From: pqc-forum@list.nist.gov on behalf of Wessel van Woerden <wesselvanwoerden@gmail.com>
Sent: Tuesday, July 25, 2023 12:32 PM
To: pqc-forum
Cc: Eamonn Postlethwaite
Subject: [pqc-forum] Round 1 (Additional Signatures) OFFICIAL COMMENT: EHTv3

Dear EHTv3 Team,

In the proposal it is written that
"Hence, to forge a signature for EHTv3 NIST category 1 parameters it may be enough to solve an approximate SVP in the Euclidean norm for the lattice LI with an approximation factor $\alpha \leq 3$. Since the approximation factor is small, that might be as hard as finding a shortest non-zero vector in LI".

The approximation factor 3 is in practice rather large.
Let's have a look at the security level 1 parameters of EHTv3, where this lattice has dimension l=451.
BKZ with a blocksize of around 205 is enough to find an approximate shortest vector of the required length, which then with reasonable probability has a small enough infinity norm (according to your arguments around Table 4).
Roughly, this leads to only 0.292*205=59.86 bits of security in the core-SVP model.
We recommend to first randomise the lattice basis for LI as the q-vectors might interfere with the usual conversion to the infinity norm, see e.g. Appendix C of [4].

To forge one actually has to solve an approximate CVP problem, this can be approached e.g. in the following ways:
1. Embed the target into a lattice of dimension l+1.
With proper randomisation and due to the q-ary structure any approximate short vector can be expected to give a solution to the approximate CVP instance with probability about 2/q. One may sieve to find many such.
2. b-BKZ reduce the lattice, then incrementally lift the target t by solving CVP in blocks [l-b:l), [l-2*b+1:l-b+1),.. following [1]. This gives about the same (Hermite)-CVP approximation factor as BKZ does for (Hermite)-SVP.

In short, a signature can be forged with an SVP/CVP oracle in dimension 205, which puts the scheme far below security level 1.
If we do not project to the 451 dimensional sublattice as in the specification, but do the same attack in the full 460 dimensions, while allowing m-l coefficients to be arbitrarily large, then we expect the successful blocksize to be even lower.

Secondly, we present a potentially better attack based on the fact that the scheme uses a low modulus q, e.g. q=47 for the security level 1 parameters.

This gives the basis profile a particular Z-shape.
Following the ideas of [2] we can first reduce the lattice such that nq of the initial vectors are q-vectors.
Then we first find many approx CVP solutions in the projected lattice p_nq(L) via sieving/batch-CVPP, and lift them to the full context by solving exact CVP on q*ZZ^nq (by simple rounding).
The probability that one such lift has enough small coefficients is rather small, but this can be compensated by the many solutions given by the sieving/batch-CVPP.
We created a script [5] that shows that an equivalent of blocksize b=170 is enough for this attack to break the security level 1 parameters of EHTv3.
All steps have a cost of the order 2^(0.292b+o(b)). This script uses some functions [3] from [2].
For security levels 3 and 5, b=300 and b=400 respectively are enough for this attack.

Best regards,

Eamonn Postlethwaite and Wessel van Woerden

[1] https://msp.org/obs/2020/4-1/p16.xhtml
[2] https://eprint.iacr.org/2023/1125.pdf
[3] https://github.com/verdiverdiverdi/ISIS-small-q
[4] https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf
[5] https://github.com/WvanWoerden/EHT/blob/main/EHT.sage
--

From: Adam Suhl <asuhl@ucsd.edu>
Sent: Saturday, July 29, 2023 12:31 AM
To: pqc-comments
Cc: pqc-forum; kryan@ucsd.edu
Subject: Round 1 (Additional Signatures) OFFICIAL COMMENT: EHT

Dear EHTv3 Team, and dear community,

We have developed and implemented a practical signature forgery attack on the category 1 security level parameters of EHTv3. Our proof-of-concept attack code is available at https://github.com/ucsd-hacc/ehtv3_cryptanalysis. Our attack uses 500000 signatures to learn an equivalent private key, and it takes approximately 122 core-hours. The underlying flaw is due to the biased distribution of signatures, and our partial key recovery algorithm exploits the special structure of the public key to find a matching private key.

EHTv3 is a signature scheme based on q-ary lattices. The private key (C, T, B) and public key (A) are rectangular matrices defined over Z_q such that A = CTB^{-1}. Furthermore, C is sparse, T is triangular, and B is invertible. Signatures are generated by first hashing to a vector h and then finding a random vector a satisfying Ca = h. Importantly, the triangular structure of T allows one to efficiently solve the Closest Vector Problem (Theorem 1 in the EHTv3 specification), so the signer finds vectors y and z such that a = Ty + z and the max norm of z is small. Finally, the signer computes signature x = By. Verification involves computing e = h - Ax = h - CTB^{-1}By = h - C(a - z) = Cz. Since C is sparse and z is small, e will be small for a correctly generated signature.

This signing method induces a secret-dependent distribution on the publicly observable values of e. We apply standard techniques to recover columns of C from this distribution, then we use a novel process to recover a valid signing key from A and the columns of C.

Recovering columns of C
Observe that an attacker who observes many signatures can compute e_i = h_i - Ax_i = Cz_i for each sample. Both C and z_i have bounded norm, so the entries of e_i are small relative to the modulus, and usually no modular reduction occurs at this step. (Our attack will discard any e_i with any entry large enough that modular reduction may have taken place.) The values of e_i then lie in (a projection of) the parallelepiped generated by the columns of C. We use standard tools to learn this distribution and infer the columns of C.

As the EHTv3 submission mentions, if C were square this would be an instance of the Hidden Parallelepiped Problem: the e_i would lie in the parallelepiped generated by the columns of C, and we want to "learn the parallelepiped" by studying the distribution of observed Cz -- but C has more columns than rows. Instead, this is an instance of the Hidden Zonotope Problem (HZP) from [DucasNguyen12]. Ducas and Nguyen present a statistical algorithm for solving HZP that involves approximating the covariance matrix of C from observed samples and learning columns of C from the zonotope using gradient descent. We implemented the algorithm of [DucasNguyen12] and confirm that it succeeds for the category 1 parameters.

[DucasNguyen12]: Leo Ducas and Phong Nguyen, "Learning a Zonotope and More: Cryptanalysis of NTRUSign Countermeasures," Asiacrypt 2012.
https://www.iacr.org/archive/asiacrypt2012/76580428/76580428.pdf

Concretely, in the category 1 security level parameters, all operations are modulo 47. C is a matrix of size 460x484 and is specially constructed so that all rows have l_1 norm 9. The vector z contains entries that lie between -3 and 3. Entries in e_i before modular reduction can be as large as +-27; discarding the roughly 10% signatures where e_i has entries large enough that modular reduction might have occurred, we ensure that each remaining e_i = C z_i over the integers. This rejection sampling, along with the way the z_i are generated in the first place, mean that the z_i are not exactly uniformly distributed, but empirically the [DucasNguyen12] algorithm succeeds anyway.

This recovers some information about the private matrix C, but it is as yet unclear how this can be used to forge signatures. In addition, the algorithm for HZP may return column vectors in any order, only recovers these vectors up to sign, and the set of recovered vectors may contain false positives. We demonstrate how this information is enough to forge EHTv3 signatures.

Recovering a private key from columns of C We assume, for the purposes of explanation, that the algorithm of [DucasNguyen12] has successfully recovered the unordered set of column vectors of C up to multiplication of each vector by 1 or -1. We wish to recover values C, T, and B that correspond to the public key that will enable us to forge signatures for this public key. Note that we do not need to recover exactly the same C, T, and B originally used to generate the public key; we only need C to be sparse, T to allow efficient CVP, and invertible B.

We use the triangular structure of T to progressively recover the correct ordering and sign of columns in C. T is a matrix of dimension kn x n, or 484 x 242. For example, if n=3, then T has the lower triangular form

[1 0 0]
[7 0 0]
[* 1 0]
[* 7 0]
[* * 1]
[* * 7]

where each * is replaced with a randomly generated value modulo 47.

Observe what happens when we compute the final column of CT. The final column of T is known, and all but the last two entries of this column are zero, so only the last two columns of C have any influence on the last column of CT. Because of the equality CT = AB, we also know that the last column (and all other columns) of CT lies in the column span of A. A is dimension 460 x 242, so a randomly chosen vector of dimension 460 is unlikely to be in this column span. Since we have a set of unordered columns of C up to sign, we can try all pairs and signs for the last two columns of C until we find one where the last column of CT is in the column span of A. Experimentally, this solution is almost always unique. Although the 484 * 483 * 2 * 2 possible pairs could be found with brute force search, we use a meet-in-the-middle attack to make this step faster.

This technique cannot be directly used to recover the order of the other columns of C. The second-to-last column of CT depends on the last four columns of C and the second-to-last column of T. There are now even more unknowns that go into this product: the choice of the third and fourth rightmost column of C, the choice of sign for these columns, and the choice of the two unknown subdiagonal values in the second-to-last column of T. We could try all 482 * 481 * 2 * 2 * 47 * 47 possibilities and check if the resulting column is in the column span of A, but the search quickly becomes prohibitively expensive as we have to guess more subdiagonal values of T.

We solve this problem by using kernels of recovered columns of C. If K is the left kernel of the rightmost two columns of C (which we recovered in the previous step), then the last two column vectors of KC are all zero. The second-to-last column of KCT therefore depends on the choice of the next two columns of C and the choice of sign, but it no longer depends on the unknown subdiagonal because they are multiplied by the zeros in KC. We search the 482 * 481 * 2 * 2 possibilities for the correct choice of C columns and signs that leads to the second-to-last column of KCT being in the column span of KA. The subdiagonal values can be recovered efficiently in a similar way, since the uncertainty in the choice of columns has been eliminated. This process repeats, recovering the order and sign of columns of C, the values of T, and allowing us to solve for the columns of B.

Our attack results in C, T, and B satisfying the key equation for EHTv3. For ease of explanation, we are leaving out some additional details like how we reduce the search space further by considering equivalent forms of different private keys for a single public key, or how we address the effect of the rank of K on the column span of KA. We will post these additional details of our attack to ePrint.

Experiments

We have implemented our full attack and have published our code at https://github.com/ucsd-hacc/ehtv3_cryptanalysis. We ran the experiments on a cluster of machines with Xeon E5-2699 v4 processors (88 cores per machine at 2.20 GHz) and the peak RAM usage was under 16 GB per machine. We began by generating one of the KAT keys with the reference implementation, and we then generated 500000 signatures using the crypto_sign function from the reference implementation. For each of these signatures, we computed the error vectors e_i = h_i - Ax_i = Cz_i, which took 15 core-minutes. We ran the algorithm from [DucasNguyen12] on this instance of HZP. It took ~20 seconds to estimate the covariance matrix, and it took ~120 core-hours to perform gradient descent to recover 519 candidate columns of C up to sign. Using these candidate columns, we performed our partial key recovery attack to obtain a private key (C, T, B). This step took around 80 core-minutes. Finally, we used the recovered key to forge a signature in 38 core-seconds. This signature passes verification as implemented in crypto_sign_open.

The attack in total took around 122 core-hours, or using the units of [Beullens22], about one laptop-weekend.

[Beullens22]: Ward Beullens, "Breaking Rainbow Takes a Weekend on a Laptop," Crypto 2022.
https://crypto.iacr.org/2022/papers/538804_1_En_16_Chapter_OnlinePDF.pdf

With additional computing time (and possibly additional signatures) we expect this attack will scale to the category 3 and category 5 parameters, although further implementation work is needed.
Section 10.1 of the specification notes that instances of EHTv4 may be transformed into instances of EHTv3, which suggests this attack would likely apply to EHTv4 as well.

Sincerely,
Keegan Ryan and Adam Suhl

From: pqc-forum@list.nist.gov on behalf of Martin Feussner <feussnermartin@gmail.com>
Sent: Monday, August 28, 2023 9:34 AM
To: pqc-forum
Cc: Wessel van Woerden; Eamonn Postlethwaite
Subject: [pqc-forum] Re: Round 1 (Additional Signatures) OFFICIAL COMMENT: EHTv3

Dear Wessel van Woerden and Eamonn Postlethwaite.

Thank you for the comment on the security of EHTv3. You have raised an interesting question on how large the SVP approximation factor may be such that the hardness of the problem is comparable to the hardness of the SVP itself in practice. Unfortunately, neither detailed analysis, nor detailed algorithm and nor computational evidence were presented in your comment. As we were not able to verify your assertions, we suggest a challenge as a response to the comment, you can find it here. That is an EHTv3 instance with smaller parameters than in the specification for the security level 1. It is based on a q-ary (for q=47) lattice of rank 280 instead of rank 460 as in the specification. We estimate the security of the challenge at around 80 bits. If your observation is correct, then you would be able to break the challenge. Besides the solution, we kindly ask you to provide your code in order to replicate the attack.

Best regards,
EHT team.

On Tuesday, 25 July 2023 at 18:31:53 UTC+2 Wessel van Woerden wrote:

> Dear EHTv3 Team,
>
> In the proposal it is written that
> "Hence, to forge a signature for EHTv3 NIST category 1 parameters it may be enough to solve an approximate SVP in the Euclidean norm for the lattice LI with an approximation factor $\alpha \leq 3$. Since the approximation factor is small, that might be as hard as finding a shortest non-zero vector in LI".
>
> The approximation factor 3 is in practice rather large.
> Let's have a look at the security level 1 parameters of EHTv3, where this lattice has dimension l=451.
> BKZ with a blocksize of around 205 is enough to find an approximate shortest vector of the required length, which then with reasonable probability has a small enough infinity norm (according to your arguments around Table 4).
> Roughly, this leads to only 0.292*205=59.86 bits of security in the core-SVP model.
> We recommend to first randomise the lattice basis for LI as the q-vectors might interfere with the usual conversion to the infinity norm, see e.g. Appendix C of [4].
>
> To forge one actually has to solve an approximate CVP problem, this can be approached e.g.

From: pqc-forum@list.nist.gov on behalf of Martin Feussner <feussnermartin@gmail.com>
Sent: Tuesday, August 29, 2023 9:50 AM
To: pqc-forum
Cc: Martin Feussner; Wessel van Woerden; Eamonn Postlethwaite
Subject: [pqc-forum] Re: Round 1 (Additional Signatures) OFFICIAL COMMENT: EHTv3

Dear all

There seems to be some sharing issues with the link (requires you to request permission to access). You can also try this link. If you're still prompted to request access, please do so and I will approve it immediately.

Best regards,
EHT team.

On Monday, 28 August 2023 at 15:34:20 UTC+2 Martin Feussner wrote:

> Dear Wessel van Woerden and Eamonn Postlethwaite.
>
> Thank you for the comment on the security of EHTv3. You have raised an interesting question on how large the SVP approximation factor may be such that the hardness of the problem is comparable to the hardness of the SVP itself in practice. Unfortunately, neither detailed analysis, nor detailed algorithm and nor computational evidence were presented in your comment. As we were not able to verify your assertions, we suggest a challenge as a response to the comment, you can find it here. That is an EHTv3 instance with smaller parameters than in the specification for the security level 1. It is based on a q−ary (for q=47) lattice of rank 280 instead of rank 460 as in the specification. We estimate the security of the challenge at around 80 bits. If your observation is correct, then you would be able to break the challenge. Besides the solution, we kindly ask you to provide your code in order to replicate the attack.
>
> Best regards,
> EHT team.
>
> On Tuesday, 25 July 2023 at 18:31:53 UTC+2 Wessel van Woerden wrote:
>
> > Dear EHTv3 Team,
> >
> > In the proposal it is written that
> > "Hence, to forge a signature for EHTv3 NIST category 1 parameters it may be enough to solve an approximate SVP in the Euclidean norm for the lattice LI with an approximation factor $\alpha \leq 3$. Since the approximation factor is small, that might be as hard as finding a shortest non−zero vector in LI".
> >
> > The approximation factor 3 is in practice rather large.

From: pqc-forum@list.nist.gov on behalf of Wessel van Woerden <wesselvanwoerden@gmail.com>
Sent: Wednesday, August 30, 2023 7:52 AM
To: pqc-forum
Cc: Martin Feussner; Wessel van Woerden; Eamonn Postlethwaite
Subject: [pqc-forum] Re: Round 1 (Additional Signatures) OFFICIAL COMMENT: EHTv3

Dear EHT Team,

The details of the algorithm and the analysis was presented in the linked script with extensive comments [1], so I do not agree that none such details were given.

Anyway, I never mind a challenge so below a solution obtained via a method similar to the first attack we presented.
In short, to solve it:
1. Create a basis of the q-ary lattice L = A*Zˆn + q*Zˆm starting with m-n q-vectors.
2. BKZ-reduce it with blocksize 100
3. Find a close vector to h using Babai reduction. Randomize with the last few basis vectors to obtain multiple close vectors and pick one that satisfies the constraints.

For step 2 I used [2] with pump_and_jump+gpu and jump=3, but with a bit more patience plain G6K should also work.
In terms of quality it seems comparable to BKZ with blocksize +-90.
The total runtime was less than 3 hours on 8 threads and 1 partial a100 gpu.
For step 3 a simple sage script is available at [3], which runs in a few seconds. The output is copied below.

```
# output of break.sage using the BKZ-100 reduced basis B100.txt:
Loading data..
Starting babai reduction..
Solution found with minl =   273   >= 1
x_vec =   (23, 22, 46, 33, 1, 7, 3, 42, 29, 17, 18, 12, 14, 15, 12, 30, 18, 27, 11, 27,
26, 18, 17, 36, 6, 38, 0, 9, 13, 31, 23, 23, 36, 26, 16, 3, 42, 0, 39, 21, 10, 16, 46, 23,
5, 9, 37, 34, 0, 19, 37, 40, 42, 31, 24, 3, 27, 31, 9, 23, 35, 29, 3, 23, 30, 19, 8, 6,
14, 11, 44, 29, 31, 17, 36, 37, 22, 32, 10, 43, 42, 25, 13, 19, 9, 22, 31, 31, 6, 31, 8,
27, 25, 42, 0, 11, 42, 8, 9, 3, 36, 2, 11, 37, 43, 21, 8, 29, 27, 46, 26, 34, 5, 37, 15,
17, 16, 21, 20, 6, 6, 43, 28, 10, 42, 14, 20, 7, 39, 40, 12, 10, 9, 41, 9, 4, 29, 17, 41,
4, 1, 38, 21, 30, 27, 8, 20, 35, 21, 21, 41, 7)
Verify solution..
Solution verified, e= (1, -8, -2, 6, 4, -12, 3, -5, 0, 3, 0, -1, 2, 6, -2, -2, 8, -7, 1,
1, 0, 11, -2, -7, -8, -7, 9, 4, 1, -4, -3, 0, 3, 4, -2, -4, -1, 0, -2, 3, -3, 3, 2, 8, -3,
-7, -1, 0, 1, -3, -5, 4, -3, -1, 3, -3, -4, -3, -7, 0, -6, 1, -5, 4, -5, 0, -6, 0, -10, -
3, 9, -3, -2, -10, 0, 0, -1, 0, -2, 1, -9, -8, 7, 0, 0, -2, 7, 0, 3, 0, 6, 0, -2, 4, 1, -
1, 4, -2, 0, -7, 0, 7, -6, 6, -1, -3, 0, 3, 0, 0, 0, 0, 0, 4, -2, 0, 0, -9, -1, 0, 0, 0,
```

0, 0, 0, 1, 0, -6, 0, 0, 0, 0, 0, 8, 6, 0, 2, 0, 0, 0, 6, -1, 0, 0, 0, 0, -3, 0, 10, 0, -17, 0, 16, 6, 9, 5, 1, -4, -18, 1, 6, -3, 17, 10, -3, -3, -2, 3, 9, -2, 3, 2, -23, 7, -1, 3, 13, 0, 10, 22, -2, 1, -2, 7, 9, -2, 14, -1, -2, 1, 4, 1, -9, -5, 0, 12, -3, 3, 4, -2, -4, -1, 0, -7, -1, 5, -7, -3, -4, -1, 1, 0, 1, 1, -1, -5, 3, -3, -6, 2, -9, -5, -6, -4, 1, -4, 4, 0, 4, 7, -9, -6, -3, -2, -3, -1, 1, 3, 5, 4, 2, 6, -4, 1, -3, 6, -3, 0, 2, -1, -4, -3, 3, -3, 0, 3, 2, 5, -3, 1, 1, -3, 8, 9, 2, 1, -2, -3, -1, -4, 0, -7, -1, -1, 0, -2, -1, 0, -6, -1)

[1] https://github.com/WvanWoerden/EHT/blob/main/EHT.sage
[2] https://github.com/WvanWoerden/G6K-GPU-Tensor
[3] https://github.com/WvanWoerden/EHT/blob/main/break/break.sage

Best regards,

Wessel van Woerden
IMB, Université de Bordeaux

On Tuesday, 29 August 2023 at 15:50:11 UTC+2 Martin Feussner wrote:

> Dear all
>
> There seems to be some sharing issues with the link (requires you to request permission to access). You can also try this link. If you're still prompted to request access, please do so and I will approve it immediately.
>
> Best regards,
> EHT team.
>
> On Monday, 28 August 2023 at 15:34:20 UTC+2 Martin Feussner wrote:
>
> > Dear Wessel van Woerden and Eamonn Postlethwaite.
> >
> > Thank you for the comment on the security of EHTv3. You have raised an interesting question on how large the SVP approximation factor may be such that the hardness of the problem is comparable to the hardness of the SVP itself in practice. Unfortunately, neither detailed analysis, nor detailed algorithm and nor computational evidence were presented in your comment. As we were not able to verify your assertions, we suggest a challenge as a response to the comment, you can find it here. That is an EHTv3 instance with smaller parameters than in the specification for the security level 1. It is based on a q-ary (for q=47) lattice of rank 280 instead of rank 460 as in the specification. We estimate the security of the challenge at around 80 bits. If your observation is correct, then you would be able to break the challenge. Besides the solution, we kindly ask you to provide your code in order to replicate the attack.
> >
> > Best regards,
> > EHT team.
> >
> >
> > On Tuesday, 25 July 2023 at 18:31:53 UTC+2 Wessel van Woerden wrote:

From: Martin Feussner <feussnermartin@gmail.com>
Sent: Sunday, November 26, 2023 8:45 AM
To: pqc-forum
Cc: Adam Suhl; pqc-forum; kr...@ucsd.edu; pqc-comments
Subject: Re: Round 1 (Additional Signatures) OFFICIAL COMMENT: EHT

Dear Keegan and Adam,

We have responded to this attack here.

Best regards,
Martin Feussner and Igor Semaev

On Saturday, 29 July 2023 at 06:30:51 UTC+2 Adam Suhl wrote:

> Dear EHTv3 Team, and dear community,
>
> We have developed and implemented a practical signature forgery attack on the category 1 security level parameters of EHTv3. Our proof-of-concept attack code is available at https://github.com/ucsd-hacc/ehtv3_cryptanalysis. Our attack uses 500000 signatures to learn an equivalent private key, and it takes approximately 122 core-hours. The underlying flaw is due to the biased distribution of signatures, and our partial key recovery algorithm exploits the special structure of the public key to find a matching private key.
>
> EHTv3 is a signature scheme based on q-ary lattices. The private key (C, T, B) and public key (A) are rectangular matrices defined over Z_q such that A = CTB^{-1}. Furthermore, C is sparse, T is triangular, and B is invertible. Signatures are generated by first hashing to a vector h and then finding a random vector a satisfying Ca = h. Importantly, the triangular structure of T allows one to efficiently solve the Closest Vector Problem (Theorem 1 in the EHTv3 specification), so the signer finds vectors y and z such that a = Ty + z and the max norm of z is small. Finally, the signer computes signature x = By. Verification involves computing e = h - Ax = h - CTB^{-1}By = h - C(a - z) = Cz. Since C is sparse and z is small, e will be small for a correctly generated signature.
>
> This signing method induces a secret-dependent distribution on the publicly observable values of e. We apply standard techniques to recover columns of C from this distribution, then we use a novel process to recover a valid signing key from A and the columns of C.
>
> # Recovering columns of C
> Observe that an attacker who observes many signatures can compute e_i = h_i - Ax_i = Cz_i for each sample. Both C and z_i have bounded norm, so the entries of e_i are small relative to the modulus, and usually no modular reduction occurs at this step. (Our attack will discard any e_i with any entry large enough that modular reduction may have taken place.) The values of e_i then lie in (a projection of) the parallelepiped generated by the columns of C. We use standard tools to learn this distribution and infer the columns of C.
>
> As the EHTv3 submission mentions, if C were square this would be an instance of the Hidden Parallelepiped Problem: the e_i would lie in the parallelepiped generated by the columns of C, and we want to "learn the parallelepiped" by studying the distribution of observed Cz -- but C has more columns than rows. Instead, this is an instance of the Hidden Zonotope Problem (HZP) from [DucasNguyen12]. Ducas and Nguyen present a statistical algorithm for solving HZP that involves approximating the covariance matrix of C from observed samples and learning columns of C from the zonotope using gradient descent. We implemented the algorithm of [DucasNguyen12] and confirm that it succeeds for the category 1 parameters.

From: pqc-forum@list.nist.gov on behalf of Martin Feussner <feussnermartin@gmail.com>
Sent: Sunday, November 26, 2023 8:47 AM
To: pqc-forum
Cc: Wessel van Woerden; Martin Feussner; Eamonn Postlethwaite
Subject: [pqc-forum] Re: Round 1 (Additional Signatures) OFFICIAL COMMENT: EHTv3

Dear Eamonn and Wessel,

We have responded to this attack here.

Best regards,
Martin Feussner and Igor Semaev