript. When* $d=1$, *it corresponds to the Ring Learning With Errors problem and we thus use the acronym* $\text{R-LWE}_{n,m,q,\mathscr{D}_s,\mathscr{D}_e}$ *instead. We define the advantage* $\text{Adv}_{\text{M-LWE}}[\mathcal{A}]$ *(or* $\text{Adv}_{\text{R-LWE}}[\mathcal{A}]$ *for the ring case) as* $|\mathbb{P}[\mathcal{A}(\mathcal{P}_1)=1] - \mathbb{P}[\mathcal{A}(\mathcal{P}_2)=1]|$. *The search versions of the above simply asks to recover* $\mathbf{S}$ *from a sample from* $\mathcal{P}_1$.

Definition 2.2 (NTRU). *Let* $\mathcal{R} = \mathbb{Z}[x]/\langle x^n+1\rangle$ *with* n *a power-of-two. Let* p *be in* $\mathbb{N}^\times$ *and* $\mathscr{D}$ *a distribution on* $\mathcal{R}$. *The NTRU problem* $\text{NTRU}_{n,p,\mathscr{D}}$ *asks to distinguish between the distribution* $\mathcal{P}$ *of* $gf^{-1} \bmod p\mathcal{R}$, *where* $f \sim \mathscr{D}$ *conditioned on* $f \bmod p\mathcal{R} \in \mathcal{R}_p^\times$ *and* $g \sim \mathscr{D}$, *and the uniform distribution. The advantage is* $\text{Adv}_{\text{NTRU}}[\mathcal{A}] = |\mathbb{P}[\mathcal{A}(\mathcal{P})=1] - \mathbb{P}[\mathcal{A}(U(\mathcal{R}_p))=1]|$.

Definition 2.3 (M-SIS & R-SIS). *Let* $\mathcal{R} = \mathbb{Z}[x]/\langle x^n+1\rangle$ *with* n *a power-of-two. Let* d, m, q *be in* $\mathbb{N}^\times$ *with* $m > d$ *and* $\beta > 0$. *The Module Short Integer Solution problem* $\text{M-SIS}_{n,d,m,q,\beta}$ *asks to find* $\mathbf{x} \in \mathcal{L}_q^\perp([\mathbf{I}_d|\mathbf{A}]) \setminus \{\mathbf{0}\}$ *such that*

[5] We note that the specifications choose $\sigma_{f,g} = \alpha\sqrt{p/2\deg(\mathcal{R})}$ as their definition of discrete Gaussian differs from ours by a factor $\sqrt{2\pi}$ in the Gaussian parameter.

$\|\mathbf{x}\|_2 \leq \beta$, *given* $\mathbf{A} \hookleftarrow U(\mathcal{R}_q^{d\times m-d})$. *When* $d = 1$, *it corresponds to the Ring Short Integer Solution problem and we thus use the acronym* $\text{R-SIS}_{n,m,q,\beta}$ *instead. We define the advantage* $\text{Adv}_{\text{M-SIS}}[\mathcal{A}]$ *(or* $\text{Adv}_{\text{R-SIS}}[\mathcal{A}]$ *for the ring case) as* $\mathbb{P}_{\mathbf{x}\leftarrow\mathcal{A}(\mathbf{A})}\left[\mathbf{x} \in \mathcal{L}_q^{\perp}([\mathbf{I}_d|\mathbf{A}]) \wedge 0 < \|\mathbf{x}\|_2 \leq \beta\right]$.

When the parameters are clear from the context, we define the hardness bound of the problem $P \in \{\text{R-LWE}, \text{M-LWE}, \text{NTRU}, \text{R-SIS}, \text{M-SIS}\}$ as $\varepsilon_P = \sup_{\mathcal{A}\ \mathsf{PPT}} \text{Adv}_P[\mathcal{A}]$. We recall that we can use a standard hybrid argument to show that $\text{M-LWE}^k_{n,d,m,q,\mathscr{D}_s,\mathscr{D}_e}$ is at least as hard as $\text{M-LWE}^1_{n,d,m,q,\mathscr{D}_s,\mathscr{D}_e}$ at the expense of a loss factor k in the reduction.

3 Syntax and Security Model of EPID Systems

An EPID system can be seen as a dynamic group signature with a built-in revocation mechanism enabling to revoke users based on the anonymous signatures they emitted. This is done through signature revocation lists SRL whose management is more subtle than it appears. The core issue stems from the discrepancy between ideal SRLs and the ones used in practical constructions. Indeed, as its name suggests, a SRL should only contain a list of valid signatures. However, starting from the inital construction of Brickell and Li [BL07], the SRL has always been constituted of only parts of the revoked signatures, which offers the following opportunities for the adversary.

- **Malicious SRLs:** A malicious verifier could place on the SRLs ill-formed elements which can create security issues, as pointed out in [ST21], and would go unnoticed as it lacks the other components of the signature to run the verification algorithm[6]. This is why the original security model [BL07], followed by most constructions (e.g. [BEF19,KFM+19,CDK+24]), assumes that the SRL is managed by a trusted entity. In a few words, the trusted authority fills the gap between ideal SRLs and real SRLs as it ensures that the elements on the latter originate from valid signatures. However, this is very restrictive in practice and contradicts the decentralized spirit of EPID systems, leading [ST21] to propose a stronger model where the adversary can freely construct its SRLs.
- **Illicit Revocation:** An adversary trying to revoke an honest platform no longer needs to forge a full-fledge signature on its behalf but only the part that is placed on the SRL. This means that honest platforms are no longer protected by the usual unforgeability notion but need a non-frameability property identified in [CDK+24].

In this work, we follow the strong security model of [ST21] that we extend to include the non-frameability aspect mentioned above.

[6] Note that including the full signatures in the SRL would theoretically solve the problem but verifying each of them upon signature would hardly be acceptable in practice.

3.1 Syntax

An EPID system is a combination 8 algorithms involving three kinds of parties, namely an issuer $\mathcal{I}$, platforms (or users) $\mathcal{P}$, and verifiers $\mathcal{V}$.

- $\mathsf{Setup}(1^\lambda)$: takes as input the security parameter λ and returns the public parameters pp of the system.
- $\mathsf{GKeyGen}(\mathsf{pp})$: takes as input the public parameters pp and generates a key pair $(\mathsf{isk}, \mathsf{ipk})$ for the issuer $\mathcal{I}$. We assume ipk contains pp.
- $\mathsf{Join}\langle\mathcal{I}(\mathsf{isk}, \mathsf{ipk}), \mathcal{P}(\mathsf{ipk})\rangle$: interactive protocol between the issuer with $(\mathsf{isk}, \mathsf{ipk})$ and a platform $\mathcal{P}$ holding ipk. If the protocol did not abort prematurely, $\mathcal{I}$ receives $\perp$ while $\mathcal{P}$ receives a signing key sk. We denote it by $(\perp, \mathsf{sk}) \leftarrow \mathsf{Join}\langle\mathcal{I}(\mathsf{isk}, \mathsf{ipk}), \mathcal{P}(\mathsf{ipk})\rangle$.
- $\mathsf{KeyRevoke}(\{\mathsf{sk}_i\}_{i=1}^N)$: takes as input N platform signing keys and returns a key revocation list $\mathsf{KRL} = \{\mathsf{KRL}[i]\}_{i=1}^N$ containing N elements.
- $\mathsf{Sign}(\mathsf{ipk}, \mathsf{sk}, M, \mathsf{SRL})$: takes as input the issuer's public key ipk, a platform signing key sk, a message M, and a signature revocation list SRL, and returns an EPID signature sig or $\perp$.
- $\mathsf{SigRevoke}(\{\mathsf{sig}_i\}_{i=1}^N)$: takes as input N EPID signatures and returns a signature revocation list $\mathsf{SRL} = \{\mathsf{SRL}[i]\}_{i=1}^N$ containing N elements.
- $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL})$: takes as input a platform signing key sk and a signature revocation list SRL, and returns 1 if there exists i such that $\mathsf{SRL}[i]$ was generated using sk (i.e., sk is revoked), and 0 otherwise.
- $\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}, M, \mathsf{SRL}, \mathsf{KRL})$: takes as input the issuer's public key ipk, an EPID signature sig, a message M, a signature revocation list SRL and a key revocation list KRL, and returns 1 if sig is a valid signature on M for the corresponding revocation lists, and 0 otherwise.

Remark 3.1. The original security model by [BL07], followed by all subsequent works, considers a $\mathsf{KeyRevoke}$ algorithm and an associated list KRL which is filled with revoked secret keys. The way it is done in practice is however very unclear as a platform is the only entity knowing its secret key (and so is the only entity able to revoke it). We nevertheless choose to retain those elements in our model to remain compatible with previous EPID systems.

3.2 Security Model

Besides correctness, an EPID system requires anonymity (a platform is anonymous within the set of registered, unrevoked platforms), unforgeability (only registered, unrevoked platforms can produce valid signatures) and non-frameability (an honest platform cannot be revoked by a signature it did not generate). Our experiments assume malicious revocation lists, which naturally encompass the case of honest revocation lists.

For correctness, we want that for all signing keys sk lawfully obtained from the Join protocol, all KRL and SRL generated from $\mathsf{KeyRevoke}$ and $\mathsf{SigRevoke}$ respectively, it holds that

$$\mathsf{Verify}(\mathsf{ipk}, \mathsf{Sign}(\mathsf{ipk}, \mathsf{sk}, M, \mathsf{SRL}), M, \mathsf{SRL}, \mathsf{KRL}) = 1 \Leftrightarrow \mathsf{sk} \notin \mathsf{KRL} \ \wedge \ \mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 0$$

3.2.1 Anonymity. Anonymity is defined in Figure 3.1 and makes use of the three following oracles. We refer to [ST21] for the rationale behind this definition but simply recall that malicious revocation lists require a cautious formalization of the signature oracle as nothing prevents the adversary from building an SRL from elements of the challenge signature $\mathsf{sig}^\star$.

- $\mathcal{O}\mathsf{Join}_{hon}()$ is an oracle playing the platform's side of the Join protocol and is then used by $\mathcal{A}$, playing the issuer, to add a new honest platform. Each call generates a platform secret key sk that is kept secret by the challenger in a set $\mathsf{S}_{\mathsf{Join}}$, initially empty at the outset of the game.
- $\mathcal{O}\mathsf{Sign}^\star(\mathsf{SRL}, M)$ is an oracle used by $\mathcal{A}$ to query a signature on M from an honest platform that is not implicitly revoked by SRL. The challenger of the experiment randomly selects a signing key sk among those not revoked by SRL (i.e., the secret keys $\mathsf{sk}_i \in \mathsf{S}_{\mathsf{Join}}$ such that $\mathsf{Identify}(\mathsf{sk}_i, \mathsf{SRL}) = 0$), and returns $\mathsf{sig} = \mathsf{Sign}(\mathsf{ipk}, \mathsf{sk}, M, \mathsf{SRL})$. The challenger also stores the pair $(\mathsf{sig}, \mathsf{sk})$ in a table $\mathsf{T}_{\mathsf{Sign}}$. We store it in a hash table, i.e., with $\mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}] = \mathsf{sk}$, and $\mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}'] = \perp$ if sig' has never been emitted by a call to $\mathcal{O}\mathsf{Sign}^\star$.
- $\mathcal{O}\mathsf{Cor}^\star(\mathsf{sig})$ is an oracle that returns the signing key $\mathsf{sk} = \mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}]$. Note that if sig has not been generated by the above oracle, then $\mathsf{sk} = \mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}] = \perp$. Once the adversary has entered the challenge phase by returning two challenge signatures $\mathsf{sig}_0, \mathsf{sig}_1$, the oracle also returns $\perp$ if sig was generated from sk_0 or sk_1. This avoids (un)intentional failure of the adversary.

The EPID system is said ε-anonymous if the advantage of any PPT adversary $\mathcal{A}$ defined by $\mathrm{Adv}_{an}[\mathcal{A}] = \left|\mathbb{P}[\mathtt{Game}_{\mathcal{A}}^{an\text{-}1}(1^\lambda) = 1] - \mathbb{P}[\mathtt{Game}_{\mathcal{A}}^{an\text{-}0}(1^\lambda) = 1]\right|$ is upper-bounded by ε.

$\mathtt{Game}_{\mathcal{A}}^{an\text{-}\rho}(1^\lambda)$:
1 $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
2 $(\mathsf{isk}, \mathsf{ipk}) \leftarrow \mathsf{GKeyGen}(\mathsf{pp})$
3 $(\mathsf{sig}_0, \mathsf{sig}_1, M, \mathsf{SRL}) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Join}_{hon}, \mathcal{O}\mathsf{Sign}^\star, \mathcal{O}\mathsf{Cor}^\star}(\mathsf{isk})$
4 $(\mathsf{sk}_0, \mathsf{sk}_1) \leftarrow (\mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}_0], \mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}_1])$
5 **if** $[\![\perp \in \{\mathsf{sk}_0, \mathsf{sk}_1\}]\!] \vee [\![\mathsf{Identify}(\mathsf{sk}_0, \mathsf{SRL}) = 1]\!] \vee [\![\mathsf{Identify}(\mathsf{sk}_1, \mathsf{SRL}) = 1]\!]$ **return** 0
6 $\mathsf{sig}^\star \leftarrow \mathsf{Sign}(\mathsf{ipk}, \mathsf{sk}_\rho, M, \mathsf{SRL})$
7 $\rho' \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Join}_{hon}, \mathcal{O}\mathsf{Sign}^\star, \mathcal{O}\mathsf{Cor}^\star}(\mathsf{isk}, \mathsf{sig}^\star)$
8 **if** sk_0 or sk_1 leaked through a query to $\mathcal{O}\mathsf{Cor}^\star$, **return** 0
9 **return** $[\![\rho = \rho']\!]$

Fig. 3.1. Anonymity Game for EPID Systems for $\rho \in \{0, 1\}$

3.2.2 Unforgeability. The game in Figure 3.2 then defines the unforgeability notion according to [ST21]. It makes use of four oracles which we define here. We denote by ctr_{cor} (resp. ctr_{sig}) a counter for the number of corrupt platforms created by $\mathcal{A}$ (resp. signatures issued by $\mathcal{A}$) at the current time, initially set at

0. The informal goal of the adversary is to produce signatures despite successive revocation of its keys.

- $\mathcal{O}\mathsf{Add}(i)$ is an oracle used by $\mathcal{A}$ to add a new honest platform i. The oracle plays both sides of the Join protocol and generates a new signing key sk_i, but nothing is returned to $\mathcal{A}$. The challenger stores a hash table $\mathsf{T}_{\mathsf{Join}}$ such that $\mathsf{T}_{\mathsf{Join}}[i] = \mathsf{sk}_i$ (and $\mathsf{T}_{\mathsf{Join}}[j] = \perp$ if j was never queried). Note that each i can only be queried once.
- $\mathcal{O}\mathsf{Join}_{cor}(i)$ is an oracle playing the issuer side of the Join protocol. It is used by $\mathcal{A}$ to add a corrupt platform, for which it receives the signing key sk at the end of the execution. Each call to this oracle increments the current value of ctr_{cor} by 1 and sets $\mathsf{T}_{\mathsf{Join}}[i] = \top$.
- $\mathcal{O}\mathsf{Sign}(i, \mathsf{SRL}, M)$ is an oracle used by $\mathcal{A}$ to query platform i for a signature on M and signature revocation list SRL. The oracle returns $\mathsf{sig} = \perp$ if $\mathsf{T}_{\mathsf{Join}}[i] \in \{\perp, \top\}$ (meaning the platform does not exist or is corrupted). Otherwise, it runs $\mathsf{sig} = \mathsf{Sign}(\mathsf{ipk}, \mathsf{T}_{\mathsf{Join}}[i], M, \mathsf{SRL})$ and returns sig. The generated signature sig is stored by the challenger in a set $\mathsf{S}_{\mathsf{Sign}}$, initially empty.
- $\mathcal{O}\mathsf{Cor}(i)$ is an oracle that returns the signing key $\mathsf{sk}_i = \mathsf{T}_{\mathsf{Join}}[i]$ or aborts if $\mathsf{T}_{\mathsf{Join}}[i] \in \{\perp, \top\}$. The challenger then sets $\mathsf{T}_{\mathsf{Join}}[i] = \top$ and stores the corrupted platform key sk_i in a set S_{cor} (initially empty). It also increments ctr_{cor} by 1.

The EPID system is said ε-unforgeable if the advantage of any PPT adversary $\mathcal{A}$ defined by $\mathrm{Adv}_{unf}[\mathcal{A}] = \mathbb{P}[\mathtt{Game}^{unf}_{\mathcal{A}}(1^\lambda) = 1]$ is upper-bounded by ε. Note that in the game below, we assume that the while loop aborts after some polynomial number of iterations, in which case the game returns 0.

$\mathtt{Game}^{unf}_{\mathcal{A}}(1^\lambda)$:
1 $\mathsf{ctr}_{cor}, \mathsf{ctr}_{sig} \leftarrow 0$
2 $\mathsf{T}_{\mathsf{Join}}, \mathsf{S}_{\mathsf{Sign}}, \mathsf{S}_{cor}, \mathsf{S}_{\mathcal{A}} \leftarrow \varnothing$
3 $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
4 $(\mathsf{isk}, \mathsf{ipk}) \leftarrow \mathsf{GKeyGen}(\mathsf{pp})$
5 **while** $[\![\mathsf{ctr}_{sig} \leq \mathsf{ctr}_{cor}]\!]$ **do**
6 $\quad \mathsf{SRL} \leftarrow \mathsf{SigRevoke}(\mathsf{S}_{\mathcal{A}})$
7 $\quad (\mathsf{sig}, M) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Add}, \mathcal{O}\mathsf{Join}_{cor}, \mathcal{O}\mathsf{Sign}, \mathcal{O}\mathsf{Cor}}(\mathsf{SRL}, \mathsf{ipk})$
8 $\quad \mathsf{KRL} \leftarrow \mathsf{KeyRevoke}(\mathsf{S}_{cor})$
9 $\quad$ **if** $[\![\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}, M, \mathsf{SRL}, \mathsf{KRL}) = 1]\!] \wedge [\![\mathsf{sig} \notin \mathsf{S}_{\mathsf{Sign}} \cup \mathsf{S}_{\mathcal{A}}]\!]$
10 $\quad\quad (\mathsf{ctr}_{sig}, \mathsf{S}_{\mathcal{A}}) \leftarrow (\mathsf{ctr}_{sig} + 1, \mathsf{S}_{\mathcal{A}} \cup \{\mathsf{sig}\})$
11 **return** 1

Fig. 3.2. Unforgeability Game for EPID Systems

3.2.3 Non-Frameability. The unforgeability notion above ensures that revoked users can no longer produce valid signatures. It must however be balanced

by a non-frameability notion that ensures that the revocation procedure does not induce collateral damages. In particular, placing elements of a signature sig on a SRL should only revoke the platform that emitted sig and not any other platforms. Otherwise, an adversary could trigger illicit revocations, as discussed above. For clarity of presentation, and for the security proofs, we denote by $i^\star$ the index of the framed honest platform.

This is formalized by the non-frameability game in Figure 3.3 which uses the same oracles as the unforgeability game. The EPID system is said ε-non-frameable if the advantage of any PPT adversary $\mathcal{A}$ defined by $\mathrm{Adv}_{nfra}[\mathcal{A}] = \mathbb{P}[\mathtt{Game}^{nfra}_{\mathcal{A}}(1^\lambda) = 1]$ is upper-bounded by ε.

$\mathtt{Game}^{nfra}_{\mathcal{A}}(1^\lambda)$:
1 $\mathsf{T}_{\mathsf{Join}}, \mathsf{S}_{cor} \leftarrow \varnothing$
2 $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
3 $(\mathsf{isk}, \mathsf{ipk}) \leftarrow \mathsf{GKeyGen}(\mathsf{pp})$
4 $(M, \mathsf{sig}, \mathsf{SRL}) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Add}, \mathcal{O}\mathsf{Join}_{cor}, \mathcal{O}\mathsf{Sign}, \mathcal{O}\mathsf{Cor}}(\mathsf{isk}, \mathsf{ipk})$
5 $\mathsf{KRL} \leftarrow \mathsf{KeyRevoke}(\mathsf{S}_{cor})$
6 **if** $[\![\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}, M, \mathsf{SRL}, \mathsf{KRL}) = 0]\!]$ **return** 0
7 **for all** $i^\star$ such that $\mathsf{T}_{\mathsf{Join}}[i^\star] \notin \{\top, \bot\}$ **do**
8 $\quad \mathsf{sk}_{i^\star} \leftarrow \mathsf{T}_{\mathsf{Join}}[i^\star]$
9 $\quad$ **if** $[\![\mathsf{Identify}(\mathsf{sk}_{i^\star}, \mathsf{SigRevoke}(\{\mathsf{sig}\})) = 1]\!]$ **and** sig was never obtained from a query $\mathcal{O}\mathsf{Sign}(i^\star, *, *)$ **return** 1
10 **return** 0

Fig. 3.3. Non-Frameability Game for EPID Systems

4 Lattice EPID with Compact Proof of Non-Revocation

In this section, we present a new construction of lattice EPID. We recall that an EPID signature essentially consists of two parts. The first one is a proof of registration, where the platform usually proves knowledge of a certificate received from the issuer, in the same vein as group signatures or anonymous credentials. It is quite standard and we essentially follow the blueprint of [AGJ+24], with some optimizations from [JS25a,JS25b] and adaptations to be compatible with the rest of our signature. The second part is dedicated to the proof of non-revocation which is the very core (and specificity) of EPID systems. As we explained in the introduction, our proof of non-revocation differs from previous approaches by its low reliance on zero-knowledge proofs, which creates new challenges that need to be addressed carefully.

4.1 High Level Description

4.1.1 Proof of Registration. The issuer, in charge of registering platforms, generates a key pair for the signature scheme of [AGJ+24,JS25b]. More precisely, it generates short matrices $\mathbf{R}_1, \mathbf{R}_2$ and computes the public key $\mathsf{pk} = \mathbf{B} = \mathbf{R}_1 + \mathbf{A}\mathbf{R}_2$, for a public matrix $\mathbf{A}$. For the enrolment phase (Join protocol), the platform $\mathcal{P}$ starts by choosing a short secret value s and embeds it into a vector $\mathbf{s}$. We describe later why we use such an embedding and not s itself, but the main reason is that it allows for more modularity and a more efficient scheme in the end. It then commits to $\mathbf{s}$ through $\mathbf{c} = \mathbf{r}_1 + \mathbf{A}\mathbf{r}_2 + \mathbf{D}\mathbf{s}$ for random $\mathbf{r}_1, \mathbf{r}_2$. As usual, it also generates a proof π_1 of well-formedness of this commitment before sending $\mathbf{c}$ and π_1 to the issuer $\mathcal{I}$. The latter then verifies that π_1 is valid and proceeds to generate a signature on $\mathbf{c}$ as in [AGJ+24]. Concretely, it selects a registration tag t and uses $(\mathbf{R}_1, \mathbf{R}_2)$ to find a short preimage $(\mathbf{v}_1', \mathbf{v}_2, \mathbf{v}_3)$ such that $[\mathbf{I}_d|\mathbf{A}]\mathbf{v}_1' + (\mathsf{t}\mathbf{G}_H - \mathbf{B})\mathbf{v}_2 + \mathbf{A}_3\mathbf{v}_3 = \mathbf{u} + \mathbf{c}$ using the recent sampler of [JS25b] recalled in Algorithm 2.1 in Section 2.4. The platform $\mathcal{P}$ then checks whether this partial signature passes verification, and finally completes it by computing $\mathbf{v}_1 = \mathbf{v}_1' - [\mathbf{r}_1|\mathbf{r}_2]^T$. That is that $[\mathbf{I}_d|\mathbf{A}]\mathbf{v}_1 + (\mathsf{t}\mathbf{G}_H - \mathbf{B})\mathbf{v}_2 + \mathbf{A}_3\mathbf{v}_3 = \mathbf{u} + \mathbf{D}\mathbf{s}$, with $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ short. The secret key of $\mathcal{P}$ is then $\mathsf{sk} = (s, \mathsf{t}, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$, i.e., the chosen secret s and the signature on (the embedding of) said secret obtained blindly in this interactive protocol.

Looking ahead, sk provides the platform with the ability to prove that its secret key s has been certified and so that 1) the platform has been registered by $\mathcal{I}$ and 2) the revocation tag tag was indeed produced with a registered key. We now explain how our new proofs of non-revocation work.

4.1.2 Proof of Non-Revocation. The starting point of the proof of non-revocation of EPID systems is the tag tag that each platform $\mathcal{P}$ must add to its signature to facilitate revocation. In our system, it is built as $\mathsf{tag} = \mathbf{a} \cdot s + \mathbf{e}$ where $\mathbf{e}$ is a short error and $\mathbf{a} = \mathcal{H}(\mathsf{inp}) \in \mathcal{R}_p^m$ for some input inp that is left unspecified for the moment. We note that the secret used here is the originally sampled one s, and not its embedding $\mathbf{s}$ signed in the Join protocol. However, we will see that the embedding is chosen so that it is easy to prove the two are related in a subsequent zero-knowledge proof without having to commit extra values. After having computed tag, the platform starts generating the non-revocation proof against the signature revocation list SRL at hand. This list contains elements inp_i and tag_i from signatures that have been revoked by a verifier (to which the platform tries to authenticate).

The overall approach of EPID systems, recalled in Figure 1.1, is to use for each i a token t_i that is close to a randomized version $\widetilde{\mathsf{tag}_i}$ of $\mathsf{tag}_i = \mathbf{a}_i \cdot s_i + \mathbf{e}_i$ if, and only if, they have been produced with the same secret key. In our case, tag randomization is done using public, short $\boldsymbol{\gamma}_i = \mathcal{H}'(\mathsf{inp}, \mathsf{inp}_i) \in \mathcal{R}^m$ and a single element t, depending on s, can be used to recover (approximately) all the tokens t_i. Concretely, let us assume that h is an element with a trapdoor allowing to generate a short x_i such that $x_i h$ is close to the element $\boldsymbol{\gamma}_i^T \mathbf{a}_i$ defined above. Then, providing $t = hs + e$ and x_i is essentially equivalent to providing

$t_i = \boldsymbol{\gamma}_i^T \mathbf{a}_i \cdot s + e_i$ as $x_i t$ is close to the latter value. Moreover, anyone can check the validity of x_i as h, $\boldsymbol{\gamma}_i$ and $\mathbf{a}_i$ are public. At this stage, one might wonder if it would not be simpler to provide the trapdoor instead of all x_i. Unfortunately, this would provide too much power to the verifier that could, for example, use it to find short pre-images of 1 and thus map t on $1s + e'$ for some short e'.

What we need now is to find the right instantiation for h and the associated trapdoors. In this paper, we choose to follow the Falcon approach for its compactness in size and dimension, and thus set h as a Falcon public key. Thanks to the associated secret key, one can sample, for every element in the SRL, a short preimage $(x_{i,1}, x_{i,2})$ such that $x_{i,1} + x_{i,2}h = \boldsymbol{\gamma}_i^T \mathbf{a}_i$. Actually, we only need $x_{i,2}$ as we only want to find an approximate preimage of $\boldsymbol{\gamma}_i^T \mathbf{a}_i$. We have thus traded uniform t_i for short, Gaussian $x_{i,2}$ and replace NIZK proofs that all t_i are valid by a NIZK proof that a unique element (t) is valid, hence the improvement over previous approaches. We also note that our approach is quite generic, and that any improvement in the design of lattice trapdoors could yield more efficient EPID constructions.

To verify the signature is valid and not revoked, the verifier checks validity of the zero-knowledge proof and of the Falcon signatures $x_{i,2}$ on $\boldsymbol{\gamma}_i^T \mathbf{a}_i$ (the latter can be computed from the revocation list and inp), and that $x_{i,2}t - \boldsymbol{\gamma}_i^T \mathsf{tag}_i$ is large. Indeed, we have

$$\begin{aligned} x_{i,2}t - \boldsymbol{\gamma}_i^T \mathsf{tag}_i &= (x_{i,2}h)s + x_{i,2}e' - \boldsymbol{\gamma}_i^T(\mathbf{a}_i s_i + \mathbf{e}_i) \\ &= (\boldsymbol{\gamma}_i^T \mathbf{a}_i - x_{i,1})s + x_{i,2}e' - \boldsymbol{\gamma}_i^T \mathbf{a}_i s_i - \boldsymbol{\gamma}_i^T \mathbf{e}_i \\ &= \boldsymbol{\gamma}_i^T \mathbf{a}_i (s - s_i) + (x_{i,2}e' - x_{i,1}s - \boldsymbol{\gamma}_i^T \mathbf{e}_i), \end{aligned}$$

which is small if $s = s_i$.

4.1.3 Proving Security. Now that we have described the generation of the tokens t_i, we need to consider more thoroughly the generation of the vector $\mathbf{a}$ as $\mathcal{H}(\mathsf{inp})$. At first sight, it seems that we only need $\mathbf{a}$ to be fresh for each signature, which would plead for defining inp as some random seed. Unfortunately, this would yield a totally insecure scheme. Indeed, after seeing $\mathsf{tag} = \mathbf{a}s + \mathbf{e}$ generated by a platform $\mathcal{P}$, an adversary could place $(\mathsf{inp}, \mathsf{tag}' = \mathsf{tag} + \mathbf{u})$ on a SRL, for some large uniform $\mathbf{u}$. It would thus trick $\mathcal{P}$ into producing t and x_2 such that $x_2 t$ would be close to $\boldsymbol{\gamma}^T \mathsf{tag}$. By removing the mask $\boldsymbol{\gamma}^T \mathbf{u}$, the attacker could then recover $x_2 t - \boldsymbol{\gamma}^T \mathsf{tag} = x_2 e' - x_1 s - \boldsymbol{\gamma}^T \mathbf{e}$, which only involves small elements and thus does not hide s. It is thus crucial for security that a platform never produces a signature for a SRL including a string inp that it has already used.

Here, one could think that the problem could be addressed by providing a way for the platform to recognize the element inp it already used. A trivial, but inefficient solution, would be to locally store every inp used by $\mathcal{P}$. Alternatively, $\mathcal{P}$ could define inp as the output of some PRF keyed with s and thus refuse to produce a signature if some element on the SRL matches this description. Without appropriate safeguards, this would however introduce new vulnerabilities since this test could be leveraged by an adversary against non-frameability.

Concretely, an adversary could produce a signature sig using the same inp as an honest platform $\mathcal{P}$ and then incite an honest[7] verifier into revoking sig (e.g., by misbehaving). This way, the verifier would place elements related to sig (and thus inp) on its SRL, leading $\mathcal{P}$ to systematically abort when producing a signature.

We then need to balance our construction of inp to ensure that an honest platform will never produce a signature if the SRL includes one it has already used while avoiding the denial of service we sketched above. To this end, we include in inp an element c built as $\mathcal{H}_1(\mathsf{seed}) \cdot s + e$ for some fresh seed seed and short e and require a proof that it is well-formed. This element c enables the platform to detect if it has already use $\mathsf{inp} = (\mathsf{seed}, c)$, in which case it aborts. At the same time, the requested proof of well-formedness prevents an adversary from mounting the attack above, as will be demonstrated in our security analysis. Note that this solution only adds a single element c to the signature and proof, regardless of the size of the SRL. Actually, we could even recycle t for this purpose but then the reduction would need some hybrid NTRU-SIS assumption that we prefer to avoid.

We have therefore defined the input inp of the hash function to generate $\mathbf{a}$ but we still have to define the output. In our security reduction, we will have to replace at some point t by an RLWE sample, thus removing the trapdoor we need to generate the $x_{i,2}$. To address that, we essentially swap the roles of h (resp. $(x_{i,1}, x_{i,2})$) and $\mathbf{a}_i$ (resp. $\boldsymbol{\gamma}_i$) and thus hide a trapdoor in $\mathbf{a}_i$, which imposes some distribution on this vector and on $\boldsymbol{\gamma}_i$. Here, one might wonder why the symmetry between the roles of h (resp. $(x_{i,1}, x_{i,2})$) and $\mathbf{a}_i$ (resp. $\boldsymbol{\gamma}_i$) is not reflected by their structure and dimensions. In particular, why can't we reduce the dimension of $\mathbf{a}_i$ and simply use some polynomial h_i that would be distributed as a Falcon public key? The reason is that we need to compute exact preimages with $\mathbf{a}_i$ because it is eventually multiplied by adversarially chosen s_i, which can be large (the well-formedness of the elements in the SRL is not checked). In particular, we cannot use approximate preimages (as we do by only providing $x_{i,2}$ for h) because the error could blow up when multiplied by s_i. We will discuss further the distribution of $\mathbf{a}_i$ and $\boldsymbol{\gamma}_i$ in Section 6.

4.2 Leveraging Subrings for Long-Term Secret

It now remains to explain the motivation behind the embedding of the long-term secret s into a vector $\mathbf{s}$. The reason is that we need to reconcile the two parts of the EPID system, namely the proof of registration and the proof of non-revocation, which have very different constraints. In particular, the latter requires to select parameters that (1) allow for distinguishing the two cases $s_i = s$ and $s_i \neq s$ (which entails some minimal bound on the modulus p), while (2) ensuring that the secret s is well hidden in all the given information (tag, t, c). The latter is argued using the R-LWE assumption whose hardness decreases as

[7] We only seek to prevent honest verifiers from mistakenly revoking $\mathcal{P}$. This property would become meaningless with malicious verifiers as they could achieve the same goal by merely placing signatures from $\mathcal{P}$ on the SRL

p grows. This leads us to work with a ring $\mathcal{R}_1$ of much larger degree than the one (let us call it $\mathcal{R}_2$) used for the Join protocol (registration). Nevertheless, both of these rings need to be somehow linked in the zero-knowledge proof π_2 to argue that the secret used in the proof of non-revocation is indeed the same as the one signed by the issuer in the Join protocol.

In [AGJ+24], and subsequent constructions [JS25a,JS25b], the zero-knowledge proof π_2 is carried over a subring of $\mathcal{R}_2$, which we denote by $\mathcal{R}_3$. The relation over $\mathcal{R}_2$ is embedded into $\mathcal{R}_3^{k_{2,3}}$ (with $k_{2,3} = \deg(\mathcal{R}_2)/\deg(\mathcal{R}_3)$) using a well-defined subring embedding $\theta_{2,3}$ that also maps ring operations of $\mathcal{R}_2$ to matrix-vector operations of $\mathcal{R}_3^{k_{2,3}}$. In our system featuring an extra ring for the proof of non-revocation, we can also embed the relations from $\mathcal{R}_1$ to $\mathcal{R}_3^{k_{1,3}}$ (with $k_{1,3} = \deg(\mathcal{R}_1)/\deg(\mathcal{R}_3)$) with an embedding $\theta_{1,3}$. The subtlety then comes in the fact that if $s \in \mathcal{R}_1$, we need to embed it into $\mathbf{s} = \widetilde{\theta}_{1,2}(s) \in \mathcal{R}_2^{k_{1,2}}$ in such a way that

$$\theta_{2,3}(\widetilde{\theta}_{1,2}(s)) = \theta_{2,3}(\mathbf{s}) = \theta_{1,3}(s), \tag{1}$$

so as to efficiently prove consistency of the secret between the relations in $\mathcal{R}_1$ and in $\mathcal{R}_2$. In [LNPS21,AGJ+24], these embeddings $\theta_{1,3}, \theta_{2,3}$ are defined so that ring operations map in a natural way. However, if we define $\widetilde{\theta}_{1,2}$ in the same way, Equation (1) only holds up to a permutation of the coefficient. We thus proceed differently and define $n_i = \deg(\mathcal{R}_i)$ for $i \in \{1,2,3\}$, and then $\theta_{i,3}$ and $\widetilde{\theta}_{1,2}$ as follows.

$$\begin{array}{rccc} \theta_{i,3}: & \mathcal{R}_i & \longrightarrow & \mathcal{R}_3^{k_{i,3}} \\ & a = \sum_{j=0}^{n_i-1} a_j x^j & \longmapsto & \begin{bmatrix} \sum_{j=0}^{n_3-1} a_{k_{i,3}j+0} \cdot x^j \\ \vdots \\ \sum_{j=0}^{n_3-1} a_{k_{i,3}j+(k_{i,3}-1)} \cdot x^j \end{bmatrix} \end{array}$$

and

$$\begin{array}{rccc} \widetilde{\theta}_{1,2}: & \mathcal{R}_1 & \longrightarrow & \mathcal{R}_2^{k_{1,2}} \\ & a = \sum_{j=0}^{n_1-1} a_j x^j & \longmapsto & \begin{bmatrix} \sum_{j=0}^{n_2-1} a_{f_0(j)} \cdot x^j \\ \vdots \\ \sum_{j=0}^{n_2-1} a_{f_{k_{1,2}-1}(j)} \cdot x^j \end{bmatrix} \end{array}$$

where $f_l(j) = k_{1,3}\lfloor j/k_{2,3} \rfloor + l \cdot k_{2,3} + (j - k_{2,3}\lfloor j/k_{2,3} \rfloor)$ for $l \in [0, k_{1,2} - 1]$ and $j \in [0, n_2 - 1]$. We can then verify that $\theta_{2,3} \circ \widetilde{\theta}_{1,2} = \theta_{1,3}$ thus satisfying Equation (1). As a result, we only need s in the witness ($\theta_{1,3}(s)$ to be specific), which avoids an extra commitment to $\mathbf{s}$ and a proof of consistency between $\mathbf{s}$ and s. Also, note that $\theta_{i,3}$ as well as $\widetilde{\theta}_{1,2}$ are obtained by mere selection of the coefficients of the original ring element. It means they do not change their coefficient embedding, up to a permutation of course. In particular, if $s \in \mathcal{R}_1$ has ternary coefficients in $\{-1, 0, 1\}$, then so will $\theta_{1,3}(s)$ and $\widetilde{\theta}_{1,2}(s)$ and vice-versa.

This embedding method gives us the freedom of using different ring degrees (and thus lattice dimensions) for the signature in Join and the proof of non-revocation. The other natural solution could have been to simply use the largest

ring for both parts, but doing so would overshoot the security in the less demanding part. Our embedding method bypasses this constraint and allows for a tighter choice of parameters and leads to more efficient sizes.

4.3 The Scheme

We now give a description of the scheme. To avoid ambiguity, we use τ_i to denote the coefficient embedding of $\mathcal{R}_i$, that is mapping an element of $\mathcal{R}_i$ to its coefficient vector of $\mathbb{Z}^{n_i}$. Also, for each ring $\mathcal{R}_i$, we define $\mathcal{R}_{i,q} = \mathcal{R}_i/q\mathcal{R}_i$ for a positive integer q. All the rings $\mathcal{R}_i$ are power-of-two cyclotomic rings of degree n_i. The verification bounds $B'_{1,1}, B'_{1,2}, B_2, B_3, \beta_F, \beta$ are all defined and discussed after the description of the scheme in Section 4.3.1.

Algorithm 4.1: Setup(1^λ)

Input: Security parameter λ.

Registration Parameters.

1. Choose $d \in \mathbb{N}^\times$
2. Choose an odd prime q s.t. $q = 5 \bmod 8$
3. Choose $w \in \mathbb{N}^\times$ s.t. $\binom{n_2}{w} \geq Q_{\mathsf{Join}}$
4. Choose $b \in \mathbb{N}^\times \cap [2, \sqrt{q}]$
5. $\mathcal{T}_w \leftarrow \{\mathfrak{t} \in \tau_2^{-1}(\{0,1\}^{n_2}) : \|\mathfrak{t}\|_1 = w\}$
6. $k \leftarrow \lceil \log_b q \rceil$
7. Choose $\ell < k$
8. $B_R \leftarrow \frac{3}{4}(\sqrt{n_2 d} + \sqrt{n_2 d(k-\ell)} + 6)$
9. $\mathbf{G}_L \leftarrow [\mathbf{I}_d | b\mathbf{I}_d | \dots | b^{\ell-1}\mathbf{I}_d]$
10. $\mathbf{G}_H \leftarrow [b^\ell \mathbf{I}_d | \dots | b^{k-1}\mathbf{I}_d]$
11. Choose $\varepsilon \in (0,1)$ ▷ (typically $\varepsilon = 2^{-40}$)
12. $r \leftarrow \sqrt{\ln(2n_2 dk/\varepsilon)/\pi}$ ▷ ($\approx \eta_\varepsilon(\mathcal{R}_2^{dk})$)
13. $s_{\mathbf{G}} \leftarrow r\sqrt{b^2+1}$
14. $s_1 \leftarrow r(b+b^{-1})\sqrt{4w^2 + 3B_R^2}$
15. $s_2 \leftarrow r(b+b^{-1}) \cdot 2w$
16. $s_{1,1} \leftarrow r(b+b^{-1})\sqrt{4w^2(b^{2\ell}-1)/(b^2-1) + 3B_R^2}$
17. $s_3 \leftarrow \sqrt{3}r(b+b^{-1})B_R$
18. $s_4 \leftarrow \sqrt{3}r(b+b^{-1})$
19. $(\alpha_{1,1}, \alpha_{1,2}) \leftarrow (s_{1,1}/(n_2\sqrt{dk_{1,2}/2} + \sqrt{n_2 d}), s_3/(n_2\sqrt{dk_{1,2}/2} + \sqrt{n_2 d}))$
20. $M_{1,i} \leftarrow \exp(\pi/\alpha_{1,i}^2)$ for $i \in [2]$
21. Select hash functions $\mathcal{G}_1 : \{0,1\}^{256} \to \mathcal{R}_{2,q}^{d\times d}$, $\mathcal{G}_2 : \{0,1\}^{256} \to \mathcal{R}_{2,q}^{d\times(k-\ell)}$, $\mathcal{G}_3 : \{0,1\}^{256} \to \mathcal{R}_{2,q}^{d}$ and $\mathcal{G}_4 : \{0,1\}^{256} \to \mathcal{R}_{2,q}^{d\times k_{1,2}}$
22. $\mathsf{seed}_{\mathsf{pp}} \hookleftarrow U(\{0,1\}^{256})$
23. $\mathbf{A} \leftarrow \mathcal{G}_1(\mathsf{seed}_{\mathsf{pp}})$ ▷ follows $U(\mathcal{R}_{2,q}^{d\times d})$
24. $\mathbf{A}_3 \leftarrow \mathcal{G}_2(\mathsf{seed}_{\mathsf{pp}})$ ▷ follows $U(\mathcal{R}_{2,q}^{d\times k-\ell})$
25. $\mathbf{u} \leftarrow \mathcal{G}_3(\mathsf{seed}_{\mathsf{pp}})$ ▷ follows $U(\mathcal{R}_{2,q}^{d})$
26. $\mathbf{D} \leftarrow \mathcal{G}_4(\mathsf{seed}_{\mathsf{pp}})$ ▷ follows $U(\mathcal{R}_{2,q}^{d\times k_{1,2}})$

Proof of Non-Revocation Parameters.

27. Choose $\eta \in \mathbb{N}^\times$ s.t. $\binom{n_1+2\eta+1}{2\eta+1} > n_1(4Q_{\mathsf{Sign}} + Q_{\mathcal{H}_1})$
28. Choose $\varepsilon_F \in (0,1)$ ▷ (typically $\varepsilon_F = 2^{-\lambda}/\Omega(N_{\mathsf{SRL}}Q_{\mathsf{Sign}})$)
29. $r_F \leftarrow \sqrt{\ln(4n_1/\varepsilon_F)/\pi}$ ▷ $\approx \eta_{\varepsilon_F}(\mathcal{R}_1^2)$

30. $\alpha \leftarrow 1.17$
31. Choose a prime p s.t. $p = \Omega(16\alpha^2 r_F^2 n_1^2(1+3\eta^2))$ and $p = 5 \bmod 8$
32. $\sigma_{f,g} \leftarrow \alpha\sqrt{\pi p/n_1}$
33. $\sigma_F \leftarrow r_F \alpha\sqrt{p}$
34. Select hash functions $\mathcal{H}_1 : \{0,1\}^{256} \to \mathcal{R}_{1,p}$, $\mathcal{H}_2 : \mathcal{R}_{1,p} \times \{0,1\}^{256} \to \tau_1^{-1}([-\eta,\eta]^{n_1})$, $\mathcal{H}_3 : \{0,1\}^{256} \times \mathcal{R}_{1,p} \to \mathcal{R}_{1,p}^2$ and $\mathcal{H}_4 : \{0,1\}^{256} \times \mathcal{R}_{1,p} \times \{0,1\}^{256} \times \mathcal{R}_{1,p} \to \mathcal{R}_1^2$, s.t. the output of $\mathcal{H}_4$ is close to the discrete Gaussian $\mathcal{D}_{\mathcal{R}_1^2,\sigma_F}$ truncated in Euclidean norm at β_F.

Output: $\mathsf{pp} = (\lambda, n_1, n_2, d, q, w, b, k, \ell, B_R, r, s_{\mathbf{G}}, s_1, s_2, s_3, s_4, \alpha_{1,i}, M_{1,i}, \eta, r_F, \alpha, p, \sigma_{f,g}, \sigma_F, \mathsf{seed}_{\mathsf{pp}})$

Algorithm 4.2: GKeyGen(pp)

Input: Public parameters pp as in Algorithm 4.1.

1. $\mathbf{R}_1, \mathbf{R}_2 \hookleftarrow \mathcal{B}_1^{d \times d(k-\ell)}$ conditioned on $\|\mathbf{R}_i\|_2 \le B_R$
2. $\mathbf{B} \leftarrow \mathbf{R}_1 + \mathbf{A}\mathbf{R}_2 \bmod q\mathcal{R}_2 \in \mathcal{R}_{2,q}^{d \times d(k-\ell)}$

Output: $\mathsf{ipk} = (\mathbf{B}, \mathsf{pp})$, and $\mathsf{isk} = (\mathbf{R}_1, \mathbf{R}_2)$

Algorithm 4.3: Join$\langle \mathcal{I}(\mathsf{isk}, \mathsf{ipk}), \mathcal{P}(\mathsf{ipk}) \rangle$

Platform $\mathcal{P}$.

1. $s \hookleftarrow U(\tau_1^{-1}(\{-1,0,1\}^{n_1}))$. ▷ $s \in \mathcal{R}_1$
2. $\mathbf{s} \leftarrow \widetilde{\theta}_{1,2}(s)$. ▷ $\mathbf{s} \in \mathcal{R}_2^{k_{1,2}}$
3. $\mathbf{r}_1, \mathbf{r}_2 \hookleftarrow U(\tau_2^{-1}(\{0,1\}^{n_2})^d)$ ▷ $\mathbf{r}_1, \mathbf{r}_2 \in \mathcal{R}_2^d$
4. $\mathbf{c} \leftarrow \mathbf{r}_1 + \mathbf{A}\mathbf{r}_2 + \mathbf{D} \cdot \mathbf{s} \bmod q\mathcal{R}_2$ ▷ $\mathbf{c} \in \mathcal{R}_{2,q}^d$
5. $\pi_1 \leftarrow \mathsf{Prove}_1((\mathbf{s}, \mathbf{r}_1, \mathbf{r}_2); \mathbf{c})$
6. Send $(\mathbf{c}, \pi_1)$ to $\mathcal{I}$

Issuer $\mathcal{I}$.

7. **if** $[\![\mathsf{Verify}_1(\pi_1; \mathbf{c}) = 0]\!]$, **abort**
8. $\mathsf{t} \leftarrow \mathcal{F}(\mathsf{st})$. ▷ See Remark 4.2 for discussion on $\mathcal{F}$
9. $\mathsf{st} \leftarrow \mathsf{st} + 1$
10. $\mathbf{v}_3 \hookleftarrow \mathcal{D}_{\mathcal{R}_2^{k-\ell}, s_4}$
11. $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1' \\ \mathbf{v}_2 \end{bmatrix} \leftarrow \mathsf{RingTruncatedSampler}(\mathbf{R}_1, \mathbf{R}_2, \mathbf{A}, \mathbf{u} + \mathbf{c} - \mathbf{A}_3\mathbf{v}_3, \mathsf{t}, s_{\mathbf{G}}, s_1, s_2, s_3, s_4)$
12. Parse $\mathbf{v}_1'$ into $[{\mathbf{v}_{1,1}'}^T | {\mathbf{v}_{1,2}'}^T]^T$ with $\mathbf{v}_{1,1}', \mathbf{v}_{1,2}' \in \mathcal{R}_2^d$
13. **if** $[\![\|\mathbf{v}_{1,1}'\|_2 > B_{1,1}']\!] \vee [\![\|\mathbf{v}_{1,2}'\|_2 > B_{1,2}']\!] \vee [\![\|\mathbf{v}_2\|_2 > B_2]\!] \vee [\![\|\mathbf{v}_3\|_2 > B_3]\!]$ **goto** 10.
14. Send $(\mathsf{t}, \mathbf{v}_{1,2}', \mathbf{v}_2, \mathbf{v}_3)$ to $\mathcal{P}$

Platform $\mathcal{P}$.

15. $\mathbf{v}_{1,1}' \leftarrow \mathbf{u} + \mathbf{c} - \mathbf{A}\mathbf{v}_{1,2}' - (\mathsf{t}\mathbf{G}_H - \mathbf{B})\mathbf{v}_2 - \mathbf{A}_3\mathbf{v}_3 \bmod q\mathcal{R}_2$
16. **if** $[\![\|\mathbf{v}_{1,1}'\|_2 > B_{1,1}']\!] \vee [\![\|\mathbf{v}_{1,2}'\|_2 > B_{1,2}']\!] \vee [\![\|\mathbf{v}_2\|_2 > B_2]\!] \vee [\![\|\mathbf{v}_3\|_2 > B_3]\!] \vee \mathsf{t} \notin \mathcal{T}_w$, **abort**
17. $\mathbf{v}_{1,1} \leftarrow \mathbf{v}_{1,1}' - \mathbf{r}_1$
18. $\mathbf{v}_{1,2} \leftarrow \mathbf{v}_{1,2}' - \mathbf{r}_2$

Output: $\mathcal{I}$ gets $\perp$, and $\mathcal{P}$ gets $\mathsf{sk} = (s, \mathsf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3)$

Algorithm 4.4: KeyRevoke($\{\mathsf{sk}_i; i \in [N]\}$)

Input: List of platform keys $\{\mathsf{sk}_i; i \in [N]\}$.

1. **for** $i = 1, \ldots, N$ **do**
2. Parse sk_i as $(s_i, *)$
3. $\mathsf{KRL}[i] \leftarrow s_i$

Output: KRL

Algorithm 4.5: Sign(ipk, sk, M, SRL)

Input: Issuer public key ipk, platform secret key sk, message M, signature revocation list SRL.

$\mathsf{seed} \hookleftarrow U(\{0,1\}^{256})$
$(e', \mathbf{e}) \hookleftarrow U(\tau_1^{-1}([-\eta, \eta]^{n_1})^3)$
$e = \mathcal{H}_2(s, \mathsf{seed})$ ▷ $e \in \tau_1^{-1}([-\eta, \eta]^{n_1})$
$c \leftarrow \mathcal{H}_1(\mathsf{seed}) \cdot s + e \bmod p\mathcal{R}_1$
$\mathsf{tag} \leftarrow \mathcal{H}_3(\mathsf{seed}, c) \cdot s + \mathbf{e} \bmod p\mathcal{R}_1$ ▷ $\mathsf{tag} \in \mathcal{R}_{1,p}^2$
$(h, \mathbf{B}_F) \leftarrow \mathsf{Falcon.KeyGen}(\mathcal{R}_1, p)$
$t \leftarrow h \cdot s + e' \bmod p\mathcal{R}_1$
for $i = 1, \ldots, |\mathsf{SRL}|$ **do**
    $(\mathsf{seed}_i, c_i, \mathsf{tag}_i) \leftarrow \mathsf{SRL}[i]$
    if $[\![\|c_i - (\mathcal{H}_1(\mathsf{seed}_i)s + \mathcal{H}_2(s, \mathsf{seed}_i)) \bmod p\mathcal{R}_1\|_\infty \leq 2\eta]\!]$, **return** sig $=\perp$
    $\mathbf{a}_i \leftarrow \mathcal{H}_3(\mathsf{seed}_i, c_i)$ ▷ $\mathbf{a}_i \in \mathcal{R}_{1,p}^2$
    $\boldsymbol{\gamma}_i \leftarrow \mathcal{H}_4(\mathsf{seed}, c, \mathsf{seed}_i, c_i)$ ▷ $\boldsymbol{\gamma}_i \in \mathcal{R}_1^2$, s.t. $\|\boldsymbol{\gamma}_i\|_2 \leq \beta_F$
    do
        $(x_{i,1}, x_{i,2}) \leftarrow \mathsf{Falcon.SamplePre}(\mathbf{B}_F, h, \boldsymbol{\gamma}_i^T \mathbf{a}_i \bmod p\mathcal{R}_1)$
    while $[\![\|[x_{i,1}|x_{i,2}]\|_2 > \beta_F]\!]$ ▷ $x_{i,1} + hx_{i,2} = \boldsymbol{\gamma}_i^T \mathbf{a}_i \bmod p\mathcal{R}_1$
$\pi_2 \leftarrow \mathsf{Prove}_2((s, \mathbf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3, \mathbf{e}, e, e'); \mathsf{seed}, c, h, t, \mathsf{tag}, (x_{i,2})_i, M, \mathcal{R}_3)$ ▷ the $x_{i,2}$ are hashed alongside M

Output: sig $= (\mathsf{seed}, c, \mathsf{tag}, h, t, x_{1,2}, \ldots, x_{|\mathsf{SRL}|,2}, \pi_2)$

Algorithm 4.6: SigRevoke($\{\mathsf{sig}_i; i \in [N]\}$)

Input: List of signatures $\{\mathsf{sig}_i; i \in [N]\}$.

for $i = 1, \ldots, N$ **do**
    Parse sig_i as $(\mathsf{seed}_i, c_i, \mathsf{tag}_i, *)$
    $\mathsf{SRL}[i] \leftarrow (\mathsf{seed}_i, c_i, \mathsf{tag}_i)$

Output: SRL

Algorithm 4.7: Identify(sk, SRL)

Input: Platform secret key sk, signature revocation list SRL.

Parse sk as $(s, *)$
$\mu \leftarrow 0$
for $i = 1, \ldots, |\mathsf{SRL}|$ **do**
    $(\mathsf{seed}_i, c_i, \mathsf{tag}_i) \leftarrow \mathsf{SRL}[i]$
    $\mu \leftarrow \mu \vee [\![\|c_i - (\mathcal{H}_1(\mathsf{seed}_i)s + \mathcal{H}_2(s, \mathsf{seed}_i)) \bmod p\mathcal{R}_1\|_\infty \leq 2\eta]\!]$

Output: μ ($\mu = 1$ if sk revoked by SRL, 0 otherwise)

Algorithm 4.8: Verify(ipk, sig, M, SRL, KRL)

Input: Issuer public key ipk, signature sig, message M, signature revocation list SRL, key revocation list KRL.

if $[\![\mathsf{sig} = \perp]\!]$ **return** 0
$\mu \leftarrow [\![\mathsf{Verify}_2(\pi_2; \mathsf{seed}, c, h, t, (x_{i,2})_i, \mathsf{tag}, M) = 1]\!]$
for $i = 1, \ldots, |\mathsf{KRL}|$ **do**
    $\mu \leftarrow \mu \wedge [\![\|c - \mathcal{H}_1(\mathsf{seed}) \cdot \mathsf{KRL}[i] \bmod p\mathcal{R}_1\|_\infty > \eta]\!]$
for $i = 1, \ldots, |\mathsf{SRL}|$ **do**
    $(\mathsf{seed}_i, c_i, \mathsf{tag}_i) \leftarrow \mathsf{SRL}[i]$
    $\mathbf{a}_i \leftarrow \mathcal{H}_3(\mathsf{seed}_i, c_i)$

8. $\boldsymbol{\gamma}_i \leftarrow \mathcal{H}_4(\mathsf{seed}, c, \mathsf{seed}_i, c_i)$
9. $x_{i,1} \leftarrow \boldsymbol{\gamma}_i^T \mathbf{a}_i - h x_{i,2} \bmod p\mathcal{R}_1$
10. $\mu \leftarrow \mu \wedge [\![\| [x_{i,1} | x_{i,2}] \|_2 \leq \beta_F]\!]$
11. $\mu \leftarrow \mu \wedge [\![\left\| x_{i,2} t - \boldsymbol{\gamma}_i^T \mathsf{tag}_i \bmod p\mathcal{R}_1 \right\|_\infty > \beta]\!]$

Output: μ ($\mu = 1$ if sig valid and not revoked, 0 otherwise)

Remark 4.1. One may be surprised, in the Sign algorithm, by the deterministic generation of the error e (Step 3) used to build c (Step 4). This deterministic generation is necessary to correctly simulate our signatures in the security analysis, where we will replace c by an R-LWE sample. As seed may appear in several signing queries and as we have only one R-LWE sample $\mathcal{H}_1(\mathsf{seed}) \cdot s + e$ associated with it, we must ensure that seed will always yield the same c and thus need the same sample, which is the case with our deterministic generation.

As this process is designed to correctly simulate honest platforms, we stress that there is no need to prove that it has been carried out correctly using NIZK proofs. A careless platform proceeding differently would only jeopardize its own security. All we need to prove about e is that it is small, irrespective of the way it was generated.

Remark 4.2. Following [AGJ+24], the function $\mathcal{F}$ in Algorithm 4.3 has no specific requirement beyond mapping injectively the $\mathsf{st}_0 + i$ (for i in $\{0, ..., Q_{\mathsf{Join}} - 1\}$) in the designated tag space $\mathcal{T}_w$. In particular, it should not create collisions of tag so as to ensure only one certificate is emitted for a given tag. In practice (see [AGJ+24,JS25a,JS25b]), the function $\mathcal{F}$ is instantiated with SHAKE256 and the initial state st_0 is generated as a random 64-byte value.

4.3.1 Verification Bounds.

We note that the bounds $B'_{1,1}, B'_{1,2}, B_2, B_3$ are set using the Gaussian tail bound of Lemma 2.1 as $c_{n_2 d} s_{1,1} \sqrt{n_2 d}$, $c_{n_2 d} s_{1,2} \sqrt{n_2 d}$, $c_{n_2 d(k-\ell)} s_4 \sqrt{n_2 d(k-\ell)}$ and $c_{n_2(k-\ell)} s_4 \sqrt{n_2(k-\ell)}$. In particular, we choose the tailcut so that the bound is verified with overwhelming probability to avoid unnecessary rejection on the issuer's part. One could decrease this tailcut to have slightly smaller parameters. We also define $B_{1,i} = B'_{1,i} + \sqrt{n_2 d}$ which corresponds to the bound on $\mathbf{v}_{1,i} = \mathbf{v}'_{1,i} - \mathbf{r}_i$.

The bound β_F is set as $c_F \sigma_F \sqrt{2n_1}$ using the Gaussian tail bound as well but with a smaller tailcut c_F to slightly optimize the linear dependency in the EPID signature size. A trade-off could be to use a larger tailcut to avoid too many restarts. Finally, the bound β is set as $\beta = \sqrt{2}\beta_F \cdot \sqrt{(1+3\eta^2)n_1}$. Although one expects to have elements much smaller than β when revoked, we have to consider a worst-case bound to balance correctness and non-frameability. In particular, our condition that $p \geq K \cdot 16\alpha^2 r_F^2 n_1^2 (1+3\eta^2)$ is so that $\beta < p/2$ (to avoid wrap-around modulo p) with a sufficient margin to ensure a negligible probability of a non-revoked platform being revoked by mistake. In our concrete parameter selection, the constant K can be chosen to be roughly $2^{\lambda/n_1} \cdot c_F^2$ which is a small constant (the factor $2^{\lambda/n_1}$ stemming from Lemma 2.3to make the correctness error negligible).

4.3.2 Relations for Prove_1 and Prove_2. The EPID system requires to prove two types of relations in zero-knowledge. The first aims at proving the commitment sent by the platform in the Join protocol is well-formed, as in the anonymous credentials of [AGJ+24]. The proof system $(\mathsf{Prove}_1, \mathsf{Verify}_1)$ is then a non-interactive zero-knowledge argument proving knowledge of $(\mathbf{s}, \mathbf{r}_1, \mathbf{r}_2) \in \mathcal{R}_2^{k_{1,2}+2d}$ such that

$$\mathbf{r}_1 + \mathbf{A}\mathbf{r}_2 + \mathbf{D}\mathbf{s} = \mathbf{c} \bmod q\mathcal{R}_2, \quad \|\mathbf{s}\|_2 \leq \sqrt{n_1}, \quad \mathbf{r}_1, \mathbf{r}_2 \in T_1^d$$

where $T_1 = \tau_2^{-1}(\{0,1\}^{n_2})$. The relation statement $\mathsf{statement}_1$ then contains all the public elements of the above relation, those not in red. The second argument is used when generating EPID signatures so that a platform can prove that (1) it is a registered platform (i.e., a group member), and (2) the tags and non-revocation tokens are well-formed (to argue the proof of non-revocation). The proof system $(\mathsf{Prove}_2, \mathsf{Verify}_2)$ is then a non-interactive zero-knowledge argument proving knowledge of by proving knowledge of $(s, \mathbf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3, \mathbf{e}, e, e')$ such that

$$\mathbf{v}_{1,1} + \mathbf{A}\mathbf{v}_{1,2} + (\mathbf{t}\mathbf{G}_H - \mathbf{B})\mathbf{v}_2 + \mathbf{A}_3\mathbf{v}_3 = \mathbf{u} + \mathbf{D}\widetilde{\theta}_{1,2}(s) \bmod q\mathcal{R}_2 \tag{2}$$

$$\|\mathbf{v}_{1,1}\|_2 \leq B_{1,1}, \quad \|\mathbf{v}_{1,2}\|_2 \leq B_{1,2}, \quad \|\mathbf{v}_2\|_2 \leq B_2, \quad \|\mathbf{v}_3\|_2 \leq B_3, \quad \mathbf{t} \in \mathcal{T}_w \tag{3}$$

$$\mathcal{H}_1(\mathsf{seed})s + e = c \bmod p\mathcal{R}_1 \tag{4}$$

$$\mathcal{H}_3(\mathsf{seed}, c)s + \mathbf{e} = \mathsf{tag} \bmod p\mathcal{R}_1 \tag{5}$$

$$hs + e' = t \bmod p\mathcal{R}_1 \tag{6}$$

$$\|s\|_2 \leq \sqrt{n_1}, \quad \|e\|_2 \leq \eta\sqrt{n_1}, \quad \|\mathbf{e}\|_2 \leq \eta\sqrt{2n_1}, \quad \|e'\|_2 \leq \eta\sqrt{n_1} \tag{7}$$

The statement $\mathsf{statement}_2$ then contains all the public elements of the above relation, those not in red. Both relations can be efficiently proven using [LNP22,LN22] as explained in Appendix A.

5 Security Analysis

We now assess the correctness, non-frameability, anonymity and unforgeability of our EPID system. The corresponding formal security statements are given in Theorems 5.1, 5.2, 5.3 and 5.4 respectively.

Theorem 5.1 (Correctness). *The EPID system described in Section 4 is correct, with correctness error*

$$N_{\mathsf{KRL}}\frac{(2\eta+1)^{n_1}}{p^{n_1}} + N_{\mathsf{SRL}}\frac{(2\beta+1)^{n_1}}{p^{n_1}},$$

with N_{KRL} and N_{SRL} the maximal number of revoked platform keys and EPID signaturesrespectively.

Proof. Let pp be generated from $\mathsf{Setup}(1^\lambda)$ and $(\mathsf{isk}, \mathsf{ipk})$ from $\mathsf{GKeyGen}(\mathsf{pp})$. Now let $(\perp, \mathsf{sk})$ be obtained from a successful execution of $\mathsf{Join}\langle\mathcal{I}(\mathsf{isk}, \mathsf{ipk}), \mathcal{P}(\mathsf{ipk})\rangle$,

and $\mathsf{SRL}, \mathsf{KRL}$ generated using $\mathsf{KeyRevoke}, \mathsf{SigRevoke}$ respectively. We let $\mathsf{sig} = \mathsf{Sign}(\mathsf{ipk}, \mathsf{sk}, M, \mathsf{SRL})$ for an arbitrary message M.
First, assume that $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 1$ or $\mathsf{KeyRevoke}(\{\mathsf{sk}\}) \in \mathsf{KRL}$.

- Assume $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 1$. It means there exists $j \in [|\mathsf{SRL}|]$ such that $\|c_j - (\mathcal{H}_1(\mathsf{seed}_j)s + \mathcal{H}_2(s, \mathsf{seed}_j)) \bmod p\mathcal{R}_1\|_\infty \leq 2\eta$. As the signature was generated honestly, it means that step 10 for the j-th loop iteration failed, yielding $\mathsf{sig} = \perp$. As a result, step 1 of Verify fails.
- Now assume that $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 0$. It means that $\mathsf{KeyRevoke}(\{\mathsf{sk}\}) \in \mathsf{KRL}$. It also ensures that $\mathsf{sig} \neq \perp$ and we can thus parse the signature as $\mathsf{sig} = (\mathsf{seed}, c, \mathsf{tag}, h, t, x_{1,2}, \ldots, x_{N,2}, \pi_2)$ where $N = |\mathsf{SRL}|$. Because we have $\mathsf{KeyRevoke}(\{\mathsf{sk}\}) \in \mathsf{KRL}$, there exists $j \in [|\mathsf{KRL}|]$ such that $\mathsf{KRL}[j] = s$. Hence, we have $c - \mathcal{H}_1(\mathsf{seed})\mathsf{KRL}[j] = \mathcal{H}_1(\mathsf{seed})(s - \mathsf{KRL}[j]) + e = e \bmod p\mathcal{R}_1$ whose infinity norm is bounded by η. So step 4 of Verify fails and gives $\mu = 0$.

Either way, we indeed have $\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}, M, \mathsf{SRL}, \mathsf{KRL}) = 0$.
Now assume that $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 0$ and that $\mathsf{KeyRevoke}(\{\mathsf{sk}\}) \notin \mathsf{KRL}$.

- Because $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 0$, it means that for all i in $[|\mathsf{SRL}|]$, we have $\|c_i - (\mathcal{H}_1(\mathsf{seed}_i)s + \mathcal{H}_2(s, \mathsf{seed}_i)) \bmod p\mathcal{R}_1\|_\infty > 2\eta$. By definition of Sign, it entails that $\mathsf{sig} \neq \perp$. As a result, step 1 of Verify passes.
- We can then parse the signature as $\mathsf{sig} = (\mathsf{seed}, c, \mathsf{tag}, h, t, x_{1,2}, \ldots, x_{N,2}, \pi_2)$. We must then check that π_2 verifies. As π_2 was generated honestly, we only need to check that the witness was valid to begin with, i.e., that it verifies Equations (2) to (7). By definition of a successful execution of Join, it holds that (2) and (3) hold. By construction, (4), (5) and (6) trivially hold as well. Finally, because of how $s, e, \mathbf{e}$ and e' are sampled, the bounds of (7) also hold. It thus ensures that π_2 is a valid proof based on the completeness of $(\mathsf{Prove}_2, \mathsf{Verify}_2)$. It means step 2 of Verify passes.
- Now let i be in $[|\mathsf{KRL}|]$. We have

$$c - \mathcal{H}_1(\mathsf{seed})\mathsf{KRL}[i] = \mathcal{H}_1(\mathsf{seed})(s - \mathsf{KRL}[i]) + e \bmod p\mathcal{R}_1.$$

  Because $\mathsf{KeyRevoke}(\{\mathsf{sk}\}) \notin \mathsf{KRL}$, it means that $s \neq \mathsf{KRL}[i]$. Our choice of parameters then yields $0 < \|s - \mathsf{KRL}[i]\|_\infty \leq 2 < p^{1/\kappa}/\sqrt{\kappa}$. Using [LS18, Cor. 1.2] ensures that $s - \mathsf{KRL}[i]$ is a unit of $\mathcal{R}_{1,p}$. As a result, we have that $\mathcal{H}_1(\mathsf{seed})(s - \mathsf{KRL}[i])$ is uniform in $\mathcal{R}_{1,p}$. Because it is independent of e (e is derived from a random oracle), it gives that $c - \mathcal{H}_1(\mathsf{seed})\mathsf{KRL}[i] \bmod p\mathcal{R}_1$ is also uniform in $\mathcal{R}_{1,p}$. By Lemma 2.3 and the union bound, it holds that step 4 gives $\mu = 0$ with probability at most $|\mathsf{KRL}|((2\eta+1)/p)^{n_1}$ which is negligible as $\eta \ll p$.
- Finally, we have that each tag_i from SRL is of the form $\mathbf{a}_i s_i + \mathbf{e}_i \bmod p\mathcal{R}_1$ for $\mathbf{a}_i = \mathcal{H}_3(\mathsf{seed}_i, c_i)$. Hence, we have

$$x_{i,2}t - \boldsymbol{\gamma}_i^T \mathsf{tag}_i = \boldsymbol{\gamma}_i^T \mathbf{a}_i(s - s_i) + (x_{i,2}e' - x_{i,1}s - \boldsymbol{\gamma}_i^T \mathbf{e}_i) \bmod p\mathcal{R}_1.$$

  We must now argue that $s \neq s_i$ for all $i \in [|\mathsf{SRL}|]$. Assume towards contradiction there exists j such that $s_j = s$. Because $\mathsf{Identify}(\mathsf{sk}, \mathsf{SRL}) = 0$, it

means in particular that $\|c_j - (\mathcal{H}_1(\mathsf{seed}_j)s + \mathcal{H}_2(s, \mathsf{seed}_j)) \bmod p\mathcal{R}_1\|_\infty > \eta$, and thence $\|c_j - (\mathcal{H}_1(\mathsf{seed}_j)s_j + \mathcal{H}_2(s_j, \mathsf{seed}_j)) \bmod p\mathcal{R}_1\|_\infty > \eta$. Yet, $c_j - (\mathcal{H}_1(\mathsf{seed}_j)s_j + \mathcal{H}_2(s_j, \mathsf{seed}_j)) = 0 \bmod p\mathcal{R}_1$ which then gives a contradiction. We then have that s differs from all the s_i. Using the same argument with Lemma 2.3 and the union bound then gives that step 11 gives $\mu = 0$ with probability at most $|\mathsf{SRL}|((2\beta+1)/p)^{n_1}$.

We can then conclude that $\mathbb{P}[\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}, M, \mathsf{SRL}, \mathsf{KRL}) = 0] \leq |\mathsf{KRL}|((2\eta+1)/p)^{n_1} + |\mathsf{SRL}|((2\beta+1)/p)^{n_1}$ as claimed. □

Note that the correctness error corresponds to the probability that an honest unrevoked user does not pass the revocation test. We select parameters for the latter to be negligible so that the user does not have to re-sign in practice. We however insist that if a user is revoked, the revocation test will *always* detect it and output 0 at verification, indeed denying access to said user. In other words, our system is designed so as to produce no false negatives, and false positives only with negligible probability.

Theorem 5.2 (Non-Frameability). *The EPID system described in Section 4 is ε_{nfra}-non-frameable, with*

$$\begin{aligned}\varepsilon_{nfra} \approx 2 \cdot \max\big(&\varepsilon_{\text{R-SIS}} + \varepsilon_{snd,2}\ ,\quad Q_{\mathsf{Join}}\big(\varepsilon_{zk,1} + Q_{\mathsf{Sign}}\varepsilon_{zk,2} + \varepsilon_{\text{M-LWE}} \\ &+ \kappa Q_{\mathcal{H}_3} p^{-n_1/\kappa} + (Q_{\mathcal{H}_3} + Q_{\mathsf{Sign}})\varepsilon_{\text{NTRU}} + 3N_{\mathsf{SRL}}Q_{\mathsf{Sign}}\varepsilon_F \\ &+\varepsilon_{snd,2} + \varepsilon_{\text{R-LWE}})\big)\,,\end{aligned}$$

where $Q_{\mathsf{Join}}, Q_{\mathsf{Sign}}, Q_{\mathcal{H}_1}, Q_{\mathcal{H}_2}, Q_{\mathcal{H}_3}$ are the maximal number of queries to Join *(per issuer key pair),* Sign *(per platform), $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{H}_3$ respectively. The quantities $\varepsilon_{\text{R-SIS}}, \varepsilon_{\text{M-LWE}}, \varepsilon_{\text{R-LWE}}, \varepsilon_{\text{NTRU}}$ then are the hardness bounds of $\text{R-SIS}_{n_1,Q_{\mathcal{H}_1}+1,p,B}$ (with $B = 2\sqrt{4\eta^2 n_1 + n_1}$), $\text{M-LWE}_{n_2,d,d,q,U(T_1),U(T_1)}$ (with $T_1 = \tau_2^{-1}(\{0,1\}^{n_2})$), $\text{R-LWE}_{n_1,4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1},p,U(S_1),U(S_\eta)}$ (where $S_\alpha = \tau_1^{-1}([-\alpha,\alpha]^{n_1})$), and along with $\text{NTRU}_{n_2,p,\mathcal{D}_{\mathcal{R}_1,\sigma_{f,g}}}$ respectively. Finally, $\varepsilon_{zk,i}$ denotes the zero-knowledge loss of* Prove_i*, while $\varepsilon_{snd,2}$ is the soundness error of* Prove_2.

Proof. We tackle two types of attacks differently. The first one is when the adversary tries to frame an honest platform without its secret key $s_{i^\star}$, while the second is when they essentially managed to recover $s_{i^\star}$. At a high level, the first type means that the attacker either managed to break the soundness of the proof system or produced some $s \neq s_{i^\star}$ such that $as + e = c = as_{i^\star} + e'$. As the token c can be seen as a binding commitment of $s_{i^\star}$, it means they broke the binding property, which reduces to solving R-SIS. For the second type, we essentially want to argue that the secret $s_{i^\star}$ of the targeted honest platform cannot be recovered from the information available to the adversary. We go through a series of hybrids to eventually rely on the hardness of *search* R-LWE.

Type ❶. As we explain above, the first type (❶) is the one where $\mathcal{A}$ wins the non-frameability game with $M, \mathsf{sig}, \mathsf{SRL}$ such that either $\mathcal{A}$ spoofed the proof π_2, or proved knowledge of a secret s that is different from that ($s_{i^\star}$) of the

framed honest platform $i^\star$. In that case, we show that we can use $\mathcal{A}$ to break the soundness of the proof system, or to produce a solution to R-SIS. Let us now show it formally.

The challenger receives a R-SIS$_{n_1,Q_{\mathcal{H}_1}+1,p,B}$ instance $[1|\mathbf{a}^T]$ where we have $\mathbf{a} = [a_i]_{i\in[Q_{\mathcal{H}_1}]} \in \mathcal{R}_{1,p}^{Q_{\mathcal{H}_1}}$. The challenger keeps a table $\mathsf{T}_{\mathcal{H}_1}$ initially empty (i.e., all entries equal to $\perp$) and a counter ctr initially set at 1. It then simulates the game perfectly except that it reprograms the queries to $\mathcal{H}_1$. On input a seed seed, the challenger checks if $\mathsf{T}_{\mathcal{H}_1}[\mathsf{seed}] = \perp$. If so, it outputs a_{ctr}, updates $\mathsf{T}_{\mathcal{H}_1}[\mathsf{seed}] \leftarrow a_{\mathsf{ctr}}$, and increments ctr by 1. If on the other hand $\mathsf{T}_{\mathcal{H}_1}[\mathsf{seed}] \neq \perp$, it simply returns $\mathsf{T}_{\mathcal{H}_1}[\mathsf{seed}]$. Because the a_i are all uniform, this does not change the view of the adversary in the random oracle model. Then, $\mathcal{A}$ eventually produces $M, \mathsf{sig}, \mathsf{SRL}$ that wins the non-frameability game in the way defined for a type ❶ attack.

We parse sig as $(\mathsf{seed}, c, \mathsf{tag}, h, t, x_{1,2}, \ldots, x_{|\mathsf{SRL}|,2}, \pi_2)$. Because sig verifies, the proof π_2 in sig must be valid. Then, except with the soundness error $\varepsilon_{snd,2}$ of the NIZK $(\mathsf{Prove}_2, \mathsf{Verify}_2)$, the challenger can extract (s, e) such that $c = \mathcal{H}_1(\mathsf{seed})s + e \bmod p\mathcal{R}_1$ with $\|s\|_2 \leq \sqrt{n_1}$ and $\|e\|_2 \leq \eta\sqrt{n_1}$. By definition of a valid type ❶ attack, it holds that $s \neq s_{i^\star}$. Nevertheless, the winning conditions also require that $\mathsf{Identify}(\mathsf{sk}_{i^\star}, \mathsf{SigRevoke}(\{\mathsf{sig}\})) = 1$. This means that $\|c - (\mathcal{H}_1(\mathsf{seed})s_{i^\star} + \mathcal{H}_2(s_{i^\star}, \mathsf{seed})) \bmod p\mathcal{R}_2\|_\infty \leq 2\eta$. We can then define $e' = \mathcal{H}_2(s_{i^\star}, \mathsf{seed})$ and $e'' = c - (\mathcal{H}_1(\mathsf{seed})s_{i^\star} + e') \bmod p\mathcal{R}_1$. It then holds that $\|e''\|_2 \leq 2\eta\sqrt{n_1}$. Also, by definition of $\mathcal{H}_2$, $\|e'\|_2 \leq \eta\sqrt{n_1}$. We then have that $c = \mathcal{H}_1(\mathsf{seed})s_{i^\star} + e' + e'' \bmod p\mathcal{R}_1$. Combining this equation with the extraction above, we obtain

$$\mathcal{H}_1(\mathsf{seed})(s - s_{i^\star}) + e - e' - e'' = 0 \bmod p\mathcal{R}_1.$$

Because of how we reprogrammed $\mathcal{H}_1$, there must exist $j \in [Q_{\mathcal{H}_1}]$ such that $\mathcal{H}_1(\mathsf{seed}) = a_j$ (if seed was never queried, the attacker would not know $\mathcal{H}_1(\mathsf{seed})$ and would have had no chance of producing a valid proof π_2). As a result, if we define $\mathbf{x} = [e - e' - e''|0 \ldots 0|s - s_{i^\star}|0 \ldots 0]^T$ where $s - s_{i^\star}$ is at the $(j+1)$-th position, we have $[1|\mathbf{a}^T]\mathbf{x} = 0 \bmod p\mathcal{R}_1$. Additionally, $\mathbf{x} \neq \mathbf{0}$ because $s \neq s_{i^\star}$, and $\|\mathbf{x}\|_2 \leq 2\sqrt{4\eta^2 n_1 + n_1} = B$. It means that $\mathbf{x}$ is a valid solution to R-SIS$_{n_1,Q_{\mathcal{H}_1}+1,p,B}$.

The advantage of $\mathcal{A}$ in successfully carrying a type ❶ attack is then bounded by $\varepsilon_{snd,2} + \varepsilon_{\text{R-SIS}}$ where $\varepsilon_{\text{R-SIS}}$ is the hardness bound of R-SIS$_{n_1,Q_{\mathcal{H}_1}+1,p,B}$.

Type ❷. The second type (❷) occurs when the conditions of first type are not met, that is, the final signature includes a proof of the secret of the framed honest platform $i^\star$. For this second type, the proof is a bit more involved. At a high level, we introduce several hybrid changes so as to simulate the EPID signatures of the framed honest platform $i^\star$ without resorting to its secret key sk. That way, we can insert a *search* R-LWE challenge where the secret would correspond to the secret of the honest platform. From the signature that frames platform $i^\star$, we would then argue that either the soundness of the NIZK has been broken, or that we can extract the platform secret and solve R-LWE. Most of the steps needed to simulate without knowing the platform secret are very

classical as they consist in guessing the targeted platform, simulating the zero-knowledge proofs, replacing hiding commitments by random values, etc. The main challenges lie in the simulation of the elements tag, c and t of the signature. For tag and c, the situation is rather simple as they already have the form of R-LWE samples. By properly programming the random oracle, one can readily insert R-LWE challenges. The simulation is more complex for t as it is supposed to be generated as $h \cdot s + e'$, where h is a Falcon trapdoor. Replacing t by a R-LWE challenge is not difficult in itself but then one loses the trapdoor and so the ability to generate the preimages $(x_{i,1}, x_{i,2})$. Our solution is to move (by carefully reprograming the ROM) the trapdoors in the vectors $\mathbf{a}_i$ used by all the users but the targeted one and then leverage the randomness brought by $\boldsymbol{\gamma}_i$ to get the expected relations $\boldsymbol{\gamma}_i^T \mathbf{a}_i = a_i = x_{i,1} + h x_{i,2}$. One then ends up with tag, c and t that are R-LWE challenges while still being to simulate the proof of non-revocation.

We now describe these different changes formally, while tracking the loss incurred by each change. We define $\mathtt{Game}_0 = \mathtt{Game}_{\mathcal{A}}^{nfra}$ to be the original non-frameability game.

$\mathtt{Game}_1$ (identity guess): The challenger proceeds as in $\mathtt{Game}_0$ except that it samples $i^+ \hookleftarrow U([Q_{\mathsf{Join}}])$ at the outset of the game. This represents a guess on the honest platform $i^\star$ that will eventually be framed by the signature of the adversary. Then, it proceeds as usual except that at step 9 of the game in Figure 3.3, the challenger returns 1 if $i^\star = i^+$ in addition to $i^\star$ being revoked by sig and sig not obtained by an $\mathcal{O}\mathsf{Sign}(i^\star, *, *)$ query. If the framed platform $i^\star$ is not i^+, the challenger aborts. Because i^+ is uniform and independent of the adversary's view, conditioning on this event happening yields $\mathbb{P}[\mathtt{Game}_1(1^\lambda) = 1] = \mathbb{P}[i^+ = i^\star]\mathbb{P}[\mathtt{Game}_1(1^\lambda) = 1 | i^+ = i^\star] + 0 = \mathbb{P}[\mathtt{Game}_0(1^\lambda) = 1]/Q_{\mathsf{Join}}$. By definition of the advantage, it then directly gives

$$\mathrm{Adv}_{\mathtt{Game}_0}[\mathcal{A}] = Q_{\mathsf{Join}} \cdot \mathrm{Adv}_{\mathtt{Game}_1}[\mathcal{A}].$$

We note that we can now assume that there will be no query $\mathcal{O}\mathsf{Cor}(i^+)$, otherwise $\mathcal{A}$ will have corrupted i^+ and would thus lose the game (as $i^+ = i^\star$ and $i^\star$ honest would be incompatible).

$\mathtt{Game}_2$ (simulating join proof for i^+): The game is the same as the previous one except for how the query $\mathcal{O}\mathsf{Add}(i^+)$ is handled. In this query, the challenger acting as the registering platform i^+ simulates the proof π_1. Concretely, instead of generating the proof legitimately using the witness $(\mathbf{s}, \mathbf{r}_1, \mathbf{r}_2)$, it uses the simulator $\mathcal{S}_1$ to produce a valid proof π_1 without resorting to the witness. By the zero-knowledge property of Prove_1, it holds that

$$\mathrm{Adv}_{\mathtt{Game}_1}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_2}[\mathcal{A}] + \varepsilon_{zk,1},$$

where $\varepsilon_{zk,1}$ is the zero-knowledge loss of Prove_1, that is the maximal advantage of an adversary in distinguishing the real proof from the simulated one.

$\mathtt{Game}_3$ (simulating sign proofs for i^+): We now introduce a change in how the queries to platform i^+ are handled. Concretely, when a query $\mathcal{O}\mathsf{Sign}(i^+, \mathsf{SRL}, M)$

happens, it proceeds normally in the signature algorithm except that it simulates the proof π_2, i.e., the challenger only changes step 14 of Sign. At this step, it uses the simulator $\mathcal{S}_2$ to produce a valid proof π_2 without resorting to the witness $(s, \mathbf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3, \mathbf{e}, e')$. By the zero-knowledge property of Prove_2, it holds that

$$\mathrm{Adv}_{\mathtt{Game}_2}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_3}[\mathcal{A}] + Q_{\mathsf{Sign}} \cdot \varepsilon_{zk,2},$$

where $\varepsilon_{zk,2}$ is the zero-knowledge loss of Prove_2, that is the maximal advantage of an adversary in distinguishing the real proof from the simulated one. As we have at most Q_{Sign} queries per platform key, there are at most Q_{Sign} proofs to simulate.

$\mathtt{Game}_4$ (simulating commitment for i^+): We now simulate the commitment $\mathbf{c}$ in the query $\mathcal{O}\mathsf{Add}(i^+)$. It is possible because the commitment randomness $\mathbf{r}_1, \mathbf{r}_2$ is no longer required to produce the simulated proofs π_2 emitted by platform i^+. To produce $\mathbf{c}$ in the query $\mathcal{O}\mathsf{Add}(i^+)$, the challenger simply samples $\mathbf{c}' \hookleftarrow U(\mathcal{R}_{2,q}^d)$ and sets $\mathbf{c} = \mathbf{c}' + \mathbf{Ds} \bmod q\mathcal{R}_2$. This change is indistinguishable under the M-LWE$_{n_2,d,d,q,U(T_1),U(T_1)}$ where $T_1 = \tau_2^{-1}(\{0,1\}^{n_2})$. The reduction would go as follows. Given an instance $(\mathbf{A}, \mathbf{c}')$, the challenger would use the instance $\mathbf{c}'$ to simulate $\mathtt{Game}_4$. If $\mathbf{c}' = \mathbf{r}_1 + \mathbf{A}\mathbf{r}_2$ for some $\mathbf{r}_i \sim U(T_1^d)$, then it perfectly simulate $\mathtt{Game}_3$. If $\mathbf{c}'$ is uniform, it then perfectly simulate $\mathtt{Game}_4$. We actually go one step further in the simulation of the commitment by directly sampling $\mathbf{c}$ uniformly without changing the distribution from the previous modification. As $\mathbf{c}'$ is uniform and independent of $\mathbf{Ds}$, it acts as a perfect mask. Hence $\mathbf{c} = \mathbf{c}' + \mathbf{Ds}$ follows the uniform distribution over $\mathcal{R}_{2,q}^d$. With these two changes, it holds that

$$\mathrm{Adv}_{\mathtt{Game}_3}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_4}[\mathcal{A}] + \varepsilon_{\text{M-LWE}},$$

where $\varepsilon_{\text{M-LWE}}$ is the hardness bound of M-LWE$_{n_2,d,d,q,U(T_1),U(T_1)}$.

$\mathtt{Game}_5$ (ensuring unit for other platforms): We now leverage the random oracle $\mathcal{H}_3$ to program (some) of its queries. We program it for every input (seed, c), except those generated by the challenger during a query $\mathcal{O}\mathsf{Sign}(i^+, *, *)$. Recall that in its legitimate use, the domain of $\mathcal{H}_3$ is $\mathcal{R}_{1,p}^2$. We now change it so that the first entry of $\mathcal{H}_3(\mathsf{seed}, c)$ for some input (seed, c) is now a unit in $\mathcal{R}_{1,p}$. In the random oracle model, it means that for each new input, we sample $u' \hookleftarrow U(\mathcal{R}_{1,p}^\times)$ and $u'' \hookleftarrow U(\mathcal{R}_{1,p})$ and set the output of $\mathcal{H}_3$ to be $[u'|u'']^T$. For clarity with the next game, we write u'' as $u'h'$ for some uniform h'. It is always possible as u' is a unit, and thus preserves the uniform distribution.

We now need to argue that this is indistinguishable for the adversary. For that, we simply need to bound the probability that at least one of $Q_{\mathcal{H}_3}$ perfectly uniform elements $u'_1, \ldots, u'_{Q_{\mathcal{H}_3}}$ is not a unit. Because they are independent and uniformly distributed, this probability is equal to $1 - \mathbb{P}_{u' \sim U(\mathcal{R}_{1,p})}[u' \in \mathcal{R}_{1,p}^\times]^{Q_{\mathcal{H}_3}}$. Because $\mathcal{R}_1$ is a cyclotomic field and that p is an unramified prime that splits into κ distinct prime ideals, Lemma 2.4 gives that the probability equals $1 - (1 - 1/p^{n_1/\kappa})^{\kappa Q_{\mathcal{H}_3}} \leq \kappa Q_{\mathcal{H}_3} p^{-n_1/\kappa}$. In particular, this probability

equals $\Delta(U(\mathcal{R}_{1,p}^{Q_{\mathcal{H}_3}}), U((\mathcal{R}_{1,p}^{\times})^{Q_{\mathcal{H}_3}}))$, which then yields

$$\mathrm{Adv}_{\mathtt{Game}_4}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_5}[\mathcal{A}] + 1 - \left(1 - \frac{1}{p^{n_1/\kappa}}\right)^{\kappa Q_{\mathcal{H}_3}} \leq \mathrm{Adv}_{\mathtt{Game}_5}[\mathcal{A}] + \frac{\kappa Q_{\mathcal{H}_3}}{p^{n_1/\kappa}},$$

For our parameters, this quantity is negligible, even for extremely large values of $Q_{\mathcal{H}_3}$ such as 2^λ.

$\mathtt{Game}_6$ (hiding trapdoor for other platforms): We again program $\mathcal{H}_3$ on all inputs (seed, c), except those generated by the challenger during a query $\mathcal{O}\mathsf{Sign}(i^+, *, *)$. Concretely, we make the following two changes. When the adversary queries $\mathcal{H}_3$ with input (seed_i, c_i), the challenger checks if it was already queried. If so it returns the stored $\mathbf{a}_i$. If not, it samples $u_i' \hookleftarrow U(\mathcal{R}_{1,p}^{\times})$ and $(h_i', \mathbf{B}_{F,i}') \leftarrow \mathsf{Falcon.KeyGen}(\mathcal{R}_1, p)$, and defines $\mathbf{a}_i = \mathcal{H}_3(\mathsf{seed}_i, c_i) = [u_i'|u_i'h_i']^T$. Similarly, for queries to $\mathcal{O}\mathsf{Sign}(i, *, *)$ with $i \neq i^+$, the challenger proceeds the same way we just described by sampling a uniform unit u' and embedding a Falcon public key in h' to define $\mathcal{H}(\mathsf{seed}, c) = [u'|u'h']^T$. The only difference comes from the fact that the second component is no longer perfectly uniform as it embeds a NTRU trapdoor. By the NTRU assumption, it holds that

$$\mathrm{Adv}_{\mathtt{Game}_5}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_6}[\mathcal{A}] + Q_{\mathcal{H}_3} \cdot \varepsilon_{\mathrm{NTRU}},$$

where $\varepsilon_{\mathrm{NTRU}}$ is the hardness bound of $\mathrm{NTRU}_{n_2,p,\mathcal{D}_{\mathcal{R}_1,\sigma_{f,g}}}$. We explain in the parameter selection why $\varepsilon_{\mathrm{NTRU}}$ largely compensates $Q_{\mathcal{H}_3}$ even for extreme values like 2^λ. At a high-level, this is due to the fact that the Falcon parameters we choose need to satisfy the constraints from the proof of non-revocation which are much more restrictive than in the standard signature use-case. The NTRU hardness then amortizes this $Q_{\mathcal{H}_3}$ loss factor.

$\mathtt{Game}_7$ (hiding preimages for i^+): We now leverage the random oracle $\mathcal{H}_4$ to program its queries made by the challenger when answering an $\mathcal{O}\mathsf{Sign}(i^+, \mathsf{SRL}, M)$ query. Concretely, instead of generating $\boldsymbol{\gamma}_j = \mathcal{H}_4(\mathsf{seed}, c, \mathsf{seed}_j, c_j)$ following a centered Gaussian distribution of parameter σ_F and then defining $a_j = \boldsymbol{\gamma}_j^T \mathbf{a}_j$, we sample $a_j \hookleftarrow U(\mathcal{R}_{1,p})$ and use the trapdoor h_j' hidden in $\mathbf{a}_j$ to sample $\boldsymbol{\gamma}_j$. Before explaining it formally, we need to take care of a small subtlety. There may exist some j such that $\mathsf{SRL}[j] = (\mathsf{seed}_j, c_j, \mathsf{tag}_j)$ with (seed_j, c_j) that was previously emitted by platform i^+, in which case there would be no hidden trapdoor in $\mathbf{a}_j = \mathcal{H}_3(\mathsf{seed}_j, c_j)$. However, this would be detected at step 10 of the signing procedure, leading the platform to abort. Indeed, if (seed_j, c_j) was reused from a previous signature emitted by i^+, it means that it is well-formed and that $c_j - (\mathcal{H}_1(\mathsf{seed}_j)s + \mathcal{H}_2(s, \mathsf{seed}_j)) = 0 \bmod p\mathcal{R}_1$, which is a fortiori bounded by 2η.

If on the other hand (seed_j, c_j) does not match any of the signatures emitted by i^+, it means that $\mathbf{a}_j = \mathcal{H}_3(\mathsf{seed}_j, c_j)$ has a hidden trapdoor, i.e., is of the form $\mathbf{a}_j = u_j'[1|h_j']^T$ where h_j' is a Falcon public key. The challenger then reprograms the $\boldsymbol{\gamma}_j = \mathcal{H}_4(\mathsf{seed}, c, \mathsf{seed}_j, c_j)$ as follows. It samples $u_j \hookleftarrow U(\mathcal{R}_{1,p})$, computes ${u_j'}^{-1}u_j \bmod p\mathcal{R}_1$ and finally samples $\boldsymbol{\gamma}_j = \mathsf{Falcon.SamplePre}(\mathbf{B}_{F,j}', h_j', {u_j'}^{-1}u_j)$.

It then reprograms $\mathcal{H}_4(\mathsf{seed}, c, \mathsf{seed}_j, c_j) = \boldsymbol{\gamma}_j$. Based on the simulatability of Falcon preimages recalled in Lemma 2.6, it holds that

$$\mathrm{Adv}_{\mathtt{Game}_6}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_7}[\mathcal{A}] + N_{\mathsf{SRL}} \cdot Q_{\mathsf{Sign}} \cdot \frac{1}{2}\left(\left(\frac{1+\varepsilon_F/2n_1}{1-\varepsilon_F/2n_1}\right)^{2n_1} - 1\right).$$

This loss is roughly equal to $N_{\mathsf{SRL}} Q_{\mathsf{Sign}} \varepsilon_F$. Choosing $\varepsilon_F = 2^{-\lambda}/(N_{\mathsf{SRL}} Q_{\mathsf{Sign}})$ makes it negligible. We note here that we could use a Rényi divergence bound instead of the statistical distance given in Lemma 2.6. Combined with the multiplicativity of the Rényi divergence and the relative error lemma from [Pre17], it would yield a tighter loss, then allowing us to choose a larger ε_F (resulting in smaller parameters overall). However, we will need a similar argument in the anonymity proof but in the statistical distance this time. It will then require the statistical distance (which is roughly $N_{\mathsf{SRL}} Q_{\mathsf{Sign}} \varepsilon_F$) to be negligible. To avoid using different arguments, we proceed the same way here.

$\mathtt{Game}_8$ (simulate $x_{i,2}$ for i^+): We now simulate the $x_{i,2}$ component of a signature produced on behalf of the platform $\mathcal{P}^+$. Instead of sampling a_i $(= \boldsymbol{\gamma}_i^T \mathbf{a}_i)$ uniformly and then using the Falcon key hidden in h to find sample preimages $(x_{i,1}, x_{i,2})$, we sample $(x_{i,1}, x_{i,2})$ from $\mathcal{D}_{\mathcal{R}_1^2, \sigma_F}$ such that $\|[x_{i,1}|x_{i,2}]\|_2 \leq \beta_F$ and compute $a_i = x_{i,1} + h x_{i,2}$. This way, we do not use the Falcon secret key $\mathbf{B}_F$. Because $\sigma_F \geq \eta_{\varepsilon_F}(\mathcal{L}_q^\perp([1|h]))$, combining Lemma 2.6 again with Lemma 2.7 for the distributions of the u_i gives that

$$\mathrm{Adv}_{\mathtt{Game}_7}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_8}[\mathcal{A}] + N_{\mathsf{SRL}} \cdot Q_{\mathsf{Sign}} \cdot \frac{1}{2}\left(\left(\frac{1+\varepsilon_F/2n_1}{1-\varepsilon_F/2n_1}\right)^{2n_1} - 1 + \frac{2\varepsilon_F}{1+\varepsilon_F}\right),$$

The loss term is roughly equal to $2N_{\mathsf{SRL}} Q_{\mathsf{Sign}} \varepsilon_F$.

$\mathtt{Game}_9$ (simulate h for i^+): Now that the secret key hidden in the h is no longer used when querying $\mathcal{O}\mathsf{Sign}(i^+, *, *)$, we can replace them by perfectly uniform elements. Under the NTRU assumption, we have

$$\mathrm{Adv}_{\mathtt{Game}_8}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_9}[\mathcal{A}] + Q_{\mathsf{Sign}} \cdot \varepsilon_{\mathrm{NTRU}},$$

where $\varepsilon_{\mathrm{NTRU}}$ is the hardness bound for $\mathrm{NTRU}_{n_2, p, \mathcal{D}_{\mathcal{R}_1, \sigma_{f,g}}}$.

Bounding advantage in $\mathtt{Game}_9$: We finally argue that if an adversary is able to win the non-frameability in $\mathtt{Game}_9$, then we can either break the soundness of the proof system or solve the R-LWE instance we embedded into the behavior of platform i^+. For that, we simulate all the tag, c and t emitted by platform i^+ when answering queries to $\mathcal{O}\mathsf{Sign}(i^+, *, *)$, but without knowledge of its secret s. Concretely, the challenger receives a *search* R-LWE instance $(\mathbf{a}, \mathbf{t} = \mathbf{a}s + \mathbf{e} \bmod p\mathcal{R}_1) \in \mathcal{R}_{1,p}^{4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1}} \times \mathcal{R}_{1,p}^{4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1}}$. We parse $\mathbf{a}$ as $\mathbf{a} = [a'_1|\mathbf{a}_1^T|h_1|\ldots|a'_{Q_{\mathsf{Sign}}}|\mathbf{a}_{Q_{\mathsf{Sign}}}^T|h_{Q_{\mathsf{Sign}}}|u_1|\ldots|u_{Q_{\mathcal{H}_1}}]^T$. Similarly, we parse $\mathbf{t}$ as $\mathbf{t} = [c_1|\mathsf{tag}_1|t_1|\ldots|c_1|\mathsf{tag}_{Q_{\mathsf{Sign}}}|t_{Q_{\mathsf{Sign}}}|b_1|\ldots|b_{Q_{\mathcal{H}_1}}]^T$.

For the i-th query to $\mathcal{H}_1$ made by the adversary with input seed, we program it to output $\mathcal{H}_1(\mathsf{seed}) = u_i$. Then, during the j-th query to $\mathcal{O}\mathsf{Sign}(i^+, *, *)$, the challenger samples seed, and programs $\mathcal{H}_1(\mathsf{seed}) = a'_j$, $c = c_j$, $\mathcal{H}_3(\mathsf{seed}, c) = \mathbf{a}_j$, $\mathsf{tag} = \mathsf{tag}_j$, $h = h_j$ and $t = t_j$ by using the elements of the R-LWE instance. All these elements are well-distributed by construction. Also, when performing the $\mathsf{Identify}$ test in step 10 of the signing procedure, it gets $(\mathsf{seed}_i, c_i, \mathsf{tag}_i)$ from the SRL and look for the $u_i = \mathcal{H}(\mathsf{seed}_i)$ (or the a'_j if seed_i was emitted by i^+ in a previous query) from the random oracle queries. It then takes the corresponding b_i (or c_j) from the R-LWE instance and checks if $\|c_i - b_i\|_\infty \leq 2\eta$ or not.

We also change the answer to $\mathcal{O}\mathsf{Cor}(i^+)$. If such a query is made, the challenger aborts the game. As mentionned in $\mathtt{Game}_1$, the latter case does not occur if the guess i^+ is correct, otherwise $\mathcal{A}$ would automatically lose the game.

We note that even though we inserted a R-LWE instance, everything follows the exact same distribution, except if the adversary has queried $\mathcal{H}_2$ on $(s, *)$ in which case we can recover s from the hash query and solve R-LWE directly. We now assume such a query to $\mathcal{H}_2$ is not made. In that case, the only difference is then that the challenger of the non-frameability game is now oblivious to the secret s of the challenge platform i^+.

At the end of the game, $\mathcal{A}$ produces $(M, \mathsf{sig}, \mathsf{SRL})$ such that $\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}, M, \mathsf{SRL}, \mathsf{KRL}) = 1$ and there exists an honest (i.e., non corrupt) platform $i^\star$ such that $\mathsf{Identify}(\mathsf{T}_{\mathsf{Join}}[i^\star], \mathsf{SigRevoke}(\{\mathsf{sig}\})) = 1$ (and where sig was never obtained from a query to $\mathcal{O}\mathsf{Sign}(i^\star, *, *)$). We note that by definition of $\mathtt{Game}_1$, we have $i^+ = i^\star$. We then parse sig as $\mathsf{sig} = (\mathsf{seed}, c, \mathsf{tag}, h, t, x_{1,2}, \ldots, x_{|\mathsf{SRL}|,2}, \pi_2)$. Because the sig verifies, the proof π_2 present in sig must be valid. Then, except with the soundness error $\varepsilon_{snd,2}$ of the NIZK $(\mathsf{Prove}_2, \mathsf{Verify}_2)$, the challenger can extract s' from the proof and return it as the R-LWE solution. By definition of a valid type ❷ attack, it holds that $s' = s_{i^\star}$ which is the secret of platform $i^\star$. Because $i^+ = i^\star$, this is the secret of platform i^+ which therefore corresponds to the correct R-LWE secret. So the challenger exploited the adversary to solve R-LWE. It yields

$$\mathrm{Adv}_{\mathtt{Game}_9}[\mathcal{A}] \leq \varepsilon_{snd,2} + \varepsilon_{\text{R-LWE}}.$$

where $\varepsilon_{\text{R-LWE}}$ is the hardness bound of $\text{R-LWE}_{n_1, 4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1}, p, U(S_1), U(S_\eta)}$ where $S_\alpha = \tau_1^{-1}([-\alpha, \alpha]^{n_1})$.

Combining everything. We can now combine the advantage from type ❶ and ❷ (with all the advantage bounds) by flipping a coin at the outset to decide which type of attack is expected. We then have

$$\begin{aligned}
\mathrm{Adv}_{nfra}[\mathcal{A}] &\leq 2 \cdot \max(\mathrm{Adv}_{\text{type ❶}}[\mathcal{A}] \ , \ \ \mathrm{Adv}_{\text{type ❷}}[\mathcal{A}]) \\
&\lesssim 2 \cdot \max\big(\varepsilon_{\text{R-SIS}} + \varepsilon_{snd,2} \ , \ \ Q_{\mathsf{Join}}\big(\varepsilon_{zk,1} + Q_{\mathsf{Sign}}\varepsilon_{zk,2} + \varepsilon_{\text{M-LWE}} \\
&\qquad + \kappa Q_{\mathcal{H}_3} p^{-n_1/\kappa} + (Q_{\mathcal{H}_3} + Q_{\mathsf{Sign}})\varepsilon_{\text{NTRU}} + 3N_{\mathsf{SRL}}Q_{\mathsf{Sign}}\varepsilon_F \\
&\qquad + \varepsilon_{snd,2} + \varepsilon_{\text{R-LWE}})\big)
\end{aligned}$$

We have an approximate inequality $\lesssim$ instead of $\leq$ because we approximated the loss terms featuring ε_F by their equivalent when ε_F tends to 0. This is done only for clarity of presentation. □

Theorem 5.3 (Anonymity). *The EPID system described in Section 4 is ε_{an}-anonymous, with*

$$\varepsilon_{an} \approx Q_{\mathsf{Join}} \Big(\varepsilon_{zk,1} + Q_{\mathsf{Sign}} \varepsilon_{zk,2} + \varepsilon_{\text{M-LWE}} + \kappa Q_{\mathcal{H}_3} p^{-n_1/\kappa} + (Q_{\mathcal{H}_3} + Q_{\mathsf{Sign}}) \varepsilon_{\text{NTRU}} + 3 N_{\mathsf{SRL}} Q_{\mathsf{Sign}} \varepsilon_F + \varepsilon_{\text{R-LWE}} \Big),$$

where $Q_{\mathsf{Join}}, Q_{\mathsf{Sign}}, Q_{\mathcal{H}_1}, Q_{\mathcal{H}_2} Q_{\mathcal{H}_3}$ are the maximal number of queries to Join *(per issuer key pair),* Sign *(per platform), $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{H}_3$ respectively. The quantities $\varepsilon_{\text{M-LWE}}, \varepsilon_{\text{R-LWE}}, \varepsilon_{\text{NTRU}}$ then are the hardness bounds of $\text{M-LWE}_{n_2,d,d,q,U(T_1),U(T_1)}$ (with $T_1 = \tau_2^{-1}(\{0,1\}^{n_2})$), decision $\text{R-LWE}_{n_1,4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1},p,U(S_1),U(S_\eta)}$ (with $S_\alpha = \tau_1^{-1}([-\alpha,\alpha]^{n_1})$), and $\text{NTRU}_{n_2,p,\mathcal{D}_{\mathcal{R}_1,\sigma_{f,g}}}$ respectively. Finally, $\varepsilon_{zk,i}$ denotes the zero-knowledge loss of* Prove_i.

Proof. At a high level, we introduce several hybrid changes so as to simulate the EPID signatures of the challenge platform without resorting to its secret key sk_ρ. In the anonymity game from Figure 3.1, this would mean that $\mathcal{A}$ has advantage 0. The proof follows the same blueprint as the non-frameability proof in the type ❷ case. Our goal is that instead of inserting a *search* R-LWE instance in the end, we replace all the tag, c, t emitted by the challenge platform $\mathcal{P}^+$ by uniform elements under the *decision* R-LWE assumption. As the steps are exactly the same, we only describe them briefly and refer to the proof of Theorem 5.2 for the details.

We define $\mathtt{Game}_0^\rho = \mathtt{Game}_{\mathcal{A}}^{an-\rho}$ to be the original anonymity game. We note that the games are parameterized by ρ but every change we make is done for both experiments, i.e., for both $\rho = 0$ and $\rho = 1$.

$\mathtt{Game}_1^\rho$ (identity guess): The challenger proceeds as in $\mathtt{Game}_0^\rho$ except that it samples $i^+ \hookleftarrow U([Q_{\mathsf{Join}}])$ at the outset of the game. We denote by $\mathcal{P}^+$ the platform that is registered at the i^+-th call to $\mathcal{O}\mathsf{Join}_{hon}$. The challenger proceeds as usual except that at step 5 of the anonymity game where it aborts and the game returns 0 if $\mathsf{sk}_\rho \neq \mathsf{sk}^+$ where sk^+ is the signing key of platform $\mathcal{P}^+$. Because i^+ is uniform and independent of the adversary's view, conditioning on this event happening yields $\mathbb{P}[\mathtt{Game}_1^\rho(1^\lambda) = 1] = \mathbb{P}[\mathsf{sk}^+ = \mathsf{sk}_\rho]\mathbb{P}[\mathtt{Game}_1^\rho(1^\lambda) = 1 | \mathsf{sk}^+ = \mathsf{sk}_\rho] + 0 = \mathbb{P}[\mathtt{Game}_0^\rho(1^\lambda) = 1]/Q_{\mathsf{Join}}$. Plugging this equation for $\rho = 0$ and $\rho = 1$ into the advantage, it gives

$$\text{Adv}_{\mathtt{Game}_0}[\mathcal{A}] = Q_{\mathsf{Join}} \cdot \text{Adv}_{\mathtt{Game}_1}[\mathcal{A}].$$

As for the non-frameability proof, we can now assume that there will be no query $\mathcal{O}\mathsf{Cor}^\star(\mathsf{sig})$ where sig was emitted by $\mathcal{P}^+$, otherwise $\mathcal{A}$ will have corrupted $\mathcal{P}^+$ and would thus lose the game.

$\mathtt{Game}_2^\rho$ (simulating join proof for $\mathcal{P}^+$): We now simulate the issuance proof π_1 in the i^+-th query to $\mathcal{O}\mathsf{Join}_{hon}$. As argued in $\mathtt{Game}_2$ of the proof of Theorem 5.2, the zero-knowledge property gives

$$\text{Adv}_{\mathtt{Game}_1}[\mathcal{A}] \leq \text{Adv}_{\mathtt{Game}_2}[\mathcal{A}] + \varepsilon_{zk,1},$$

$\underline{\mathtt{Game}_3^\rho \text{ (simulating sign proofs for } \mathcal{P}^+)}$: We now simulate the signing proofs π_2 emitted by $\mathcal{P}^+$ during queries to $\mathcal{O}\mathsf{Sign}^\star$. As argued in $\mathtt{Game}_3$ of the proof of Theorem 5.2, the zero-knowledge property gives

$$\mathrm{Adv}_{\mathtt{Game}_2}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_3}[\mathcal{A}] + Q_{\mathsf{Sign}} \cdot \varepsilon_{zk,2},$$

$\underline{\mathtt{Game}_4^\rho \text{ (simulating commitment for } \mathcal{P}^+)}$: We now simulate the commitment $\mathbf{c}$ in the i^+-th query to $\mathcal{O}\mathsf{Join}_{hon}$. As argued in $\mathtt{Game}_4$ of the proof of Theorem 5.2, the M-LWE$_{n_2,d,d,q,U(T_1),U(T_1)}$ (for $T_1 = \tau_2^{-1}(\{0,1\}^{n_2})$) assumption gives

$$\mathrm{Adv}_{\mathtt{Game}_3}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_4}[\mathcal{A}] + \varepsilon_{\text{M-LWE}},$$

$\underline{\mathtt{Game}_5^\rho \text{ (ensuring unit for other platforms)}}$: We now ensure the first components of $\mathcal{H}_3(\mathsf{seed}, c)$ is a unit for all queries to $\mathcal{H}_3$ other than that made in queries to $\mathcal{O}\mathsf{Sign}^\star$ on behalf of $\mathcal{P}^+$. As argued in $\mathtt{Game}_5$ of the proof of Theorem 5.2, Lemma 2.4 gives

$$\mathrm{Adv}_{\mathtt{Game}_4}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_5}[\mathcal{A}] + 1 - \left(1 - \frac{1}{p^{n_1/\kappa}}\right)^{\kappa Q_{\mathcal{H}}} \leq \mathrm{Adv}_{\mathtt{Game}_5}[\mathcal{A}] + \frac{\kappa Q_{\mathcal{H}}}{p^{n_1/\kappa}},$$

$\underline{\mathtt{Game}_6^\rho \text{ (hiding trapdoor for other platforms)}}$: We now hide a Falcon trapdoor in the second component of $\mathcal{H}_3(\mathsf{seed}, c)$ for all queries to $\mathcal{H}_3$ other than that made in queries to $\mathcal{O}\mathsf{Sign}^\star$ on behalf of $\mathcal{P}^+$. As argued in $\mathtt{Game}_6$ of the proof of Theorem 5.2, the NTRU$_{n_1,p,\mathcal{D}_{\mathcal{R}_1,\sigma_{f,g}}}$ assumption gives

$$\mathrm{Adv}_{\mathtt{Game}_5}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_6}[\mathcal{A}] + Q_{\mathcal{H}_3} \cdot \varepsilon_{\text{NTRU}},$$

$\underline{\mathtt{Game}_7^\rho \text{ (hiding preimages for } \mathcal{P}^+)}$: We now reprogram the queries to $\mathcal{H}_4$ with input $(\mathsf{seed}, c, \mathsf{seed}_i, c_i)$ made by the challenger when answering a $\mathcal{O}\mathsf{Sign}^\star$ query on behalf of $\mathcal{P}^+$. As argued in $\mathtt{Game}_7$ of the proof of Theorem 5.2, the challenger is able to detect when they do not have a trapdoor to do so (but in that case they should return $\mathsf{sig} = \perp$ anyway and so do not have to simulate the $\mathcal{H}_4(\mathsf{seed}, c, \mathsf{seed}_i, c_i)$), and if they do, the simulatability of Falcon preimages of Lemma 2.6 gives

$$\mathrm{Adv}_{\mathtt{Game}_6}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_7}[\mathcal{A}] + N_{\mathsf{SRL}} \cdot Q_{\mathsf{Sign}} \cdot \frac{1}{2}\left(\left(\frac{1+\varepsilon_F/2n_1}{1-\varepsilon_F/2n_1}\right)^{2n_1} - 1\right).$$

$\underline{\mathtt{Game}_8^\rho \text{ (simulate } x_{i,2} \text{ for } \mathcal{P}^+)}$: We now simulate the $x_{i,2}$ component of a signature when answering a $\mathcal{O}\mathsf{Sign}^\star$ query on behalf of $\mathcal{P}^+$. As argued in $\mathtt{Game}_8$ of the proof of Theorem 5.2, Lemma 2.6 again with Lemma 2.7 give

$$\mathrm{Adv}_{\mathtt{Game}_7^\rho}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_8^\rho}[\mathcal{A}] + N_{\mathsf{SRL}} \cdot Q_{\mathsf{Sign}} \cdot \frac{1}{2}\left(\left(\frac{1+\varepsilon_F/2n_1}{1-\varepsilon_F/2n_1}\right)^{2n_1} - 1 + \frac{2\varepsilon_F}{1+\varepsilon_F}\right),$$

$\underline{\mathtt{Game}_9^{\rho}}$ (simulate h for $\mathcal{P}^+$): We now simulate the Falcon keys h when answering a $\mathcal{O}\mathsf{Sign}^\star$ query on behalf of $\mathcal{P}^+$. As argued in $\mathtt{Game}_9$ of the proof of Theorem 5.2, the $\mathrm{NTRU}_{n_1,p,\mathcal{D}_{\mathcal{R}_1,\sigma_{f,g}}}$ assumption gives

$$\mathrm{Adv}_{\mathtt{Game}_8}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_9}[\mathcal{A}] + Q_{\mathsf{Sign}} \cdot \varepsilon_{\mathrm{NTRU}},$$

$\underline{\mathtt{Game}_{10}^{\rho}}$ (simulate tag, c and t for $\mathcal{P}^+$): We now simulate all the tag, c and t emitted when answering a $\mathcal{O}\mathsf{Sign}^\star$ query on behalf of $\mathcal{P}^+$. All these elements are now replaced by uniformly random elements, and so are the $\mathcal{H}_1(\mathsf{seed})s + \mathcal{H}_2(s, \mathsf{seed})$ in the test of step 10 of the Sign algorithm. This change cannot be distinguished by the adversary under the *decision* R-LWE assumption. Concretely, the challenger receives a *decision* R-LWE instance $(\mathbf{a}, \mathbf{t}) \in \mathcal{R}_{1,p}^{4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1}} \times \mathcal{R}_{1,p}^{4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1}}$. We parse $\mathbf{a}$ as $\mathbf{a} = [a'_1|\mathbf{a}_1^T|h_1|\ldots|a'_{Q_{\mathsf{Sign}}}|\mathbf{a}_{Q_{\mathsf{Sign}}}^T|h_{Q_{\mathsf{Sign}}}|u_1|\ldots|u_{Q_{\mathcal{H}_1}}]^T$. Similarly, we parse $\mathbf{t}$ as $\mathbf{t} = [c_1|\mathsf{tag}_1|t_1|\ldots|c_1|\mathsf{tag}_{Q_{\mathsf{Sign}}}|t_{Q_{\mathsf{Sign}}}|b_1|\ldots|b_{Q_{\mathcal{H}_1}}]^T$.

For the i-th query to $\mathcal{H}_1$ made by the adversary with input seed, we program it to output $\mathcal{H}_1(\mathsf{seed}) = u_i$. Then, during the j-th query to $\mathcal{O}\mathsf{Sign}^\star$ on behalf of $\mathcal{P}^+$, the challenger samples seed, and programs $\mathcal{H}_1(\mathsf{seed}) = a'_j$, $c = c_j$, $\mathcal{H}_3(\mathsf{seed}, c) = \mathbf{a}_j$, $\mathsf{tag} = \mathsf{tag}_j$, $h = h_j$ and $t = t_j$ by using the elements of the R-LWE instance. Also, when performing the Identify test in step 10 of the signing procedure, it gets $(\mathsf{seed}_i, c_i, \mathsf{tag}_i)$ from the SRL and look for the $u_i = \mathcal{H}(\mathsf{seed}_i)$ (or the a'_j if seed_i was emitted by i^+ in a previous query) from the random oracle queries. It then takes the corresponding b_i (or c_j) from the R-LWE instance and checks if $\|c_i - b_i\|_\infty \leq 2\eta$ or not.

As for the non-frameability proof, we also need to change the behavior of queries $\mathcal{O}\mathsf{Cor}^\star(\mathsf{sig})$ where $\mathsf{T}_{\mathsf{Sign}}[\mathsf{sig}] = \mathsf{sk}^+$ is the key of platform $\mathcal{P}^+$. If such a query is made, the challenger aborts the game. As mentionned in $\mathtt{Game}_1$, the latter case does not occur if the guess i^+ is correct, otherwise $\mathcal{A}$ would automatically lose the game.

So if $\mathbf{t} = \mathbf{a}s + \mathbf{e} \bmod p\mathcal{R}_1$, then the challenger perfectly simulate $\mathtt{Game}_9^{\rho}$. And if $\mathbf{t}$ is uniform, then it perfectly simulate $\mathtt{Game}_{10}^{\rho}$. It then yields

$$\mathrm{Adv}_{\mathtt{Game}_9}[\mathcal{A}] \leq \mathrm{Adv}_{\mathtt{Game}_{10}}[\mathcal{A}] + \varepsilon_{\text{R-LWE}},$$

where $\varepsilon_{\text{R-LWE}}$ is the hardness bound of $\text{R-LWE}_{n_1,4Q_{\mathsf{Sign}}+Q_{\mathcal{H}_1},p,U(S_1),U(S_\eta)}$ where $S_\alpha = \tau_1^{-1}([-\alpha,\alpha]^{n_1})$. We insist that as opposed to the non-frameability proof where we use a *search* R-LWE instance, we here use a *decision* instance. As for the non-frameability, the adversary could also query $\mathcal{H}_2$ on (s, seed), but this would mean it solved the search instance of R-LWE anyway.

Bounding advantage in $\underline{\mathtt{Game}_{10}}$: As the signatures emitted by platform $\mathcal{P}^+$ no longer depends on sk^+ (and because $\mathsf{sk}^+ = \mathsf{sk}_\rho$ by our guess in $\mathtt{Game}_1^{\rho}$), it means that the view of the adversary no longer depends on ρ. As a result, we have $\mathrm{Adv}_{\mathtt{Game}_{10}}[\mathcal{A}] = 0$. Combining all the previous advantage equations gives

$$\begin{aligned}\mathrm{Adv}_{an}[\mathcal{A}] \lesssim Q_{\mathsf{Join}} \Big(&\varepsilon_{zk,1} + Q_{\mathsf{Sign}}\varepsilon_{zk,2} + \varepsilon_{\text{M-LWE}} + \kappa Q_{\mathcal{H}_3} p^{-n_1/\kappa} \\ &+ (Q_{\mathcal{H}_3} + Q_{\mathsf{Sign}})\varepsilon_{\mathrm{NTRU}} + 3N_{\mathsf{SRL}}Q_{\mathsf{Sign}}\varepsilon_F + \varepsilon_{\text{R-LWE}}\Big)\end{aligned}$$

We have an approximate inequality $\lesssim$ instead of $\leq$ because we approximated the loss terms featuring ε_F by their equivalent when ε_F tends to 0. This is done only for clarity of presentation. □

Theorem 5.4 (Unforgeability). *The EPID system described in Section 4 is ε_{unf}-unforgeable, with $\varepsilon_{unf} = 3\max(\lambda_1, \lambda_2, \lambda_4)$ where $\lambda_1 = N_{\mathsf{SRL}}\varepsilon_{snd,2}$, $\lambda_2 = \varepsilon_{nfra}$ defined in Theorem 5.2, and*

$$\lambda_4 \approx N_{\mathsf{SRL}}\left(\varepsilon_{snd,2} + 2\max\left(h^{\circ d}\left(C(|\mathcal{T}_w| - Q_{\mathsf{Join}})\varepsilon_{\text{M-SIS},1}\right),\right.\right.$$
$$\left.\left.C^2\frac{n_1}{n_2}\varepsilon_{\text{M-LWE}} + \frac{1+\varepsilon}{1-\varepsilon}\left(\varepsilon_{snd,1} + 4M_{1,1}M_{1,2}h^{\circ d}\left(\frac{C}{(1-p_{\text{tail}})^4}Q_{\mathsf{Join}}\varepsilon_{\text{M-SIS},2}\right)\right)\right)\right),$$

where $\varepsilon_{snd,i}$ is the soundness error of Prove_i, $\varepsilon_{\text{M-LWE}}, \varepsilon_{\text{M-SIS},1}, \varepsilon_{\text{M-SIS},2}$ are the respective hardness bounds of $\text{M-LWE}_{n_2,d,d,q,\mathcal{B}_1,\mathcal{B}_1}$, $\text{M-SIS}_{n_2,d,2d+k-\ell+2,q,\beta_1}$, and $\text{M-SIS}_{n_2,d,2d+k-\ell,q,\beta_2}$ with

$$\beta_1 = \sqrt{\left(\sqrt{B_{1,1}^2 + B_{1,2}^2} + \sqrt{n_2 d}B_2\right)^2 + B_3^2 + n_1 + 1}$$
$$\beta_2 = 2\sqrt{\left(\sqrt{B_{1,1}^2 + B_{1,2}^2} + \sqrt{n_2 d}\sqrt{B_2^2 + n_1}\right)^2 + B_3^2}$$

The constant $C \approx 2$ is the one from [AGJ+24, Lem. 2.3], and the probability p_{tail} is the tailcut probability used to set the Gaussian verifications bounds $B'_{1,1}, B'_{1,2}, B_2, B_3$. Finally, the function $h^{\circ d}$ corresponds to the d-th composition power of the function h defined by

$$h(x) = (k-\ell)\varepsilon_{\text{M-LWE}} + \delta\left(2(k-\ell)\varepsilon_{\text{M-LWE}} + \delta\left((k-\ell)\varepsilon_{\text{M-LWE}} + x\right)^{\frac{2\lambda-1}{2\lambda}}\right)^{\frac{2\lambda-1}{2\lambda}},$$

with

$$\delta = 1 + Q_{\mathsf{Join}}\left(\lambda - \frac{1}{2}\right)\left(\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{2d(\ell+1)(n_2-1)+4d+4}\left(\frac{1+\varepsilon/n_2dk}{1-\varepsilon/n_2dk}\right)^{2n_2dk} - 1\right)^2$$

The expression of λ_4 features an intricate term with the function $h^{\circ d}$. At first glance, it is not trivial to see what security loss this term entails. Fortunately, the authors of [AGJ+24, App. A] provide a comprehensive bound on $h^{\circ d}$ and explanations to argue that applying $h^{\circ d}$ only decreases the bit security of the input x by only a few bits (e.g., $2d$ bits which is constant). We refer to the latter for more details.

Proof. We now prove the unforgeability of our scheme. For that, we let $\mathcal{A}$ be a probabilistic polynomial time adversary that wins the unforgeability game from Figure 3.2. It happens only if the loop condition turned out to be false at some point, meaning that $\mathsf{ctr}_{sig} > \mathsf{ctr}_{cor}$. Said differently, it means that $\mathcal{A}$ managed to produce $\mathsf{ctr}_{cor} + 1$ valid EPID signatures $(\mathsf{sig}_i)_i$ on messages $(M_i)_i$, despite

systematic revocation of the previous ones, while having corrupted ctr_{cor} platforms. We partition the possible forgeries with four different conditions, which we later refer to as four diffent types of forgeries. We treat them sequentially for clarity.

- Case (1). One of the $\mathsf{ctr}_{cor}+1$ signatures contains malformed elements, meaning the zero-knowledge proof π_2 was spoofed. This is infeasible based on the soundness of the NIZK.
- Case (2). Case (1) does not happen, but one of the sig_i revokes an honest platform (meaning one of the s_i (used to form tag_i, t_i, c_i in sig_i) is the secret of an honest platform). This case is actually covered by the non-frameability of Theorem 5.2.
- Case (3). Cases (1) and (2) do not happen, but there exists $i < j$ such that $s_i = s_j$ (where s_k is the secret used to form tag_k, t_k, c_k which exist because case (1) does not happen, and where s_k does not revoke an honest user because case (2) does not happen either). In that case, it means that sig_j is valid with respect to $\mathsf{SRL} = \mathsf{SigRevoke}(\{\mathsf{sig}_1, \ldots, \mathsf{sig}_{j-1}\})$ which contains sig_i. As a result, the adversary bypassed the revocation for a non-honest secret. Based on how we chose the bound β, this cannot happen.
- Case (4). None of the previous cases happen. This means that all the elements are well-formed, i.e., from a short secret s_i, that none of the secret s_i revokes an honest platform, and that all the s_i are distinct. Because all the s_i are non-honest and at most ctr_{cor} of them correspond to corrupt secrets, it ensures one of the s_i is neither honest, nor corrupt. This scenario thus means that the adversary managed to register a new platform, breaking the unforgeability of the signature with efficient protocols of [AGJ+24].

We now detail formally each case. For clarity, we denote by $\mathrm{Adv}_{(i)}[\mathcal{A}]$ the advantage of the adversary in winning the unforgeability game in case (i) for $i \in \{1,2,3,4\}$.

Case (1). In the first case, the challenger expects the adversary to spoof at least one of the $\mathsf{ctr}_{cor}+1$ proofs $\pi_{2,i}$. As a result, upon obtaining the $\mathsf{ctr}_{cor}+1$ signatures sig_i, the challenger guesses which proof was falsified by sampling $i^+ \hookleftarrow U([\mathsf{ctr}_{cor}+1])$ and uses it to break the soundness game of the proof system $(\mathsf{Prove}_2, \mathsf{Verify}_2)$. It then directly holds that

$$\mathrm{Adv}_{(1)}[\mathcal{A}] \leq (\mathsf{ctr}_{cor}+1)\varepsilon_{snd,2}$$

where $\varepsilon_{snd,2}$ is the soundness error of the NIZK $(\mathsf{Prove}_2, \mathsf{Verify}_2)$.

Case (2). In case (2), the challenger expects the adversary to produce at least one signature revoking an honest platform. As such, there exists $j \in [\mathsf{ctr}_{cor}+1]$ and $\mathsf{sk}_{i^\star}$ an honest platform key registered via a call to $\mathcal{O}\mathsf{Add}(i^\star)$ such that $\mathsf{Identify}(\mathsf{sk}_{i^\star}, \mathsf{SigRevoke}(\{\mathsf{sig}_j\})) = 1$. Because sig_j does not belong to $\mathsf{S}_{\mathsf{Sign}}$, it means it was never obtained from a query to $\mathcal{O}\mathsf{Sign}(i^\star, *, *)$. Thence, $(M_j, \mathsf{sig}_j, \mathsf{SRL}_{:j-1})$ satisfies the winning conditions of the non-frameability game of Figure 3.3 (where $\mathsf{SRL}_{:j-1} = \mathsf{SigRevoke}(\{\mathsf{sig}_1, \ldots, \mathsf{sig}_{j-1}\})$). Because the challenger knows all the honest keys, it can easily test which signature sig_j revokes

which honest key sk_{i^*}, without having to guess which of the $\mathsf{ctr}_{cor}+1$ signatures is involved. In addition, the oracles in the unforgeability game are the same than for the non-frameability one, at the exception that the adversary is also given isk in the non-frameability game (which gives it more power). Hence, we can directly argue that case (2) is covered by the non-frameability we proved in Theorem 5.2. We then have

$$\mathrm{Adv}_{(2)}[\mathcal{A}] \leq \varepsilon_{nfra},$$

where ε_{nfra} is defined in Theorem 5.2.

Case (3). In this case, we can assume that the proofs are well-formed as case (1) does not happen. This means that for each of the $\mathsf{ctr}_{cor}+1$ signatures $\mathsf{sig}_i = (\mathsf{seed}_i, c_i, \mathsf{tag}_i, h_i, t_i, x_{1,2}, \ldots, x_{|\mathsf{SRL}_{:i-1}|,2}, \pi_{2,i})$, there exists a tuple $(s_i, \mathfrak{t}_i, \mathbf{v}_{1,1,i}, \mathbf{v}_{1,2,i}, \mathbf{v}_{2,i}, \mathbf{v}_{3,i}, \mathbf{e}_i, e_i, e'_i)$ that verifies the relation described in Equations (2) to (7). More precisely, it holds that

$$\mathbf{v}_{1,1,i} + \mathbf{A}\mathbf{v}_{1,2,i} + (\mathfrak{t}_i\mathbf{G}_H - \mathbf{B})\mathbf{v}_{2,i} + \mathbf{A}_3\mathbf{v}_{3,i} = \mathbf{u} + \mathbf{D}\widetilde{\theta}_{1,2}(s_i) \bmod q\mathcal{R}_2 \tag{8}$$

$$\|\mathbf{v}_{1,1,i}\|_2 \leq B_{1,1}, \quad \|\mathbf{v}_{1,2,i}\|_2 \leq B_{1,2}, \quad \|\mathbf{v}_{2,i}\|_2 \leq B_2, \quad \|\mathbf{v}_{3,i}\|_2 \leq B_3, \quad \mathfrak{t}_i \in \mathcal{T}_w \tag{9}$$

$$c_i = \mathcal{H}_1(\mathsf{seed}_i)s_i + e_i \bmod p\mathcal{R}_1 \tag{10}$$

$$\mathsf{tag}_i = \mathcal{H}_3(\mathsf{seed}_i, c_i)s_i + \mathbf{e}_i \bmod p\mathcal{R}_1 \tag{11}$$

$$t_i = h_i s_i + e'_i \bmod p\mathcal{R}_1 \tag{12}$$

$$\|s_i\|_2 \leq \sqrt{n_1}, \quad \|e_i\|_2 \leq \eta\sqrt{n_1}, \quad \|\mathbf{e}_i\|_2 \leq \eta\sqrt{2n_1}, \quad \|e'_i\|_2 \leq \eta\sqrt{n_1} \tag{13}$$

Per definition of case (3), there exists $i < j$ such that $s_i = s_j$. We now show this case cannot happen based on the verification of non-revocation we perform. It holds that $\mathsf{Verify}(\mathsf{ipk}, \mathsf{sig}_j, M_j, \mathsf{SRL}_{:j-1}, \mathsf{KRL}) = 1$, which necessarily entails that the i-th iteration of the loop in Algorithm 4.8 gives $\mu = 1$ at both checks. It means that

$$\|[x_{i,1}|x_{i,2}]\|_2 \leq \beta_F, \quad \text{and} \quad \left\|x_{i,2}t_j - \boldsymbol{\gamma}_i^T\mathsf{tag}_i \bmod p\mathcal{R}_1\right\|_\infty > \beta, \tag{14}$$

where $\mathbf{a}_i = \mathcal{H}_3(\mathsf{seed}_i, c_i)$, $\boldsymbol{\gamma}_i = \mathcal{H}_4(\mathsf{seed}_j, c_j, \mathsf{seed}_i, c_i)$, $x_{i,1} = \boldsymbol{\gamma}_i^T\mathbf{a}_i - h_j x_{i,2} \bmod p\mathcal{R}_1$, and where $x_{i,2}$ is the i-th revocation token present in sig_j (i.e., the one with respect to $\mathsf{SRL}_{:j-1}[i]$). From Equations (11) and (12) for indices i and j respectively, we have

$$\begin{aligned} x_{i,2}t_j - \boldsymbol{\gamma}_i^T\mathsf{tag}_i &= \boldsymbol{\gamma}_i^T\mathbf{a}_i(s_j - s_i) + x_{i,2}e'_j - x_{i,1}s_j - \boldsymbol{\gamma}_i^T\mathbf{e}_i \bmod p\mathcal{R}_1 \\ &= x_{i,2}e'_j - x_{i,1}s_j - \boldsymbol{\gamma}_i^T\mathbf{e}_i \bmod p\mathcal{R}_1, \end{aligned}$$

where the second equality stems from the fact that $s_i = s_j$ by assumption of case (3). We now have

$$\begin{aligned} \left\|x_{i,2}e'_j - x_{i,1}s_j - \boldsymbol{\gamma}_i^T\mathbf{e}_i\right\|_\infty &= \left\|\langle [x_{i,1}|x_{i,2}|\boldsymbol{\gamma}_i^T]\,,\ [-s_j|e'_j|\mathbf{e}_i^T]\rangle\right\|_\infty \\ &\leq \left\|[x_{i,1}|x_{i,2}|\boldsymbol{\gamma}_i^T]\right\|_2 \cdot \left\|[-s_j|e'_j|\mathbf{e}_i^T]\right\|_2 \\ &\leq \sqrt{\beta_F^2 + \beta_F^2} \cdot \sqrt{n_1 + \eta^2 n_1 + \eta^2 \cdot 2n_1} \\ &= \sqrt{2}\beta_F \cdot \sqrt{(1+3\eta^2)n_1} \\ &= \beta, \end{aligned}$$

where the first inequality is a Cauchy-Schwarz-like inequality which holds when $\mathcal{R}_1$ is a power-of-two cyclotomic ring, which is the case, and where the second inequality stems from Equation (13) and the first part of Equation (14). As a result, we have $\left\|x_{i,2}t_j - \boldsymbol{\gamma}_i^T \mathsf{tag}_i \bmod p\mathcal{R}_1\right\| \leq \beta$ which contradicts the second part of Equation (14). It then means that case (3) cannot happen. We thus deduce

$$\mathrm{Adv}_{(3)}[\mathcal{A}] = 0,$$

Case (4). We now assume all previous cases do not happen. Because case (1) does not happen, we again have that for all $i \in [\mathsf{ctr}_{cor}+1]$, Equations (8) to (13) hold. Then, because cases (2) and (3) do not happen, we also know that all the involved s_i are non-honest and distinct. By s non-honest, we mean that s was either produced by a query to $\mathcal{O}\mathsf{Add}$ but later corrupted by a query to $\mathcal{O}\mathsf{Cor}$, or was produced by a query to $\mathcal{O}\mathsf{Join}_{cor}$, or was never certified by the issuer at all. Overall, it means that all the s_i are either corrupt or not certified. Because they are all distinct, and because at most ctr_{cor} of them are corrupt by definition of ctr_{cor}, there must exists j such that s_j was never certified by $\mathcal{I}$.

The challenger then guesses which signature is concerned by sampling $j^+ \hookleftarrow U([\mathsf{ctr}_{cor}+1])$. It then extracts the proof π_{2,j^+} which is a valid proof, and gets some $(\mathbf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3, s_j)$ that verifies Equations (8) and (9) as well as $\|\widetilde{\theta}_{1,2}(s_j)\|_2 = \|s_j\|_2 \leq \sqrt{n_1}$. If we denote by $\mathbf{m} = \widetilde{\theta}_{1,2}(s_j)$, and if the guess was correct, it means that $(\mathbf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3)$ is a valid forgery for the standard model signature of [AGJ+24,JS25b] for the message $\mathbf{m}$. As signatures are issued blindly, we in fact are in the exact same situation as the anonymous credentials of [AGJ+24] or the recent blind signature of [JS25a] (with the modified sampler of [JS25b]). We can thus simulate the Join oracle using the unforgeability challenger from said anonymous credentials, and in the end return $(\mathbf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3)$ as valid forgery on the new message $\mathbf{m}$ (it is new if the guess j^+ is correct). Adapting the analysis of [AGJ+24] (detailed in [Jeu24]) to the recent sampler of [JS25b] then yields

$$\begin{aligned}\mathrm{Adv}_{(4)}[\mathcal{A}] \lesssim (\mathsf{ctr}_{cor}+1)\Big(&\varepsilon_{snd,2} + 2\max\Big(h^{\circ d}\left(C(|\mathcal{T}_w| - Q_{\mathsf{Join}})\varepsilon_{\text{M-SIS},1}\right),\\ &C^2 k_{1,2}\varepsilon_{\text{M-LWE}} + \frac{1+\varepsilon}{1-\varepsilon}\left(\varepsilon_{snd,1} + 4M_{1,1}M_{1,2}h^{\circ d}\left(\frac{C}{(1-p_{\text{tail}})^4}Q_{\mathsf{Join}}\varepsilon_{\text{M-SIS},2}\right)\right)\Big)\Big),\end{aligned}$$

where $\varepsilon_{snd,i}$ is the soundness error of Prove_i, $\varepsilon_{\text{M-LWE}}, \varepsilon_{\text{M-SIS},1}, \varepsilon_{\text{M-SIS},2}$ are the respective hardness bounds of $\text{M-LWE}_{n_2,d,d,q,\mathcal{B}_1,\mathcal{B}_1}$, $\text{M-SIS}_{n_2,d,2d+k-\ell+2,q,\beta_1}$, and $\text{M-SIS}_{n_2,d,2d+k-\ell,q,\beta_2}$ with

$$\beta_1 = \sqrt{\left(\sqrt{B_{1,1}^2 + B_{1,2}^2} + \sqrt{n_2 d}B_2\right)^2 + B_3^2 + n_1 + 1}$$

$$\beta_2 = 2\sqrt{\left(\sqrt{B_{1,1}^2 + B_{1,2}^2} + \sqrt{n_2 d}\sqrt{B_2^2 + n_1}\right)^2 + B_3^2}$$

The constant $C \approx 2$ is the one from [AGJ+24, Lem. 2.3], and the probability p_{tail} is the tailcut probability used to set the Gaussian verifications bounds

$B'_{1,1}, B'_{1,2}, B_2, B_3$. Finally, the function $h^{\circ d}$ corresponds to the d-th composition power of the function h defined by

$$h(x) = (k-\ell)\varepsilon_{\text{M-LWE}} + \delta\left(2(k-\ell)\varepsilon_{\text{M-LWE}} + \delta\left((k-\ell)\varepsilon_{\text{M-LWE}} + x\right)^{\frac{2\lambda-1}{2\lambda}}\right)^{\frac{2\lambda-1}{2\lambda}},$$

with

$$\delta = 1 + Q_{\mathsf{Join}}\left(\lambda - \frac{1}{2}\right)\left(\left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{2d(\ell+1)(n_2-1)+4d+4}\left(\frac{1+\varepsilon/n_2dk}{1-\varepsilon/n_2dk}\right)^{2n_2dk} - 1\right)^2$$

Combining everything. We can now combine the advantage from all cases by flipping a coin at the outset of the game to decide which type of forgery is expected. Because case (3) cannot happen as we have described, the challenger only needs to sample $\mathsf{coin} \hookleftarrow U(\{1,2,4\})$. We then have

$$\mathrm{Adv}_{unf}[\mathcal{A}] \leq 3\max\left(\mathrm{Adv}_{(1)}[\mathcal{A}], \mathrm{Adv}_{(2)}[\mathcal{A}], \mathrm{Adv}_{(4)}[\mathcal{A}]\right).$$

Plugging the derived inequalities yields the theorem. We note that in the unforgeability game, all the first ctr_{sig} signatures are placed on the SRL. If the maximal size of an SRL is N_{SRL}, it means that we must have $\mathsf{ctr}_{cor} < \mathsf{ctr}_{sig} \leq N_{\mathsf{SRL}}$. □

6 Parameter Selection and Performance

6.1 Parameter Selection

Proof of Non-Revocation Parameters. We now explain how we select parameters based on the security requirements we identified. As our revocation mechanism is modular and has limited connection with the registration step and zero-knowledge proofs, we can choose parameters almost independently for each of these components. The most demanding component is the proof of non-revocation, which is the cornerstone of our EPID system. As opposed to regular Falcon signatures [PFH+20] where the main security requirements are $\beta_F \leq p$, ours require $\beta = \sqrt{2}\beta_F\sqrt{(1+3\eta^2)n_1}$ to be sufficiently smaller than $p/2$ so that a non-revoked platform only fails the revocation test with negligible probability. This in turn requires p to be much larger than in the Falcon signature scheme. However, choosing a larger p also decreases the R-LWE robustness, which in turn requires a larger lattice dimension and thus ring degree for $\mathcal{R}_1$. We thus first select the parameters of the proof of non-revocation so that the scheme is correct and yields secure R-LWE and NTRU assumptions. We want to keep the modulus p and dimension the smallest possible as this will represent the major part of the EPID signature size (because of the constant number of uniform elements, and the $x_{i,2}$). We also note that our choice of η in Setup (Algorithm 4.1) is motivated by thwarting algebraic attacks on R-LWE such as that of [AG11]. Indeed, our security proof requires a large number of samples ($n_1(4Q_{\mathsf{Sign}} + Q_{\mathcal{H}_1})$ equations over $\mathbb{Z}$). If the error bound is too small, one can turn R-LWE into a

noiseless polynomial system which can be efficiently solved using root finding algorithms for multivariate polynomials.

Instantiating Samplers. Our choice of using Falcon for (1) generating the $x_{i,2}$, and (2) carrying out the security reduction through the $\mathbf{a}_i$ and $\boldsymbol{\gamma}_i$, is motivated by its compactness. Nevertheless, this component could be swapped for another trapdoor sampling mechanism in a relatively modular way. We however observe that (1) and (2) have different requirements. For (1), we need a functional, secure and efficient sampler as it is used in the scheme itself. On the other hand, the situation for (2) only requires the *existence* of a sampler. As long as one is able to embed a trapdoor in the $\mathbf{a}_i$ that is indistinguishable from random, and that is able to produce preimages $\boldsymbol{\gamma}_i$ following a certain distribution, the security proof can be completed using the same steps we did. In particular, we could use more aggressive parameters for $\mathbf{a}_i$ and $\boldsymbol{\gamma}_i$ even if we do not know a concrete sampler supporting them. This would open up for attacks only if such parameters are *impossible* for a sampler. In other words, any practical attack against the resulting scheme would readily give an impossibility result on samplers, which suggests some margin between the parameters (that we use) supporting concrete reduction and the ones that would yield an insecure system.

Join Parameters. We select the registration parameters by following the parameter selection methodology of [AGJ+24,JS25b]. In particular, the parameters are chosen so that the different assumptions M-SIS$_1$, M-SIS$_2$ and M-LWE and the sampler are secure. The only link with the previous component is the dimension of the embedding $\mathbf{s} = \widetilde{\theta}_{1,2}(s)$ which is $k_{1,2} = n_1/n_2$.

Zero-Knowledge Arguments Parameters. Finally, we use the zero-knowledge proof framework of [LNP22] to instantiate Prove_1 and Prove_2, including the optimizations of [LNP22, Sec 4.4, App. A] and [LN22] to reach smaller proof sizes. As we have discussed in Section 4.2, the zero-knowledge proofs are performed in a subring $\mathcal{R}_3$ of degree n_3 ($< n_2 < n_1$) so as to improve the proof size as well. We refer to [LNPS21,LNP22,AGJ+24] for more details on the benefits of using subrings in lattice zero-knowledge proofs. The ring $\mathcal{R}_3$ also determines the exact embedding formula between $\mathcal{R}_1$ and $\mathcal{R}_2$. The relation in Prove_1 only requires proving equations in $\mathcal{R}_1$, which is already detailed in [AGJ+24,JS25a]. We thus follow the same parameter selection methodology in this case.

For Prove_2, we have to prove relations with two different moduli. As in the verifiable encryption proof in [LNP22,JS25a], one could prove the relations modulo p, which are all linear, using the approximate range proof component of [LNP22]. It allows for choosing the proof modulus $q_{\pi_2} = qq_1$ as in Prove_1 (albeit with a larger q_1). However, doing so would require roughly $q_{\pi_2} \gtrsim p^2$ to ensure the mod-p equations interpreted over $\mathbb{Z}$ do not wrap-around modulo q_{π_2}. Because in our case p will be much larger than q, this approach would be very heavy on the parameters. To avoid it, we decide to set the modulus of π_2 to be $q_{\pi_2} = qp$, which means both moduli must split in 2 prime ideal factors. We also need to choose q appropriately so that $qp - 1$ has an even divisor that enables a sufficient compression for the zero-knowledge proof. We refer to [LNP22, App. A]

for more details but simply note that such a q is easy to find. We could instead change p, but we want to keep it as small as possible.

The security of the zero-knowledge proof framework [LNP22] is based on standard assumptions like M-SIS and M-LWE. In particular, the soundness errors $\varepsilon_{snd,1}, \varepsilon_{snd,2}$ and the zero-knowledge security $\varepsilon_{zk,1}, \varepsilon_{zk,2}$ can be expressed from the hardness bounds of some M-SIS and M-LWE assumptions. Their exact expression is deferred to Lemma A.1 and A.2 in Appendix A.

6.2 Security Estimations

The security of our system is then estimated using the security proofs of Theorems 5.2, 5.3 and 5.4. The hardness bounds for the plethora of lattice assumptions for the three different components (M-SIS, M-LWE, R-SIS, R-LWE, NTRU) is estimated using the lattice estimator [APS15]. The algebraic structure underlying these assumptions is commonly assumed not to incur a noticeable cryptanalytic advantage. We thus estimate these structured assumptions by their unstructured variants (i.e., when embedded in $\mathbb{Z}$). The different hardness bounds are then plugged in the corresponding formulae to derive the overall security. We opt for the more realistic lattice sieving cost model where the hardness bound is estimated as $2^{-(0.292B+16.4)}$, where B is the minimal BKZ blocksize. We nevertheless aim for $\lambda = 128$, while other works would aim for around $\lambda = 110$ but in the Core-SVP model. Our choice of cost model thus matches these works but we believe it is a more transparent and realistic security estimate. We also note that we follow the common practice of not accounting for guessing losses, as long as they are polynomially bounded. As a result, the guessing factor Q_{Join} in ε_{nfra} and ε_{an}, as well as the factor $(\mathsf{ctr}_{cor} + 1) \leq N_{\mathsf{SRL}}$ in ε_{unf} are not taken into account in the security estimation. We nevertheless choose reasonable parameters that keep these loss factor acceptable, while covering a wide range of applications. Concretely, we choose $Q_{\mathsf{Join}} = Q_{\mathsf{Sign}} = 2^{32}$, $Q_{\mathcal{H}_i} = 2^{64}$, and set parameters for different values of N_{SRL} ranging from 0 to 100000.

6.3 Performance and Comparison

Suggested Parameters. Overall, we give in Tables B.1, B.2, B.3 and B.4 a complete example set of parameters to instantiate all the components with, for $N_{\mathsf{SRL}} = 1000$. We also report in Table 6.1 a summary of the efficiency and security estimates for different values of N_{SRL}. The full parameter sets, efficiency and security estimates are computed using a script built upon the lattice estimator, which we make publicly available[8]. We note that the bit size of discrete Gaussian vectors is estimated using the entropy bound, which can be closely approached using the rANS encoding as discussed in [ETWY22]. More precisely, for a discrete Gaussian over $\mathbb{Z}^n$ of parameter σ, the estimated bit size is $n(1/2 + \log_2 \sigma)$. Overall, the size in bits of an EPID signature is computed as

$$|\mathsf{sig}| = 256 + 5n_1\lceil \log_2 p \rceil + |\pi_2| + N_{\mathsf{SRL}} \cdot \lceil n_1(1/2 + \log_2 \sigma_F) \rceil$$

[8] https://github.com/latticeepid/lattice-epid

where $|\pi_2|$ is described in Appendix A for completeness.

N_{SRL}	$\lvert\mathsf{transcript}\rvert$	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{sig}\rvert$	ε_{nfra}	ε_{an}	ε_{unf}
10	27.99 KB	7.09 KB	171.51 KB	127.4	141.1	125.4
100	27.99 KB	7.09 KB	649.29 KB	127.4	141.1	125.4
500	27.99 KB	7.09 KB	2774.74 KB	127.0	141.1	125.3
1000	27.99 KB	7.09 KB	5432.95 KB	127.2	141.1	125.3
10000	27.99 KB	7.09 KB	53327.72 KB	127.3	140.9	125.4
100000	27.99 KB	7.09 KB	532869.96 KB	127.2	140.9	125.3

Table 6.1. Efficiency and security estimates of our EPID scheme for different values of N_{SRL}. The value $\lvert\mathsf{transcript}\rvert$ represents the size of the transcript during the Join protocol. The security estimates are given in bit security (i.e., $-\log_2(x)$). In all cases, the correctness error is at most $2^{-\lambda}$.

We observe that the modularity of our approach ensures the sizes of the transcript, secret key, issuing parameters and the signature (except for the $x_{i,2}$) stay constant with N_{SRL}. The only dependencies in N_{SRL} come with the number of Falcon signatures, and as such the smoothing loss ε_F. The effect of N_{SRL} on ε_F is sufficiently mild to incur no noticeable security loss overall, and as a result the only difference comes in the signature size. Tighter parameters on the proof of non-revocation, especially a smaller modulus p, would then drastically improve our performance and the way the EPID signature size scales with N_{SRL}.

Comparison with Other Post-Quantum EPIDs. Let us now compare our EPID system with the existing post-quantum EPID, comparison which is summarized in Table 6.2. As described in the introduction, there is, to our knowledge, only two constructions based on symmetric primitives [BEF19,CDK+24], and one lattice-based construction [KFM+19]. For completeness, we note that another paper [CXTW23] also focuses on lattice EPID but does not describe the proof of non-revocation which is yet the core component of an EPID system. It is therefore impossible to estimate the performance of their construction, let alone its security, thus preventing any meaningful comparison with other constructions.

The first symmetric-based EPID from [BEF19] achieves signature sizes of 216.82 MB for a group size of 2^{30}. They however do not specify the impact of the SRL and the proof of non-revocation on their system which makes the comparison less obvious. In particular, the size of the SRL, which seems to be the relevant factor in most EPID constructions, is never made explicit. The authors also propose another EPID construction for a specific setting, reaching as low as 5.05 MB for a similar group size. However, in this construction the group is static, which means every time a new platform needs to be registered, all the existing platform have to restart the Join procedure. Beyond the practicality of this approach, this questions the opportunity of using EPID systems in this context. One could indeed leverage the frequent group reset to revoke users,

which makes EPID revocation features less appealing. In all cases, this makes this scheme hardly comparable to other EPID systems.

The recent hash-based construction of [CDK+24] achieves EPID signature sizes of around 11.8 MB for a group size of 2^{40} and a maximal SRL size of 1000. Even though, the size of their proof of non-revocation scales linearly with N_{SRL}, the slope of this linear scaling is much larger than ours which explains our advantage compared to their sizes. In addition, the unit cost of their EPID signature, i.e., the size of the component that does not depend on N_{SRL}, is much larger than ours with around 1 MB for a comparable group size.

Finally, the lattice-based construction of [KFM+19], whose structural differences with our approach are highlighted in Figure 1.1, yields signatures whose size is 854 MB for 1000 revoked signatures and a group size of 2^{32}.

	Type	**Total**
[BEF19]	Symmetric	216.82 MB
[KFM+19]	Lattice	854 MB
[CDK+24]	Symmetric	11.5 MB
Ours	Lattice	5.31 MB

Table 6.2. Comparison of EPID signature sizes of the existing post-quantum constructions. Sizes are given in MB, for $N_{\mathsf{SRL}} = 1000$ and group size of $Q_{\mathsf{Join}} = 2^{32}$, and for a security parameter of $\lambda = 128$.

Regarding timing performances, our reliance on well-known lattice building blocks means our system is also very likely to outperform symmetric constructions relying on general-purpose NIZKs. Even though we leave the implementation of our system for future work, recent papers have shown concrete evidence of the efficiency of the building blocks we use [AGJ+24,LSS24,JS25a,JS25b] for the group signature component. The proof of non-revocation then simply consists in generating one Falcon key and N Falcon signatures which would also be very efficient compared to proving knowledge of $O(N)$-dimensional witnesses using [LNP22] or [BS23].

We also note that our construction is the only one that provably supports malicious revocation lists, which is better aligned with the decentralized spirit of EPID systems. Beyond the sole application to EPID systems, our decentralized revocation strategy is very modular and could benefit other privacy-enhanced authentication mechanisms such as anonymous credentials.

References

AG11. S. Arora and R. Ge. New Algorithms for Learning in Presence of Errors. In *ICALP*, 2011.

AGJ+24. S. Argo, T. Güneysu, C. Jeudy, G. Land, A. Roux-Langlois, and O. Sanders. Practical Post-Quantum Signatures for Privacy. In *CCS*, 2024.

AKSY22. S. Agrawal, E. Kirshanova, D. Stehlé, and A. Yadav. Practical, Round-Optimal Lattice-Based Blind Signatures. In *CCS*, 2022.

APS15. M. R. Albrecht, R. Player, and S. Scott. On the Concrete Hardness of Learning With Errors. *J. Math. Cryptol.*, 2015.

Ban93. W. Banaszczyk. New Bounds in Some Transference Theorems in the Geometry of Numbers. *Math. Ann.*, 1993.

BEF19. D. Boneh, S. Eskandarian, and B. Fisch. Post-quantum EPID Signatures from Symmetric Primitives. In *CT-RSA*, 2019.

BL07. E. Brickell and J. Li. Enhanced Privacy ID: a Direct Anonymous Attestation Scheme with Enhanced Revocation Capabilities. In *WPES*, 2007.

BL10. E. Brickell and J. Li. Enhanced Privacy ID from Bilinear Pairing for Hardware Authentication and Attestation. In *SocialCom PASSAT*, 2010.

BLNS23. J. Bootle, V. Lyubashevsky, N. K. Nguyen, and A. Sorniotti. A Framework for Practical Anonymous Credentials from Lattices. In *CRYPTO*, 2023.

BMW03. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions. In *EUROCRYPT*, 2003.

BS04. D. Boneh and H. Shacham. Group signatures with verifier-local revocation. In *CCS*, 2004.

BS23. W. Beullens and G. Seiler. LaBRADOR: Compact Proofs for R1CS from Module-SIS. In *CRYPTO*, 2023.

BSZ05. M. Bellare, H. Shi, and C. Zhang. Foundations of Group Signatures: The Case of Dynamic Groups. In *CT-RSA*, 2005.

CDK+24. L. Chen, C. Dong, N. El Kassem, C. J. P. Newton, and Y. Wang. A New Hash-Based Enhanced Privacy ID Signature Scheme. In *PQCrypto*, 2024.

Cha82. D. Chaum. Blind Signatures for Untraceable Payments. In *CRYPTO*, 1982.

Cha85. D. Chaum. Showing Credentials Without Identification: Signatures Transferred Between Unconditionally Unlinkable Pseudonyms. In *EUROCRYPT*, 1985.

CL02. Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO*, 2002.

CS03. J. Camenisch and V. Shoup. Practical Verifiable Encryption and Decryption of Discrete Logarithms. In *CRYPTO*, 2003.

CvH91. D. Chaum and E. van Heyst. Group Signatures. In *EUROCRYPT*, 1991.

CXTW23. L. Chen, Z. Xu, T. Tu, and Z. Wang. Lattice-Based Privacy Enhanced Identity Protocol for SDO Services. In *ICSIP*, 2023.

DP16. L. Ducas and T. Prest. Fast Fourier Orthogonalization. In *ISSAC*, 2016.

DY05. Y. Dodis and A. Yampolskiy. A Verifiable Random Function with Short Proofs and Keys. In *PKC*, 2005.

ETWY22. T. Espitau, M. Tibouchi, A. Wallet, and Y. Yu. Shorter Hash-and-Sign Lattice-Based Signatures. In *CRYPTO*, 2022.

EWY23. T. Espitau, A. Wallet, and Y. Yu. On Gaussian Sampling, Smoothing Parameter and Application to Signatures. In *ASIACRYPT*, 2023.

FHS19. G. Fuchsbauer, C. Hanser, and D. Slamanig. Structure-Preserving Signatures on Equivalence Classes and Constant-Size Anonymous Credentials. *J. Cryptol.*, 2019.

GPV08. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for Hard Lattices and New Cryptographic Constructions. In *STOC*, 2008.

HPS98. J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A Ring-Based Public Key Cryptosystem. In *ANTS*, 1998.

ISO13. ISO/IEC. ISO/IEC 20008-2:2013 Information Technology — Security Techniques — Anonymous Digital Signatures — Part 2: Mechanisms using a Group Public Key. https://www.iso.org/standard/56916.html, 2013.

Jeu24. C. Jeudy. *Design of Advanced Post-Quantum Signatures Schemes*. PhD thesis, Université de Rennes 1, Rennes, France, 2024. Accessible at https://tel.archives-ouvertes.fr/tel-04727543v1.

JRS23. C. Jeudy, A. Roux-Langlois, and O. Sanders. Lattice Signature with Efficient Protocols, Application to Anonymous Credentials. In *CRYPTO*, 2023.

JS24. C. Jeudy and O. Sanders. Improved Lattice Blind Signatures from Recycled Entropy. *IACR Cryptol. ePrint Arch.*, page 1289, 2024.

JS25a. C. Jeudy and O. Sanders. Improved Lattice Blind Signatures from Recycled Entropy. In *CRYPTO*, 2025.

JS25b. C. Jeudy and O. Sanders. Worst-Case Lattice Sampler with Truncated Gadgets and Applications. In *ASIACRYPT*, 2025.

KFM+19. N. El Kassem, L. Fiolhais, P. Martins, L. Chen, and L. Sousa. A Lattice-Based Enhanced Privacy ID. In *WISTP*, 2019.

Kle00. P. N. Klein. Finding the closest lattice vector when it's unusually close. In *SODA*, 2000.

LLNW14. A. Langlois, S. Ling, K. Nguyen, and H. Wang. Lattice-Based Group Signature Scheme with Verifier-Local Revocation. In *PKC*, 2014.

LN22. V. Lyubashevsky and N. K. Nguyen. BLOOM: Bimodal Lattice One-Out-of-Many Proofs and Applications. *ASIACRYPT*, 2022.

LNP22. V. Lyubashevsky, N. K. Nguyen, and M. Plançon. Lattice-Based Zero-Knowledge Proofs and Applications: Shorter, Simpler, and More General. *CRYPTO*, 2022.

LNPS21. V. Lyubashevsky, N. K. Nguyen, M. Plançon, and G. Seiler. Shorter Lattice-Based Group Signatures via "Almost Free" Encryption and Other Optimizations. In *ASIACRYPT*, 2021.

LPR10. V. Lyubashevsky, C. Peikert, and O. Regev. On Ideal Lattices and Learning with Errors over Rings. In *EUROCRYPT*, 2010.

LS15. A. Langlois and D. Stehlé. Worst-case to Average-case Reductions for Module Lattices. *DCC*, 2015.

LS18. V. Lyubashevsky and G. Seiler. Short, Invertible Elements in Partially Splitting Cyclotomic Rings and Applications to Lattice-Based Zero-Knowledge Proofs. In *EUROCRYPT*, 2018.

LSS24. V. Lyubashevsky, G. Seiler, and P. Steuer. The LaZer Library: Lattice-Based Zero Knowledge and Succinct Proofs for Quantum-Safe Privacy. In *CCS*, 2024.

MP12. D. Micciancio and C. Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In *EUROCRYPT*, 2012.

MR07. D. Micciancio and O. Regev. Worst-Case to Average-Case Reductions Based on Gaussian Measures. *SIAM J. Comput.*, 2007.

PFH+20. T. Prest, P.-A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Ricosset, G. Seiler, W. Whyte, and Z. Zhang. *FALCON. Tech. rep.*, 2020. Available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022.

Pre17. T. Prest. Sharper Bounds in Lattice-Based Cryptography Using the Rényi Divergence. In *ASIACRYPT*, 2017.

San21. O. Sanders. Improving Revocation for Group Signature with Redactable Signature. In *PKC*, 2021.

San22. O. Sanders. EPID with Efficient Proof of Non-Revocation. In *ASIA CCS*, 2022.
ST21. O. Sanders and J. Traoré. EPID with Malicious Revocation. In *CT-RSA*, 2021.

A Security of the Zero-Knowledge Proof Systems

For completeness, we state here the main security results for the proof systems $(\mathsf{Prove}_i, \mathsf{Verify}_i)$ used in our EPID scheme. They are based on the proof system of [LNP22,LN22], which has since its introduction been used in several constructions, e.g., [LNP22,AKSY22,JRS23,BLNS23,AGJ+24,JS25a,JS25b]. As a result, we only adapt the security statement to our parameters and refer to [LNP22] and subsequent constructions for the details and associated security proofs. We only mention that as we use the bimodal optimizations from [LN22], we include the bimodal bit $\mathfrak{b}$ in the witness. A detailed description of how to embed the relations and adapt them to the bimodal setting is for example given in the full version [JS24, App. A] of [JS25a]. We recall that c_N corresponds to the N-dimensional discrete Gaussian tailcut that yield a Euclidean tail bound with overwhelming probability. To avoid introducing different notations for the two different instantiations of the framework of [LNP22], we use the same parameter names $(n_3, \rho, \widetilde{\eta}, m_1, d_2, m_2, \ell, D, \gamma, \text{ etc.})$ for both. We however note that their actual values are different depending on the relation to prove, as shown in Tables B.3 and B.4. We also denote by $[a^{-1}]_b$ the inverse of a in $\mathbb{Z}_b$ when it exists.

A.1 Join Proof

Lemma A.1 (Security of Prove_1). *Let $\mathcal{R}_3$ be the power-of-two cyclotomic ring of degree n_3. Let $\rho, \widetilde{\eta}$ define the challenge space $\mathcal{C}_{\rho,\widetilde{\eta}} = \{c \in \tau_3^{-1}([-\rho,\rho]^{n_3}) : c^* = c \wedge \sqrt[64]{\|c^{64}\|_1} \leq \widetilde{\eta}\}$. Let M_1, M_2, M_3 be in $(1,\infty)$, and let $\alpha_j = \sqrt{\pi/\ln M_j}$ for $j \in [3]$. Let $B = \sqrt{2n_2d + n_1 + 1}$ be a bound on the witness $\mathbf{s}_1 = (\mathbf{s}, \mathbf{r}_1, \mathbf{r}_2, \mathfrak{b})$ as specified in Section 4.3.2, and $m_1 = k_{2,3}(2d + k_{1,2}) + 2$ be the dimension of the witness embedded in $\mathcal{R}_3$. We then define $\sigma_1 = \alpha_1\widetilde{\eta}B$, $\sigma_2 = \alpha_2\widetilde{\eta}\sqrt{n_3m_2}$, and $\sigma_3 = \alpha_3\sqrt{337}B$. We then define $B_{arp} = \frac{2}{\sqrt{26}}c_{256}\sigma_3\sqrt{256}$. Then, let d_2, m_2, ℓ, q_1 be positive integers such that*

$$q_1 \text{ prime s.t. } q_1 = 5 \bmod 8,$$
$$qq_1 > \max\left(n_1,\ 41n_3m_1B_{arp},\ B_{arp}^2 - n_1,\ B_{arp}\left(B_{arp} + \sqrt{n_3(2dk_{2,3}+1)}\right)\right),$$
$$\left|q_1[q_1^{-1}]_q - q[q^{-1}]_{q_1}\right| > B_{arp}.$$

We then define $q_{\min} = \min(q, q_1)$ and $q_{\pi_1} = qq_1$. We also let γ be an even divisor of $q_{\pi_1} - 1$, and D be the largest integer such that $2^{D-1}\rho n_3 < \gamma$. Then, $(\mathsf{Prove}_1, \mathsf{Verify}_1)$ is knowledge sound with an extractor running in expected polynomial time and soundness error

$$\varepsilon_{snd,1} = \frac{2}{|\mathcal{C}_{\rho,\widetilde{\eta}}|} + q_{\min}^{-n_3/2} + q_{\min}^{-2\ell} + 2^{-128} + \varepsilon^{sound}_{\text{M-SIS}}$$

and zero-knowledge with loss $\varepsilon_{zk,1} = \varepsilon^{zk}_{\text{M-LWE}} + \mathsf{negl}(\lambda)$. *The term* $\varepsilon^{sound}_{\text{M-SIS}}$ *is the hardness bound of* $\text{M-SIS}_{n_3,d_2,m_1+m_2,q_{\pi_1},\beta_{\pi_1}}$ *where*

$$\beta_{\pi_1} = 4\widetilde{\eta}\sqrt{4c^2_{n_3m_1}\sigma_1^2 n_3 m_1 + \left(2c_{n_3m_2}\sigma_2\sqrt{n_3m_2} + (2^D\widetilde{\eta}+\gamma)\sqrt{n_3d_2}\right)^2},$$

while $\varepsilon^{zk}_{\text{M-LWE}}$ *is the hardness bound of* $\text{M-LWE}_{n_3,m_2-d_2-\lfloor 256/n_3\rfloor-\ell-1,m_2,q_{\pi_1},\mathcal{B}_1,\mathcal{B}_1}$ *(with* $\mathcal{B}_1$ *the centered binomial distribution of parameter* 1 *over* $\mathcal{R}_3$*).*

The size of π_1 is computed from the parameters of the proof system introduced in Lemma A.1 by

$$\begin{aligned}|\pi_1| = {} & n_3d_2\left(\lceil\log_2 q_{\pi_1}\rceil - D + 2.25\right) + \left(\frac{256}{n_3} + 2\ell + 1\right)\lceil\log_2 q_{\pi_1}\rceil \\ & + n_3\lceil\log_2(2\rho+1)\rceil + n_3m_1\left(\frac{1}{2} + \log_2\sigma_1\right) \\ & + n_3(m_2-d_2)\left(\frac{1}{2}+\log_2\sigma_2\right) + 256\left(\frac{1}{2}+\log_2\sigma_3\right).\end{aligned}$$

A.2 Sign Proof

Lemma A.2 (Security of Prove_2). *Let* $\mathcal{R}_3$ *be the power-of-two cyclotomic ring of degree* n_3*. Let* $\rho, \widetilde{\eta}$ *define the challenge space* $\mathcal{C}_{\rho,\widetilde{\eta}} = \{c \in \tau_3^{-1}([-\rho,\rho]^{n_3}) : c^* = c \wedge \sqrt[64]{\|c^{64}\|_1} \leq \widetilde{\eta}\}$*. Let* M_1, M_2, M_3 *be in* $(1,\infty)$*, and let* $\alpha_j = \sqrt{\pi/\ln M_j}$ *for* $j \in [3]$*. Let* $B = \sqrt{B_{1,1}^2 + B_{1,2}^2 + B_2^2 + B_3^2 + w + n_1 + 4\eta^2 n_1 + 1}$ *be a bound on the witness* $\mathbf{s}_1 = (s, \mathfrak{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3, \mathbf{e}, e, e', \mathfrak{b})$ *as specified in Section 4.3.2, and*

$$m_1 = k_{2,3}(2d + (k-\ell)(d+1) + 1) + 4k_{1,3} + 9$$

be the dimension of the witness embedded in $\mathcal{R}_3$*. We then define* $\sigma_1 = \alpha_1\widetilde{\eta}B$*,* $\sigma_2 = \alpha_2\widetilde{\eta}\sqrt{n_3m_2}$*, and* $\sigma_3 = \alpha_3\sqrt{337}B$*. We then define* $B_{arp} = \frac{2}{\sqrt{26}}c_{256}\sigma_3\sqrt{256}$*. Then, let* d_2, m_2, ℓ *be positive integers. Also, the moduli* q *and* p *of the EPID system should be chosen such that*

$$\begin{aligned} & qp > \max\left(B_{1,1}^2, B_{1,2}^2, B_2^2, B_3^2, w, n_1, 2\eta^2 n_1\right), \\ & qp > B_{arp}^2 - \min\left(B_{1,1}^2, B_{1,2}^2, B_2^2, B_3^2, w, n_1, 2\eta^2 n_1\right), \\ & qp > \max\left(\ 41n_3m_1B_{arp}, w + \sqrt{wn_2}\right), \\ & \left|p[p^{-1}]_q - q[q^{-1}]_p\right| > B_{arp}. \end{aligned}$$

We then define $q_{\min} = \min(q,p)$ *and* $q_{\pi_2} = qp$*. We also let* γ *be an even divisor of* $q_{\pi_2} - 1$*, and* D *be the largest integer such that* $2^{D-1}\rho n_3 < \gamma$*. Then,* $(\mathsf{Prove}_2, \mathsf{Verify}_2)$ *is knowledge sound with an extractor running in expected polynomial time and soundness error*

$$\varepsilon_{snd,2} = \frac{2}{|\mathcal{C}_{\rho,\widetilde{\eta}}|} + q_{\min}^{-n_3/2} + q_{\min}^{-2\ell} + 2^{-128} + \varepsilon^{sound}_{\text{M-SIS}}$$

and zero-knowledge with loss $\varepsilon_{zk,2} = \varepsilon^{zk}_{\text{M-LWE}} + \mathsf{negl}(\lambda)$. *The term* $\varepsilon^{sound}_{\text{M-SIS}}$ *is the hardness bound of* $\text{M-SIS}_{n_3,d_2,m_1+m_2,q_{\pi_2},\beta_{\pi_2}}$ *where*

$$\beta_{\pi_2} = 4\widetilde{\eta}\sqrt{4c^2_{n_3m_1}\sigma_1^2 n_3 m_1 + \left(2c_{n_3m_2}\sigma_2\sqrt{n_3m_2} + (2^D\widetilde{\eta}+\gamma)\sqrt{n_3 d_2}\right)^2},$$

while $\varepsilon^{zk}_{\text{M-LWE}}$ *is the hardness bound of* $\text{M-LWE}_{n_3,m_2-d_2-\lfloor 256/n_3\rfloor-\ell-1,m_2,q_{\pi_2},\mathcal{B}_1,\mathcal{B}_1}$ *(with* $\mathcal{B}_1$ *the centered binomial distribution of parameter* 1 *over* $\mathcal{R}_3$*).*

The size of π_2 is computed from the parameters of the proof system introduced in Lemma A.2 by

$$\begin{aligned}|\pi_2| = n_3 d_2 \left(\lceil \log_2 q_{\pi_2}\rceil - D + 2.25\right) + \left(\frac{256}{n_3} + 2\ell + 1\right)\lceil \log_2 q_{\pi_2}\rceil \\ + n_3\lceil \log_2(2\rho+1)\rceil + n_3 m_1\left(\frac{1}{2} + \log_2\sigma_1\right) \\ + n_3(m_2-d_2)\left(\frac{1}{2}+\log_2\sigma_2\right) + 256\left(\frac{1}{2}+\log_2\sigma_3\right).\end{aligned}$$

B Suggested Parameter Set

Symbol	Description	Value
	Signature Parameters	
λ	Security parameter	128
n_2	Signature ring degree	256
d	Module rank	4
q	Modulus	$506773 \approx 2^{18.95}$
k	Gadget length	5
ℓ	Number of truncated gadget entries	2
b	Gadget base	14
ε	Smoothing loss for samplers	2^{-40}
s_1	Gaussian width 1 of sampler	5877.412
s_2	Gaussian width 2 of sampler	482.646
s_3	Gaussian width 3 of sampler ($\mathbf{v}_{1,2}$)	5857.561
s_4	Gaussian width 4 of sampler ($\mathbf{v}_2, \mathbf{v}_3$)	83.597
$s_{1,1}$	Final Gaussian width of $\mathbf{v}_{1,1}$	8955.538
w	Hamming weight of tags	5
Q_{Join}	Group size	2^{32}
α_1, α_2	Rejection sampling slack (sec. proof)	$8.48, 5.54$
$M_{1,1}, M_{1,2}$	Rejection sampling repetition rate (sec. proof)	$1.045, 1.108$
$B'_{1,1}$	Verification bound of $\mathbf{v}'_{1,1}$	149905.338
$B'_{1,2}$	Verification bound of $\mathbf{v}'_{1,2}$	98048.794
$B_{1,1}$	Verification bound of $\mathbf{v}_{1,1}$	149937.338
$B_{1,2}$	Verification bound of $\mathbf{v}_{1,2}$	98080.794
B_2	Verification bound of $\mathbf{v}_2$	2174.860
B_3	Verification bound of $\mathbf{v}_3$	1258.307
	Security Estimates	
BKZ❶	Required BKZ blocksize for M-SIS❶	633
BKZ❷	Required BKZ blocksize for M-SIS❷	545
BKZ	Required BKZ blocksize for M-LWE	460
$\varepsilon_{\text{M-SIS},1}$	Hardness bound for M-SIS (1)	$2^{-201.5}$
$\varepsilon_{\text{M-SIS},2}$	Hardness bound for M-SIS (2)	$2^{-175.8}$
$\varepsilon_{\text{M-LWE}}$	Hardness bound for M-LWE	$2^{-150.9}$
	Efficiency Estimates	
$\lvert\mathsf{sig}\rvert$	Size of transcript signature ($\mathsf{t}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3$)	4.89 KB
$\lvert\mathsf{sk}\rvert$	Size of secret key ($s, \mathsf{t}, \mathbf{v}_{1,1}, \mathbf{v}_{1,2}, \mathbf{v}_2, \mathbf{v}_3$)	7.09 KB

Table B.1. Suggested parameter set for the Join parameters.

Symbol	Description	Value
	Signature Parameters	
λ	Security parameter	128
n_1	Signature ring degree	2048
p	Modulus	$55473438037 \approx 2^{35.7}$
Q_{Sign}	Maximal number of signer per platform	2^{32}
N_{SRL}	Maximal size of SRL	1000
η	Error bound	5
α	Falcon trapdoor quality	1.17
$\sigma_{f,g}$	Falcon key sampling width	10792.905
ε_F	Falcon smoothing loss	$2^{-174.55}$
σ_F	Falcon preimage sampling width	1772660.617
β_F	Falcon preimage tail bound	47399304.968
β	Non-revocation bound	26445923884.993
	Security Estimates	
ε_{cor}	Correctness error (Thm 5.1)	$2^{-130.8}$
BKZ	Required BKZ blocksize for R-SIS	14764
BKZ	Required BKZ blocksize for NTRU	1962
BKZ	Required BKZ blocksize for R-LWE	581
$\varepsilon_{\text{R-SIS}}$	Hardness bound for R-SIS	$2^{-4334.6}$
$\varepsilon_{\text{NTRU}}$	Hardness bound for NTRU	$2^{-590.2}$
$\varepsilon_{\text{R-LWE}}$	Hardness bound for R-LWE	$2^{-186.3}$
	Efficiency Estimates	
$\lvert\mathsf{uniform}\rvert$	Size of uniform elements $(\mathsf{seed}, h, c, t, \mathsf{tag})$	45.03 KB
$\lvert x_{1,2}\rvert$	Size of one Falcon signature	5.31 KB
$\lvert\{x_{i,2}\}_i\rvert$	Size of N_{SRL} Falcon signatures	5314.45 KB

Table B.2. Suggested parameter set for the proof of non-revocation.

Symbol	Description	Value
	Proof System Parameters	
λ	Security parameter	128
n_3	Proof system ring degree	64
$k_{2,3}$	Subring embedding dimension ($\theta_{2,3}$)	4
d_2	Module rank	20
q_1	Modulus factor	$523637 \approx 2^{19}$
$q_{\min}$	Smallest modulus factor	506773
q_{π_1}	Proof system modulus (qq_1)	265365093401
ℓ	Soundness amplification dimension	4
m_1	Witness dimension	66
m_2	Dimension of ABDLOP randomness	58
χ	Distribution of ABDLOP randomness	$\mathcal{B}_1$
ρ	Infinity norm of challenges	8
$\widetilde{\eta}$	Manhattan-like norm of challenges	93
$(\alpha_1, \alpha_2, \alpha_3)$	Rejection sampling slacks	$(3.01, 3.01, 3.01)$
(M_1, M_2, M_3)	Rejection sampling repetition rates	$(\sqrt{2}, \sqrt{2}, \sqrt{2})$
σ_1	First rejection sampling width	18197.935
σ_2	Second rejection sampling width	17059.415
σ_3	Third rejection sampling width	3592.147
γ	Mask commitment compression parameter	571538
D	Witness commitment compression parameter	11
	Security Estimates	
$\mathrm{BKZ}^{\mathrm{sound}}_{\text{M-SIS}}$	Required BKZ blocksize for M-SIS, Lem. A.1	403
$\mathrm{BKZ}^{\mathrm{zk}}_{\text{M-LWE}}$	Required BKZ blocksize for M-LWE, Lem. A.1	435
$\varepsilon^{\mathrm{sound}}_{\text{M-SIS}}$	Hardness bound for M-SIS, Lem. A.1	$2^{-134.2}$
$\varepsilon^{\mathrm{zk}}_{\text{M-LWE}}$	Hardness bound for M-LWE, Lem. A.1	$2^{-143.6}$
$\varepsilon_{snd,1}$	Soundness error	$2^{-128.7}$
$\varepsilon_{zk,1}$	Zero-Knowledge security bound	$2^{-143.6}$
	Efficiency Estimates	
$\lvert\pi_1\rvert$	Proof size	20.73 KB

Table B.3. Suggested parameter set for the issuance proof (Prove_1).

Symbol	Description	Value
	Proof System Parameters	
λ	Security parameter	128
n_3	Proof system ring degree	64
$k_{1,3}$	Subring embedding dimension ($\theta_{1,3}$)	32
$k_{2,3}$	Subring embedding dimension ($\theta_{2,3}$)	4
d_2	Module rank	23
$q_{\min}$	Smallest modulus factor	506773
q_{π_2}	Proof system modulus (qp)	28112440614324601
ℓ	Soundness amplification dimension	4
m_1	Witness dimension	265
m_2	Dimension of ABDLOP randomness	80
χ	Distribution of ABDLOP randomness	$\mathcal{B}_1$
ρ	Infinity norm of challenges	8
$\widetilde{\eta}$	Manhattan-like norm of challenges	93
$(\alpha_1, \alpha_2, \alpha_3)$	Rejection sampling slacks	$(3.01, 3.01, 3.01)$
(M_1, M_2, M_3)	Rejection sampling repetition rates	$(\sqrt{2}, \sqrt{2}, \sqrt{2})$
σ_1	First rejection sampling width	50172282.021
σ_2	Second rejection sampling width	20035.267
σ_3	Third rejection sampling width	9903663.065
γ	Mask commitment compression parameter	629250462
D	Witness commitment compression parameter	21
	Security Estimates	
$\mathrm{BKZ}^{\mathrm{sound}}_{\text{M-SIS}}$	Required BKZ blocksize for M-SIS, Lem. A.2	388
$\mathrm{BKZ}^{\mathrm{zk}}_{\text{M-LWE}}$	Required BKZ blocksize for M-LWE, Lem. A.2	537
$\varepsilon^{\mathrm{sound}}_{\text{M-SIS}}$	Hardness bound for M-SIS, Lem. A.2	$2^{-129.8}$
$\varepsilon^{\mathrm{zk}}_{\text{M-LWE}}$	Hardness bound for M-LWE, Lem. A.2	$2^{-173.4}$
$\varepsilon_{snd,2}$	Soundness error	$2^{-128.2}$
$\varepsilon_{zk,2}$	Zero-Knowledge security bound	$2^{-173.5}$
	Efficiency Estimates	
$\lvert\pi_2\rvert$	Proof size	73.46 KB

Table B.4. Suggested parameter set for the signing proof (Prove_2).