Cryptanalysis of Lattice-Based Sequentiality Assumptions and Proofs of Sequential Work

Chris Peikert* Yi Tang†

December 6, 2023

Abstract

This note describes a total break of the sequentiality assumption (and broad generalizations thereof) underlying the candidate lattice-based proof of sequential work (PoSW) recently proposed by Lai and Malavolta at CRYPTO 2023. Specifically, for sequentiality parameter T and SIS parameters $n, q, m = n \log q$, the attack computes a solution of norm $(m+1)^{\log_k T}$ (or norm $O(\sqrt{m})^{\log_k T}$ with high probability) in depth $\tilde{O}_{n,q}(k \log_k T)$, where the integer $k \leq T$ may be freely chosen. (The $\tilde{O}$ notation hides polylogarithmic factors in the variables appearing in its subscript.)

In particular, with the typical parameterization $\log q = \tilde{O}_{n,T}(1)$, for $k = 2$ the attack finds a solution of quasipolynomial norm $O(\sqrt{m})^{\log T}$ in only *polylogarithmic* $\tilde{O}_{n,T}(1)$ depth; this strongly falsifies the assumption that finding such a solution requires depth *linear* in T. Alternatively, setting $k = T^{\epsilon}$, the attack finds a solution of polynomial norm $O(\sqrt{m})^{1/\epsilon}$ in depth $\tilde{O}_{n,T}(T^{\epsilon})$, for any constant $\epsilon > 0$.

We stress that the attack breaks the *assumption* underlying the proposed PoSW, but not the *PoSW itself* as originally defined. However, the attack does break a *slight modification* of the original PoSW, which has an essentially identical security proof (under the same kind of falsified assumption). This suggests that whatever security the original PoSW may have is fragile, and further motivates the search for a PoSW based on a sound lattice-based assumption.

1 Sequentiality Assumption

We recall the lattice-based candidate sequentiality assumption recently proposed by Lai and Malavolta [LM23] (with some slight differences in the notation). This assumption was used as the foundation for a candidate *proof of sequential work* (PoSW); see Section 3 for further details.

Let q be a positive integer modulus and $\mathbf{g} \in \mathbb{Z}_q^{\ell}$ be a suitable "gadget" vector; for concreteness, we use the standard powers-of-two gadget $\mathbf{g} = (1, 2, 4, \ldots, 2^{\ell-1})^t$ for $\ell = \lceil \log_2 q \rceil$, but all of our results easily adapt to other choices of gadgets (see [MP12] for further details). Let $\mathbf{g}^{-1} \colon \mathbb{Z}_q \to \mathbb{Z}^{\ell}$ be the corresponding "(bit) decomposition" function: $\mathbf{g}^{-1}(u)$ is binary and hence "short," and $\langle \mathbf{g}, \mathbf{g}^{-1}(u) \rangle = \mathbf{g}^t \cdot \mathbf{g}^{-1}(u) = u$ for any $u \in \mathbb{Z}_q$. Finally, extend the gadget and its decomposition operation to work on matrices (including vectors) as follows: for any positive integer n, let $\mathbf{G}_n := \mathbf{I}_n \otimes \mathbf{g}^t \in \mathbb{Z}_q^{n \times n\ell}$ denote the block-wise application of $\mathbf{g}^t$ to each ℓ-dimensional column block, and let $\mathbf{G}_n^{-1}(\cdot)$ denote the entry-wise application of $\mathbf{g}^{-1}$ on any matrix $\mathbf{U}$ having n rows, so that $\mathbf{G}_n \cdot \mathbf{G}_n^{-1}(\mathbf{U}) = \mathbf{U}$.

*University of Michigan, cpeikert@umich.edu.

†University of Michigan, yit@umich.edu.

For dimensions n and $m = n\ell$, matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, vector $\mathbf{u}_0 \in \mathbb{Z}_q^n$, and sequentiality parameter T, Lai and Malavolta [LM23] consider the following linear system, where $\mathbf{G} = \mathbf{G}_n \in \mathbb{Z}_q^{n \times m}$:[1]

$$\underbrace{\begin{pmatrix} \mathbf{G} & & & & \\ \mathbf{A} & \mathbf{G} & & & \\ & \mathbf{A} & \ddots & & \\ & & \ddots & \mathbf{G} & \\ & & & \mathbf{A} & \mathbf{G} \end{pmatrix}}_{\mathbf{A}_T \in \mathbb{Z}_q^{Tn \times Tm}} \cdot \underbrace{\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_T \end{pmatrix}}_{\mathbf{x} \in \mathbb{Z}^{Tm}} = \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \in \mathbb{Z}_q^{Tn} \ . \tag{1.1}$$

In [LM23] it is assumed that for sufficiently large q, given *random* $(\mathbf{A}, \mathbf{u}_0)$, computing a "somewhat short" solution $\mathbf{x} \in \mathbb{Z}^{Tm}$ to Equation (1.1)—specifically, one having Euclidean norm $\|\mathbf{x}\| \leq O(n)^{2 \log T}$—requires $\Omega(T)$ sequential time.[2] Notice that a "short" *binary* solution, which has Euclidean norm $\|\mathbf{x}\| \leq \sqrt{Tm}$, can be computed in depth proportional to T, as $\mathbf{x}_1 = \mathbf{G}_n^{-1}(\mathbf{u}_0) \in \mathbb{Z}^m$ and then $\mathbf{x}_i = -\mathbf{G}_n^{-1}(\mathbf{A}\mathbf{x}_{i-1}) \in \mathbb{Z}^m$ for $i = 2, \ldots, T$. This works because

$$\mathbf{A}\mathbf{x}_{i-1} + \mathbf{G}\mathbf{x}_i = \mathbf{A}\mathbf{x}_{i-1} - \mathbf{A}\mathbf{x}_{i-1} = \mathbf{0} \ .$$

The reason for the gap between the notion of "short" obtained by the above computation, versus "somewhat short" in the assumption, is the $O(n)^{2 \log T}$ "slack factor" in the proof of sequential work from [LM23]. More specifically, the honest prover can use a "short" solution to convince the verifier, but the knowledge extractor can extract only a "somewhat short" solution from any (possibly malicious) prover that manages to convince the verifier. The attack described below in Section 2 crucially exploits this gap, using a low-depth computation to find a solution that is significantly longer than the "short" one, but still below the "somewhat short" threshold.

2 Attack

Here we describe an attack on the above-described sequentiality assumption from [LM23]. The attack applies to a broad generalization of the system in Equation (1.1), namely, any one in which the matrix is block lower-triangular and has "gadget" matrices $\mathbf{G}$ (or any other matrices having known trapdoors) as the diagonal blocks.

Trapdoors. We first recall from [MP12] the notion of a (gadget) *trapdoor* for a matrix $\mathbf{A} \in \mathbb{Z}_q^{N \times W}$, for any dimensions N, W. This is any "short" matrix $\mathbf{R} \in \mathbb{Z}^{W \times M}$, where $M = N\ell$, for which

$$\mathbf{A}\mathbf{R} = \mathbf{G}_N = \mathbf{I}_N \otimes \mathbf{g}^t \in \mathbb{Z}_q^{N \times M} \ .$$

More precisely, $\mathbf{R}$ should have suitably bounded spectral norm $\sigma_1(\mathbf{R}) := \max_{\mathbf{w} \neq \mathbf{0}} \|\mathbf{R}\mathbf{w}\| / \|\mathbf{w}\|$.

[1] For convenience, we have made a slight but immaterial tweak to the system appearing in [LM23], by appending the $\mathbf{G}$ matrix in the bottom-right block, appending the component $\mathbf{x}_T$ to the solution vector, and requiring the bottom-most block on the right-hand side to be zero, not chosen by the adversary. It is easy to see that the two systems are equivalent. In addition, [LM23] considers a more compact and "algebraically structured" version of the system over a certain polynomial ring, where each n-by-n block of $\mathbf{A}$ is the (structured) multiplication matrix of a random ring element. Our attack works for arbitrary $\mathbf{A}$, so for simplicity and generality we adopt the above presentation.

[2] In [LM23] the assumption is actually stated in terms of the ℓ_∞ norm instead of the Euclidean norm. Our attack is most naturally analyzed in the Euclidean norm, but it obtains the same bounds in the ℓ_∞ norm because the latter is upper bounded by the former.

Using such a trapdoor, it is easy to compute, in low depth, a comparably short solution $\mathbf{X} \in \mathbb{Z}^{W \times K}$ to $\mathbf{AX} = \mathbf{U}$, for any $\mathbf{U} \in \mathbb{Z}_q^{N \times K}$: simply let $\mathbf{X} = \mathbf{R} \cdot \mathbf{G}_N^{-1}(\mathbf{U})$. This works because both $\mathbf{R}$ and $\mathbf{G}_N^{-1}(\mathbf{U})$ are short, and

$$\mathbf{AX} = \mathbf{AR} \cdot \mathbf{G}_N^{-1}(\mathbf{U}) = \mathbf{G}_N \cdot \mathbf{G}_N^{-1}(\mathbf{U}) = \mathbf{U} \ .$$

Recall that $\mathbf{G}_N^{-1}$ applies $\mathbf{g}^{-1}$ to each entry of $\mathbf{U}$ independently, so $\mathbf{X}$ can be computed in depth $\tilde{O}_{N,q}(1)$, i.e., polylogarithmic in N and q.

More generally, for some relations (including the specific one from [LM23]) it suffices, and will yield better bounds, to have a "partial" trapdoor with respect to the top n rows, for some $n \leq N$. This is a short matrix $\mathbf{R} \in \mathbb{Z}^{W \times m}$, where $m = n\ell$, for which

$$\mathbf{AR} = \begin{pmatrix} \mathbf{I}_n \\ \mathbf{0} \end{pmatrix} \otimes \mathbf{g}^t = \begin{pmatrix} \mathbf{G}_n \\ \mathbf{0} \end{pmatrix} \in \mathbb{Z}_q^{N \times m} \ .$$

Using such a trapdoor it is easy to compute, in depth only $\tilde{O}_{n,q}(1)$, a short solution $\mathbf{X} = \mathbf{R} \cdot \mathbf{G}_n^{-1}(\mathbf{U}) \in \mathbb{Z}^{W \times K}$ to $\mathbf{AX} = \left(\begin{smallmatrix} \mathbf{U} \\ \mathbf{0} \end{smallmatrix}\right) \in \mathbb{Z}^{N \times K}$ for any $\mathbf{U} \in \mathbb{Z}_q^{n \times K}$. (Of course, this technique naturally generalizes to any particular set of rows, or other choices of subspaces.)

2.1 General Attack

The basic idea of the attack is to *recursively* compute, in fairly low depth, a trapdoor for the block-triangular matrix of the linear system in question. The trapdoor can then be used to find a particular solution (as described above) for any desired right-hand side.

The base case of the recursion is where the system's matrix is $\mathbf{G}_n$ for some (typically small) n, or is some other matrix having a known trapdoor. For the recursive case, suppose that the system's matrix $\mathbf{A}$ has $N = N_0 + N_1$ rows and block lower-triangular form

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_0 & \\ \mathbf{W} & \mathbf{A}_1 \end{pmatrix} \ , \tag{2.1}$$

where $\mathbf{A}_0$ and $\mathbf{A}_1$ respectively have N_0 and N_1 rows; typically, one would take $N_0 \approx N_1$. Note that the matrix $\mathbf{A}_T$ from the system in Equation (1.1) has this form, where $\mathbf{A}_0 = \mathbf{A}_1 = \mathbf{A}_{T/2}$ (for even T, and similarly for odd T) and $\mathbf{W}$ is all zeros except in its upper-rightmost n-by-m block.

Suppose that trapdoors $\mathbf{R}_0, \mathbf{R}_1$ are known—typically, via *parallel* recursive calls—for $\mathbf{A}_0, \mathbf{A}_1$, respectively. Then letting $M_b = N_b\ell$ for $b \in \{0, 1\}$, a trapdoor $\mathbf{R}$ for $\mathbf{A}$ is

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_0 & \\ & \mathbf{R}_1 \end{pmatrix} \begin{pmatrix} \mathbf{I}_{M_0} & \\ \mathbf{R}' & \mathbf{I}_{M_1} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_0 & \\ \mathbf{R}_1\mathbf{R}' & \mathbf{R}_1 \end{pmatrix} \text{ where } \mathbf{R}' = -\mathbf{G}_{N_1}^{-1}(\mathbf{W}\mathbf{R}_0) \in \mathbb{Z}^{M_1 \times M_0} \ . \tag{2.2}$$

This is because

$$\mathbf{AR} = \begin{pmatrix} \mathbf{A}_0\mathbf{R}_0 & \\ \mathbf{W}\mathbf{R}_0 + \mathbf{A}_1\mathbf{R}_1\mathbf{R}' & \mathbf{A}_1\mathbf{R}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{G}_{N_0} & \\ \mathbf{W}\mathbf{R}_0 - \mathbf{W}\mathbf{R}_0 & \mathbf{G}_{N_1} \end{pmatrix} = \mathbf{G}_N \ .$$

Note that $\mathbf{R}$ can be computed (given $\mathbf{R}_0, \mathbf{R}_1, \mathbf{W}$) in depth $\tilde{O}_{N,q}(1)$, and that

$$\sigma_1(\mathbf{R}) \leq \max\{\sigma_1(\mathbf{R}_0), \sigma_1(\mathbf{R}_1)\} \cdot (1 + \sigma_1(\mathbf{R}')) \ ,$$

i.e., the spectral norm of the trapdoor $\mathbf{R}$ for $\mathbf{A}$ is essentially a $\sigma_1(\mathbf{R}')$ factor larger than for those of $\mathbf{A}_0, \mathbf{A}_1$. Because $\mathbf{R}' \in \mathbb{Z}^{M_1 \times M_0}$ has binary entries, $\sigma_1(\mathbf{R}') \leq M$ where $M = N\ell$. Or, using a randomized, subgaussian variant of $\mathbf{g}^{-1}$, we get that $\sigma_1(\mathbf{R}') = O(\sqrt{M})$ with high probability.

Larger arity. More generally, suppose that we can split $\mathbf{A}$ into $k \geq 2$ row and column blocks, as

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_0 & & & \\ \mathbf{W}_{1,0} & \mathbf{A}_1 & & \\ \vdots & \vdots & \ddots & \\ \mathbf{W}_{k-1,0} & \mathbf{W}_{k-1,1} & \cdots & \mathbf{A}_{k-1} \end{pmatrix}, \tag{2.3}$$

where $\mathbf{A}_i$ has N_i rows. Also suppose that a trapdoor $\mathbf{R}_i$ for $\mathbf{A}_i$ is known (typically, via recursion). Then it can be verified that a trapdoor for $\mathbf{A}$ is

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_0 & & & \\ & \mathbf{R}_1 & & \\ & & \ddots & \\ & & & \mathbf{R}_{k-1} \end{pmatrix} \begin{pmatrix} \mathbf{I}_{M_0} & & & \\ \mathbf{R}'_{1,0} & \mathbf{I}_{M_1} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{R}'_{k-1,0} & \mathbf{R}'_{k-1,1} & \cdots & \mathbf{I}_{M_{k-1}} \end{pmatrix}$$

where

$$\mathbf{R}'_{i,j} = -\mathbf{G}_{N_i}^{-1}\Big(\mathbf{W}_{i,j}\mathbf{R}_j + \sum_{j<j'<i} \mathbf{W}_{i,j'}\mathbf{R}_{j'}\mathbf{R}'_{j',j}\Big) .$$

These blocks can be computed in $k-1$ sequential stages, in parallel over all $j = 0, \ldots, k-1$, by computing $\mathbf{R}'_{i,j}$ for $i = j+1, \ldots, k-1$. So, $\mathbf{R}$ can be computed in depth $\tilde{O}_{N,q}(k)$. And similarly to above, the spectral norm of $\mathbf{R}$ is bounded by $\max_i \sigma_1(\mathbf{R}_i) \cdot (1+M)$, and the $1+M$ factor can be replaced by $O(\sqrt{M})$ with high probability using a randomized $\mathbf{g}^{-1}$.

2.2 Optimization with Partial Trapdoors

For the specific relation from [LM23] (see Equation (1.1)), it suffices to compute a *partial* trapdoor with respect to just the top n rows (regardless of the total number of rows N), which yields better bounds on the spectral norms and computation depths. This can be done by a simple optimization of the above approach.

Suppose that $\mathbf{A}$ is as in Equation (2.1), that only the top n rows of $\mathbf{W}$, denoted $\mathbf{W}_n$, are possibly nonzero (which is indeed the case in Equation (1.1)), and that *partial* trapdoors $\mathbf{R}_0, \mathbf{R}_1$ with respect to the top n rows of $\mathbf{A}_0, \mathbf{A}_1$ (respectively) are known. Then letting $m = n\ell$, a partial trapdoor with respect to the top n rows of $\mathbf{A}$ is

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_0 & \\ & \mathbf{R}_1 \end{pmatrix} \begin{pmatrix} \mathbf{I}_m \\ \mathbf{R}' \end{pmatrix} = \begin{pmatrix} \mathbf{R}_0 \\ \mathbf{R}_1\mathbf{R}' \end{pmatrix} \text{ where } \mathbf{R}' = -\mathbf{G}_n^{-1}(\mathbf{W}_n\mathbf{R}_0) \in \mathbb{Z}^{m \times m} . \tag{2.4}$$

This follows from a similar calculation as the one above, and the fact that

$$\mathbf{A}_1\mathbf{R}_1 \cdot \mathbf{G}_n^{-1}(\mathbf{W}_n\mathbf{R}_0) = \begin{pmatrix} \mathbf{G}_n \\ \mathbf{0} \end{pmatrix} \cdot \mathbf{G}_n^{-1}(\mathbf{W}_n\mathbf{R}_0) = \begin{pmatrix} \mathbf{W}_n\mathbf{R}_0 \\ \mathbf{0} \end{pmatrix} = \mathbf{W}\mathbf{R}_0 ,$$

where the last equality holds because only the top n rows of $\mathbf{W}$ are possibly nonzero.

Similarly to above, $\mathbf{R}$ can be computed in depth $\tilde{O}_{n,q}(1)$; note that here N is replaced by n, because $\mathbf{R}_1$ has only $m = n\ell$ (not $M = N\ell$) columns. And just as above, $\sigma_1(\mathbf{R}) \leq \max\{\sigma_1(\mathbf{R}_0), \sigma_1(\mathbf{R}_1)\} \cdot (1 + \sigma_1(\mathbf{R}'))$. However, because $\mathbf{R}' \in \mathbb{Z}^{m \times m}$—unlike above, its dimension does not grow with the size of $\mathbf{A}$ or the level of the recursion—a better bound $\sigma_1(\mathbf{R}') \leq m$ holds, and $\sigma_1(\mathbf{R}') = O(\sqrt{m})$ holds with high probability when using a randomized $\mathbf{g}^{-1}$. Finally, a similar optimization also works when splitting $\mathbf{A}$ into $k \geq 2$ row and column blocks, as detailed above.

2.3 Analysis of the Full Attack

Suppose that the matrix $\mathbf{A}$ has the required block lower-triangular form, with $\mathbf{G}_n$ blocks along the diagonal and a total of T row blocks, hence $N = Tn$ rows.

- By recursively applying the approach from Section 2.1 with arity $k = 2$ and $N_0 = \lfloor N/2 \rfloor$, a computation of total depth $\tilde{O}_{N,q}(\log T) = \tilde{O}_{n,q,T}(1)$ yields a trapdoor having spectral norm bounded by $(M+1)^{\lceil \log T \rceil}$ where $M = N\ell$, or $O(\sqrt{M})^{\lceil \log T \rceil}$ with high probability using a randomized $\mathbf{g}^{-1}$.
- In particular, the system from Equation (1.1) has the required form, and to find a short solution it suffices to have only a *partial* trapdoor with respect to the top n rows. So, using the optimized approach from Section 2.2, a computation of total depth $\tilde{O}_{n,q}(\log T) = \tilde{O}_{n,q,T}(1)$ yields such a trapdoor with spectral norm $(m+1)^{\lceil \log T \rceil}$ where $m = n\ell$, or $O(\sqrt{m})^{\lceil \log T \rceil}$ with high probability.
- Using arity $k \in [2, T]$, a computation of total depth $\tilde{O}_{N,q}(k \log_k T)$ yields a trapdoor of spectral norm $(M+1)^{\lceil \log_k T \rceil}$ in general, and for the particular system in Equation (1.1), a computation of total depth $\tilde{O}_{n,q}(k \log_k T)$ yields a trapdoor of spectral norm $(m+1)^{\lceil \log_k T \rceil}$. For example, by setting $k = T^{\varepsilon}$ for an arbitrarily small constant $\varepsilon > 0$, in depth $\tilde{O}_{n,q}(T^{\varepsilon})$ we get a trapdoor of polynomially bounded spectral norm $(m+1)^{\lceil 1/\varepsilon \rceil}$ for the system in Equation (1.1).

Finally, in the typical instantiation where $\log q = \tilde{O}_{n,T}(1)$, the q subscripts in the above $\tilde{O}$ expressions can be replaced by (or absorbed into the preexisting) n, T.

For $\log q = o(n)$ and hence $m = o(n^2)$, the above falsifies (even a major weakening of) the assumption made in [LM23], which posits that computing a solution of norm $O(n)^{2 \log T}$ requires depth $\Omega(T)$ (or in weaker form, depth $\Omega(T^{\varepsilon})$ for some constant $\varepsilon > 0$).

3 Is the Proof of Sequential Work Insecure?

Lai and Malavolta [LM23] also gave a candidate *proof of sequential work* (PoSW) protocol, and proved its security based on the sequentiality assumption stated in Section 1. While the attack from Section 2 *falsifies this assumption*, and thus renders the security proof vacuous, it does not immediately follow that the *PoSW itself* is broken.

So far, our attempts to attack the PoSW have led to a curious state of affairs. On the one hand, we have not been able to break the PoSW as originally defined in [LM23]. However, our techniques do break a *slight modification* of the original PoSW, which has an essentially identical security proof under the same kind of (false) sequentiality assumption. This suggests that whatever security the original PoSW may have is fragile, and not due to any intentional design choice or technique in the security proof. Additional ideas might lead to a successful attack against the original PoSW; alternatively, it might actually be secure, perhaps with a different security proof under some other plausible assumption. We leave these topics for future work.

In Section 3.1 below, we summarize the main challenges we encountered in our attempt to attack the original PoSW. Then in Section 3.2 we describe a slight variant of the PoSW, note why it has an essentially identical security proof, and explain how the attack from Section 2 breaks it.

3.1 Challenge in Attacking the PoSW

Structure of the PoSW. In the PoSW, the prover computes a short solution $\mathbf{x}$ to Equation (1.1), then engages in an interactive public-coin protocol to convince the verifier that it knows such a solution. (The

protocol can be made non-interactive in the usual way via the Fiat–Shamir transform.) To do this, it uses a small random challenge from the verifier to linearly "fold" the first and second halves of $\mathbf{x}$ together, which yields a somewhat longer solution of half the dimension, then recursively proves knowledge of this folded solution.

More precisely, consider a "truncated" version of Equation (1.1) without the final component $\mathbf{x}_T$ or the rightmost column block of the matrix, so that the final component $-\mathbf{u}_{T-1}$ on the right-hand side is typically nonzero. The prover first announces this $\mathbf{u}_{T-1}$ (or equivalently, $\mathbf{x}_T$) as its claimed result of the sequential computation. Then it gives a proof of knowledge of a solution to the truncated system: it announces $\mathbf{x}_{T/2}$ (assume that T is even for simplicity), which the verifier checks is short enough, and the verifier announces a small random challenge r. Observe that the remaining halves of $\mathbf{x}$ form two solutions to reduced-dimension instances of the truncated Equation (1.1), with known right-hand sides of the appropriate form. So, the prover linearly combines these solutions using r, and the resulting (somewhat longer) solution is proved recursively in the same manner, until the dimension is small enough to simply reveal and check the solution. Note that with each successive stage of the recursion, the verifier must apply a more relaxed norm check (by some multiplicative factor) on the prover's announced short value, because folding increases the solution norm.

The difficulty. The key issue in attacking the PoSW seems to be as follows: (1) the prover must first announce some $\mathbf{u}_{T-1}$ (or $\mathbf{x}_T$) to the verifier; (2) the prover can know *at most one* short solution for whatever value it announces; and (3) the only way we see to convince the verifier is by knowing a solution that is *nearly as short* as what the honest prover would compute. We elaborate on each of these points next.

The announced value of $\mathbf{u}_{T-1}$ represents the claimed final result of the computation of depth $\approx T$ that the prover supposedly performed to get a solution to (truncated) Equation (1.1). It is a straightforward exercise to show that computing *distinct* short solutions, for *any* fixed and possibly adversarially chosen right-hand side, is at least as hard as solving the SIS problem for the random matrix $\mathbf{A}$. So, under the standard assumption that SIS is intractable (see, e.g., [Ajt96, MR04]), once an efficient prover (of any depth) reveals some $\mathbf{u}_{T-1}$, it can know at most one short solution. The same goes for the later stages of the recursive protocol with lower-dimensional instances of (truncated) Equation (1.1), where the first and last components of the right-hand side are determined by the previous stages.[3]

From this point, the only way we see to convince the verifier is by proceeding exactly as the honest (specified) prover would, using a single known solution. Because of how the solution norm expands under "folding," and in particular because the verifier's challenge is multiplied by the *second* half of the solution, it appears that convincing the verifier requires *starting from a solution that is nearly as short—at least in its second half—as what the honest prover would compute*. We do not see a way to generate such a short solution in depth significantly less than T. Again we stress that the attack from Section 2 achieves low depth by exploiting the moderately large slack factor; in particular, the solution is merely "somewhat short" in its second half (and final quarter, etc.), and need not be as short as what the honest prover computes.

3.2 Breaking a Modified PoSW

The above discussion motivates a natural alternative folding operation in the PoSW: multiply the verifier's random challenge r by the *first*, rather than the *second*, half of the solution. That is, instead of folding a solution $\mathbf{x}$ into the lower-dimensional one $\mathbf{x}' = \mathbf{x}_{\text{first}} + r \cdot \mathbf{x}_{\text{last}}$, use $\mathbf{x}' = r \cdot \mathbf{x}_{\text{first}} + \mathbf{x}_{\text{last}}$. The security proof from [LM23] (modified in the obvious way) holds equally well for this option, because the two halves are

[3] This state of affairs is quite different from the context of the attack from Section 2, where the adversary is not bound to any particular right-hand side of truncated Equation (1.1), and knows short solutions to many different right-hand sides via its trapdoor.

treated symmetrically. (As far as we can tell, in [LM23] the choice of folding operation between these two options was arbitrary.)

Interestingly, with this trivial modification—along with a mild relaxation of the norm checks in the verifier—*the PoSW can be broken* by the attack from Section 2. Essentially, this is because the $\mathbf{x}_{\text{last}}$ constructed by the attack is a small factor longer than $\mathbf{x}_{\text{first}}$, so the two summands $r \cdot \mathbf{x}_{\text{first}}, \mathbf{x}_{\text{last}}$ in the modified folding operation are more "balanced," and their sum passes the verifier's (appropriately relaxed) norm checks. To obtain a complete working attack, we also need to ensure that the revealed $\mathbf{x}_{T/2}$ components (at every stage of the recursion) are sufficiently short. This is easy to achieve by an appropriate instantiation of the attack, which makes the final block of the (partial) trapdoor very short at each stage of its recursive construction.

The details. We first instantiate the attack from Section 2, including the optimization from Section 2.2, so that it recursively constructs trapdoors whose bottom-most blocks of $m = n\ell$ rows are *binary*. This instantiation applies to any matrix $\mathbf{A}$ that can be expressed as in Equation (2.3), with lower-rightmost block $\mathbf{A}_{k-1} = \mathbf{G}_n$ (so $N_{k-1} = n$). This is indeed the case for the system from [LM23] as given in Equation (1.1), and for any other block lower-triangular system with $\mathbf{G}_n$ for its diagonal blocks. Observe that $\mathbf{R}_{k-1} = \mathbf{I}_m$ is a trapdoor for $\mathbf{A}_{k-1}$, so the bottom-most m rows of the constructed trapdoor $\mathbf{R}$ for $\mathbf{A}$ have the form $\mathbf{R}_{k-1}\mathbf{G}^{-1}(\star) = \mathbf{G}^{-1}(\star)$, which is binary as desired. Recursively using, say, $k = 3$ with $N_0 = N_1 + 1 = N/2$ yields the same asymptotic norm and depth bounds given in Section 2.3.

With the above instantiation, observe that the constructed trapdoors are binary not only in their own bottom-most row blocks, but also in those of their *top halves*, their *top quarters*, and so on. This holds by induction, because the top half of each trapdoor equals the previously constructed trapdoor (possibly augmented with an additional zero block).

For a (partial) trapdoor $\mathbf{R}$ recursively constructed as above from trapdoors $\mathbf{R}_0, \mathbf{R}_1$, consider the following proof of knowledge of such a trapdoor, which closely follows and generalizes the above-described PoSW. (All of the following applies equally well for the PoSW itself, which is merely for a specific right-hand side of Equation (1.1).) The prover first announces and removes the (binary) Tth row block. The proving stage then works as follows:

- The prover announces and removes the $(T/2)$th row block, which the verifier checks is short enough. (Note by the above observation that in the first iteration this block is binary, and hence satisfies the verifier's norm check.)
- The verifier announces a small random challenge r.
- The prover applies the alternative folding operation to the halves of the remaining trapdoor, and recursively proves knowledge of the result. Recall that the verifier must use a larger norm bound (by some suitable multiplicative factor) with each step of the recursion.

In any iteration, the folded trapdoor (which is the input to the recursive call) has the form $\overline{\mathbf{R}}' = \overline{\mathbf{R}}_0 \cdot (r\mathbf{I}) + \overline{\mathbf{R}}_1 \cdot \mathbf{G}^{-1}(\star)$, where $\overline{\mathbf{R}}_0, \overline{\mathbf{R}}_1$ are respectively the top and bottom halves of input (with its middle block removed). Its middle block, which will be announced in the next recursive call and needs to be sufficiently short, has a moderately larger norm than those of the middle blocks of $\overline{\mathbf{R}}_i$, by a factor given by r and $\mathbf{G}^{-1}(\star)$. By an inductive argument, these middle blocks have suitably bounded norms, growing exponentially with the depth of recursion, hence so does their folding. So, as long as the verifier's norm bounds grow accordingly throughout the recursion, all of its norm checks will be satisfied, and the verifier will ultimately accept.

References

[Ajt96] M. Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996. Page 6.

[LM23] R. W. F. Lai and G. Malavolta. Lattice-based timed cryptography. In *CRYPTO*, pages 782–804. 2023. Pages 1, 2, 3, 4, 5, 6, and 7.

[MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718. 2012. Pages 1 and 2.

[MR04] D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. Preliminary version in FOCS 2004. Page 6.