A lean BIKE KEM design for ephemeral key agreement

Nir Drucker[1], Shay Gueron[2,3], and Dusan Kostic[4]

[1] IBM Research, Israel, [2]University of Haifa, Israel,
[3]Meta, USA [4]Amazon, USA

Abstract. The QC-MDPC code-based KEM BIKE is an alternative candidate for standardization for the NIST Post-Quantum Cryptography Standardization Project. Per NIST's report [2] "The BIKE cryptosystem was initially designed for ephemeral key use but has now been claimed to also support static key use". BIKE uses the BGF decoder of [9] where its Decoding Failure Rate (DFR) is estimated by means of an extrapolation method. While this methodology provides a solid indication for a very small DFR, which is required for an IND-CCA claim, it may still be considered short of a proven upper bound, as stated in [2], "... and an upper bound on the decoding failure rate has yet to be found". Nevertheless, the IND-CPA security of BIKE is established without a small DFR requirement on the decoder, and this property suffices for protocols that use ephemeral keys. This is the case for protocols that maintain the modern notion of forward secrecy (hence avoid static keys), where a prominent example is TLS 1.3.
This paper examines the communication bandwidth and the performance of a BIKE design that targets only the ephemeral key use cases, i.e., settles with IND-CPA security. We call this design "Lean-BIKE". This study illustrates the incremental cost of the IND-CCA property. We argue that it would be useful to standardize two configurations of BIKE: a) "Lean-BIKE" that enjoys the reduced cost of an IND-CPA KEM, for the major class of forward secrecy supporting usages; b) BIKE whose IND-CCA security could be established by either a finer proof methodology for the BGF decoder or with another decoder that has a proven DFR upper bound.

Keywords: BIKE, Post-Quantum Cryptography, NIST, QC-MDPC codes, IND-CPA, IND-CCA, Ephemeral keys, Engineering DFR.

1 Introduction

NIST post-quantum cryptography standardization project is in its Round 4 final stages, where four Key Encapsulation Mechanisms (KEMs) are evaluated for standardization, as alternatives. Bit Flipping Key Encapsulation (BIKE) is one of them. It is a Quasi-Cyclic Moderate-Density Parity-Check (QC-MDPC) code-based KEM, where its latest version is defined in [3].

Per BIKE specification [3], BIKE is a CPA KEM secure under the two standard assumptions on the hardness of: a) Quasi-Cyclic (QC) Syndrome Decoding $QCSD_{r,t}$; b) QC Codeword Finding $QCCF_{r,w}$ problem. For CCA security (see [12] for details) a third assumption is needed: the correctness of the decoder which is being used. This translates to a sufficiently low DFR.

1.1 The state of the BIKE

BIKE went through several improvement cycles from Round 1 to Round 4 during the standardization project. The different specification document versions of BIKE are posted at https://bikesuite.org/spec.html.

The initial BIKE design (Round 1) defined three BIKE algorithms named BIKE-1/BIKE-2/BIKE-3 and targeted only IND-CPA security. As the NIST report [2] states "The BIKE cryptosystem was initially designed for ephemeral key use but has now been claimed to also support static key use",

IND-CCA security was first claimed with Round 2 version (v3.0). The statement was based on an extrapolation method [16]: simulations on small-dimension versions of the cryptosystem are used for collecting statistical information on the decoding failures, and the resulting curve is extrapolated to dimensions that are estimated to provide the target DFR (2^{-128} for Level 1 security). This seemed like a promising direction for claiming IND-CCA security for BIKE. However, the paper [8], that built a secure decoder that operates in constant time, already pointed out the existence of weak keys that slipped under the simulations' radar, and that different extrapolation rules could be used and give different DFR estimations. Accordingly, subsequent BIKE versions updated the parameters. Following work tried to characterize larger classes of weak keys. As stated in the NIST report [2]:

> "The BIKE specification now claims IND-CCA security, citing additional analysis to support their claim [17, 18]. Iterative, bit-flipping decoders are not characterized by a bounded decoding radius; thus, there is an expected nonzero probability of decoding failure"

Recently, the work of [19] showed that for BIKE Level-1 the average decryption failure rate can be upper bounded only by $2^{-116.61}$ and not by 2^{-128} as required for IND-CCA security. This enabled a key recovery attack on BIKE with a complexity of $2^{116.61}$ when ciphertexts cannot be reused, and worse - with $2^{98.77}$ complexity when ciphertexts reusing is allowed, i.e., in a multi-target setting. Interestingly, binding BIKE ciphertexts to the public keys to avoid multi-target attacks for BIKE was proposed and performance-profiled in [10] already in 2021.

BIKE specification v4.0 converged from three options to a single KEM version (called "BIKE") by leveraging the fast (constant-time) polynomial inversion algorithm and implementation [7] that improved the key generation of (the previously called) BIKE-2. This reduced the communication bandwidth of BIKE to a more competitive level and reduced the implementation complexity of the BIKE package.

BIKE specification v5.0 proposed a "biased" sampling method intended to address some timing attacks on the IND-CCA security. As pointed out in [6] and [11], was unnecessary (the mitigation was already in place in the constant time implementation [5]) and led to an implementation bug. Version 5.0 was fixed shortly after its submission and before a public release on NIST's site making v5.1 the current version of BIKE.

Our proposal. NIST found BIKE's IND-CCA claim to be encouraging and selected it as an alternative for standardization to allow BIKE's developers to extend their study and perhaps overcome the problem of establishing an upper bound for the decoder's DFR. It is unclear (at least to us) how BIKE's assumption on the BGF decoder's DFR can be proved and establish the IND-CCA security claims. For this reason, we go back to BIKE basics and consider defining an efficient BIKE construction that targets only IND-CPA security, and can be used (only) in cases where one-time (ephemeral) keys are required. To this end, we identify the "extra" steps and parameter choices that BIKE uses for the sake of achieving IND-CCA security, remove or modify them, and measure the resulting performance. Note that an IND-CPA version of BIKE is discussed in our previous study [4], but in this paper, we account for all the recent advances in BIKE such as the Black-Gray-Flip (BGF) decoder [9] and the fast polynomial inversion [7], where both techniques require specific parameter choices.

We examine the communication bandwidth and the performance of a BIKE design that we call "Lean-BIKE", that targets only the ephemeral keys use cases. We show an optimized Lean-BIKE design which is more efficient than the (presumably) IND-CCA secure version. We suggest that this design could be a useful companion for BIKE, for usages and protocols such as TLS 1.3 that intend to preserve forward secrecy.

2 BIKE in a nutshell

Preliminaries and notation. Let $\mathbb{F}_2$ be the finite field of characteristic 2. Let $\mathcal{R}$ be the polynomial ring $\mathbb{F}_2[X]/\langle X^r - 1\rangle$, for some positive r. The Hamming weight of every element $v \in \mathcal{R}$ is denoted by $wt(v)$. We denote protocol failures by $\perp$. Uniform random sampling from a set W is denoted by $w \xleftarrow{\$} W$ and sampling from a distribution $\mathcal{D}$ over W is denoted by $w \xleftarrow{\mathcal{D}} W$.

Fixed-weight sampling. Let len, w be positive integers with $len > w$, and let $\mathcal{S}_{len,w}$ denote the set of $\binom{len}{w}$ subsets of $\{0, 1, \ldots, len - 1\}$ with cardinality w. Let $\mathcal{D}$ be a probability distribution over $\mathcal{S}_{len,w}$. An algorithm that samples an element from $\mathcal{S}_{len,w}$ according to a distribution $\mathcal{D}$, i.e., executing $\xleftarrow{\mathcal{D}} \mathcal{S}_{len,w}$, is called a fixed-weight sampling algorithm. The notation $\xleftarrow{\mathcal{D}} \mathcal{S}_{len,w}(m)$ refers to pseudorandom fixed-weight sampling starting from a seed m,

$(sk, \sigma, h) \leftarrow \text{KeyGen}()$

1. $\sigma \xleftarrow{\$} \{0,1\}^{256}$
2. $h_0, h_1 \xleftarrow{\mathcal{D}_1} (\mathcal{S}_{r,d})^2$
3. $sk = (h0, h1, \sigma)$
4. $pk = h = h_1 h_0^{-1}$
5. Return (sk, pk)

$(C, K) \leftarrow \text{Encaps}(h)$

1. $m \xleftarrow{\$} \{0,1\}^{256}$
2. $e = (e_0, e_1) \xleftarrow{\mathcal{D}_2} \mathcal{S}_{2r,t}(m||h)$
3. $C = (c_0, c_1) = (e_0 + e_1 h, m \oplus \mathbf{L}(e_0, e_1))$
4. $K = \mathbf{K}(m, C)$
5. Return (C, K)

$m = \text{Decaps}(sk, \sigma, h, C)$

1. $e' = (e'_0, e'_1) = \text{Decode}(sk, C)$
2. $m' = c_1 \oplus \mathbf{L}(e')$
3. If $\mathcal{S}_{n,t}(m'||h) \neq e'$ then $m' = \sigma$.
4. Return $\mathbf{K}(m', C)$

Fig. 1. A schematic description of the KEM BIKE in its Fourth Round BIKE specification v5.1 [3]. Here, $\mathcal{D}_1$ and $\mathcal{D}_2$ are two different "biased" (nonuniform) distributions [6, 15]. The red colored elements are those that are not required for IND-CPA security and that we can remove for the Lean-BIKE. The concatenation of h, marked in blue, that is not part of BIKE v5.1, is the recommendation made in [10] in order to bind the public key to the errors vector e, which was proposed in order to thwart multi-key (multi-target). It may be a useful protective measure to be added in the IND-CCA setting. The communication bandwidth of the KEM depends on the value of r.

BIKE. Defined with parameters r, t, w, d (where $d = w/2$), the KEM BIKE (as in [3]) is outlined in Figure 1. The functions $\mathbf{K}$ and $\mathbf{L}$ are treated as random oracles, each having a specific instantiation. The Decode function takes a ciphertext (syndrome $s \in \mathcal{R}$) and a secret key (parity-check polynomials $(h_0, h_1) \in (\mathcal{S}_{r,d})^2$) as input, and produces a sparse vector $(e_0, e_1) \in \mathcal{S}_{2r,t}$ as output.

3 The cost of IND-CCA design

To achieve the IND-CCA property BIKE uses the implicit-rejection version of Fujisaki-Okamoto (FO) transformation ($FO^{\not\perp}$, as described in [14]). Figure 1

marks these steps in red. The key generation needs to use an extra 256 bits of randomness (σ) and also store them as part of the secret key. The encapsulation extends the ciphertext with additional 256 bits of data ($m \oplus \mathbf{L}(e_0, e_1)$) and requires computing the value $\mathbf{L}(e_0, e_1)$. Lastly, the decapsulation is no longer allowed to return $\perp$ on decoding failure, while also needing to perform the re-encryption step to verify that the encapsulator followed the encapsulation process.

The implementation of IND-CCA introduces higher costs due to the requirement that the re-encryption and equality check $\mathcal{S}_{n,t}(m') \neq e'$ implementation must run in constant time, as failing to do so would leave the system vulnerable to the attack outlined in [13]. This additional step was incorporated into the IND-CCA version of BIKE for re-encryption purposes.

One might argue that, even when only considering IND-CPA security for BIKE, it could be valuable, from a security standpoint, to maintain the $FO^{\not\perp}$ transformation and, specifically, the aforementioned step. Here, the question is whether a constant-time implementation is still necessary. Even with timing leakage, the leaked information cannot be used for the same (ephemeral) key again. Nevertheless, we argue that there is still a potential risk of multi-target attacks because the timing depends solely on m (and not on the secret-public key pair). To address this concern, a possible mitigation strategy (except for a constant-time implementation) is to follow our recommendation in [10], which suggests binding the errors vector to the public key during encapsulation (see the blue h in Figure 1).

Finally, we note that using the $FO^{\not\perp}$ transformation is not mandatory for IND-CPA BIKE.

Engineering DFR notion and targets for IND-CPA designs. IND-CPA security, manifested by using ephemeral keys, has much lower requirements on the KEM design, and specifically for BIKE. Here, decoding failures need *not* be hidden from a passive observer of the ciphertext nor from an adversary that submits a "poisoned" ciphertext and checks for a decoding failure. Obviously, no information is gleaned from a decoding failure when the private key is used only once. In this scenario, the DFR of the decoder is not a security feature. Thus, we replace the requirement for a (low) upper bound on the DFR with a notion that we call "engineering DFR":

> **Engineering DFR:** design (parameterize) a decoder such that its DFR conforms with the tolerance level of failure in the system.

In real systems, the tolerance level is much more lenient than the 2^{-128} DFR required (for L1) to support IND-CCA security. As an illustration, consider the well known target "five nines" (99.999%) gold standard of system availability. This would translate to a DFR of no more than $2^{\log_2(10^{-5})} = 2^{-16.61}$ (similarly, for 6 nines (99.9999%) and 7 nines (99.99999%) reliability, we would need to mandate a DFR of $2^{-19.93}$ and $2^{-23.25}$, respectively). Network errors in server-client communication occur at higher rates.

Table 1. Latency in processor cycles of BIKE's keygen + decapsulation with different decoder configurations (using a different number of iterations) that meet a different Engineering DFR levels. The boldfaced row is the current BIKE (v5.1) that targets IND-CCA security.

Level	r	#iter	DFR	AES or	AVX2		AVX512	
			(log_{10})	SHA3	BIKE [3]	-FO$^{\not\perp}$	BIKE [3]	-FO$^{\not\perp}$
L1	10,499	3	-5	A	930,055	876,080	563,777	523,922
L1	10,499	3	-5	S	963,061	893,378	568,439	513,432
L1	10,627	3	-6	A	911,090	857,115	577,264	537,409
L1	10,627	3	-6	S	937,813	868,130	593,661	538,654
L1	10,499	4	-7	A	1,050,742	996,767	625,755	585,900
L1	10,499	4	-7	S	1,062,933	993,250	634,734	579,727
L1	**12,323**	5		S	**1,434,345**		**796,453**	
L3	20,233	3	-5	A	2,898,945	2,773,710	1,581,796	1,495,419
L3	20,233	3	-5	S	2,972,337	2,816,474	1,607,429	1,492,798
L3	20,107	4	-6	A	3,340,856	3,215,621	1,751,045	1,664,668
L3	20,107	4	-6	S	3,294,385	3,138,522	1,776,840	1,662,209
L3	20,261	4	-7	A	3,305,129	3,179,894	1,768,474	1,682,097
L3	20,261	4	-7	S	3,331,911	3,176,048	1,801,795	1,687,164
L3	**24,659**	5		S	**4,471,475**		**2,116,853**	

The Lean-BIKE configuration. Our proposed Lean-BIKE design is a configuration of BIKE where we:

1. Remove the $FO^{\not\perp}$ transformation.
2. Select the parameters for the BGF decoder to target Engineering DFR.
3. Instantiate $\mathbf{L}, \mathbf{K}$ functions with AES-CTR PRNG and SHA-2.
4. Use uniform random sampling.

We refer the reader to the illustration in Figure 1. We also mention some advantages of (re-)using AES for the PRNG, in addition to higher performance compared to SHA3: a real usage would use the shared key established through the KEM in order to encrypt/authenticate some payload, and this encryption is likely to use AES. Furthermore, a multi-block hash could be used for enhanced performance, as described in [4].

4 Results: Lean-BIKE performance

To evaluate Lean-BIKE performance, we compare it to the implementation of the current BIKE specification [3]. We explore several variants of Lean-BIKE to illustrate the relative impact of the different components of the KEM.

For each variant, we measured a *constant-time* implementation (available in [5]) and ran it on an Intel AVX2 and an Intel AVX512 architecture. For that, we used an Intel(R) Xeon(R) Platinum 8488C CPU @ 2.40GHz processor.

For the Lean-BIKE configuration, we decided to select three levels of engineering DFR for testing targeting reliability of five, six, and seven 9's (99.999%,

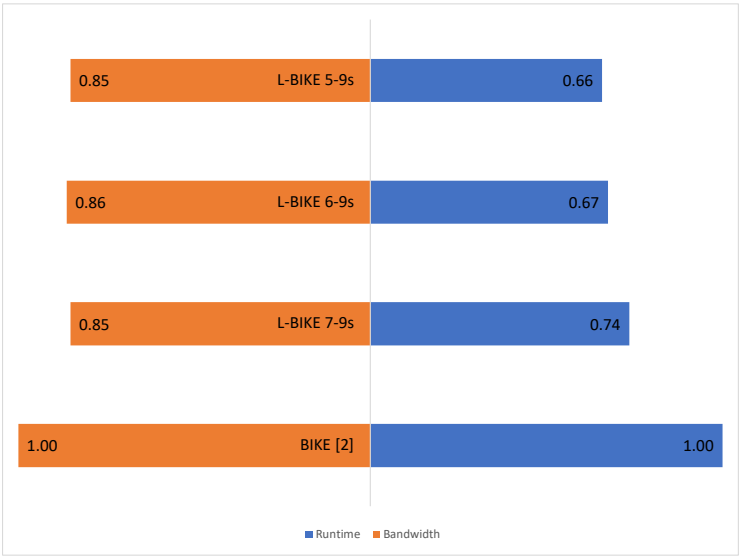

Fig. 2. Runtime and bandwidth requirement of Lean-BIKE L1 relative to BIKE [3]. Three versions of Lean-BIKE are given, achieving reliability of five, six, and seven 9's.

99.9999%, and 99.99999%). For each DFR target we chose a set of $(r, \#iter)$ pairs (where $\#iter$ denotes the number of iterations for the BGF decoder) that achieved the DFR, measured the performance, and selected the best $(r, \#iter)$ pair. The results are summarized in Table 4. Figures 2 and 3 show the *relative* performance results of the different configurations, for BIKE L1 and BIKE L3, respectively. In both figures, the reference point (corresponding to "1") is the implementation of BIKE as specified in [3].

Defending against multi-target attacks is an especially useful property for IND-CCA KEMs. Adding binding of the public key to the message in BIKE to address multi-target attacks, as proposed in [10], introduces additional latency in BIKE's encapsulation. For example, this would make BIKE's IND-CCA version encapsulation about 20% slower (consequently, making the Lean-BIKE variants relatively even more efficient).

5 Conclusion

Design flexibility. An interesting property of an IND-CPA version of BIKE is that some design choices can be made, unilaterally, by the communicating parties, without interoperability consequences. For example, the decapsulating party can choose different decoders that meet its engineering DFR and performance targets and possibly other design metrics (e.g., hardware cost or software simplicity). Similarly, when no re-encryption step is involved, the encapsulating party can employ different PRNGs for computing the errors vector. Furthermore,

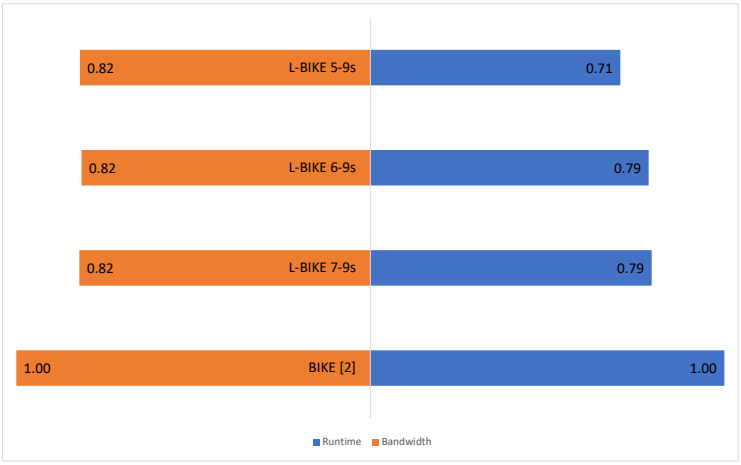

Fig. 3. Runtime and bandwidth requirement of Lean-BIKE L3 relative to BIKE [3]. Three versions of Lean-BIKE are given, achieving reliability of five, six, and seven 9's.

the FO$^{\not\perp}$ transformation has more effect on the decapsulating party than on the encapsulating party, due to the re-encryption step. Consequently, one option for device designers who consider implementing only the encapsulation steps in their devices is to always use the IND-CCA version of BIKE. This will allow them to be interoperable with both IND-CPA and IND-CCA servers, leaving the responsibility of choosing the security guarantees to the server's configuration. Of course, the device's implementation should be flexible enough to support different r values.

IND-CCA, IND-CPA, and forward secrecy. NIST has selected "Kyber" for a KEM standardization which is currently undergoing. It is also approaching a decision on standardizing an alternative KEM that is not lattice-based, and where BIKE is one of the candidates. It is therefore important to consider some merits, shortcomings, and options for BIKE. In this context, we outlined the difficulty of claiming a proven IND-CCA property for BIKE. As shown in [12] if BIKE is instantiated with a decoder that has a DFR of 2^{-128} (for Level 1 security) it would be IND-CCA secure. However, the current methodology for estimating the DFR of a QC-MDPC decoder relies on simulations statistics over low dimensions (where failures can be observed) followed by extrapolation to the required higher dimensions. Such a study [9] supports the DFR estimation of the BGF decoder (that BIKE uses) with some solid confidence but it does not establish a proven upper bound. Future advances may close this gap, so we recommend keeping the full-fledged version of BIKE [3]. On the other hand, usages (e.g., TLS 1.3) that target forward secrecy and hence use ephemeral keys can settle with an IND-CPA KEM. These seem to be a significant class of usages, and we cite the following quote:

> [M. Campagna & P. Kampanakis; pqc-forum; Dec 12, 2023]
> ... For systems that don't require forward secrecy, the only use cases we see are ones in which there is a requirement that both the ciphertext and the data to be encrypted are relatively small. Not too many use cases come to mind, offline key escrow where storage or transmission is at a premium. ... It may also encourage developers to forgo providing forward secrecy at the prospects of bandwidth benefits. We see this as a rollback in cryptographic engineering practices.

Targeting IND-CCA property. We point out that, at least for two code-based KEMs that remain in NIST's standardization project, Hamming Quasi-Cyclic (HQC) [1] and BIKE, achieving an IND-CCA security claim is a much higher bar to chase and to implement, compared to IND-CPA.

HQC. Similarly to BIKE, the IND-CCA property of HQC requires a sufficiently low upper bound for the decoding failure. Currently this property depends on the validity of a modeling assumption [1]:

> "to provide an upper bound on the decryption failure probability, an analysis of the distribution of the error vector [..] is provided in Section 2.4",

where Section 2.4 states that

> "We will make the simplifying assumption that the coordinates e'_k of $\mathbf{e}'$ are independent variables [..] We support this modeling of the otherwise intractable weight distribution of e' by extensive simulations",

which are further described in [1].

Thus, currently two finalist code-based KEMs, HQC and BIKE, need to rely on some assumption that is backed up by (extensive) simulations for the IND-CCA claim. Without getting into assessing the assumptions and their experimental justification, it is obvious that no such assumptions are required for only IND-CPA security, and for this property there is a direct reduction to a hard problem in coding theory.

A final statement. We assert that for targeting IND-CPA-only usages, we might as well consider a leaner version of BIKE, with a security reduction to a well established assumption on coding. To this end, we proposed the Lean-BIKE configuration that enjoys higher performance and lower communication bandwidth compared to its (potential) IND-CCA counterpart. We recommend standardizing BIKE in both configurations.

References

1. Aguilar Melchor, C., Aragon, N., Bettaieb, S., Lo ic, B., Blazy, O., Bos, J., Deneuville, J.C., Dion, A., Gaborit, P., Lacan, J., Persichetti, E., Robert, J.M., Véron, P., Zémor, G.: Hamming Quasi-Cyclic (HQC) (2024), https://pqc-hqc.org/download.php?file=hqc-specification_2024-02-23.pdf

2. Alagic, Gorjan and Apon, Daniel and Cooper, David and Dang, Quynh and Dang, Thinh and Kelsey, John and Lichtinger, Jacob and Miller, Carl and Moody, Dustin and Peralta, Rene and Perlner, Ray and Robinson, Angela and Smith-Tone, Daniel and Liu, Yi-Kai: Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process (2022). https://doi.org/https://doi.org/10.6028/NIST.IR.8413-upd1
3. Aragon, N., Barreto, P.S.L.M., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.C., Gaborit, P., Ghosh, S., Gueron, S., Güneysu, T., Melchor, C.A., Misoczki, R., Persichetti, E., Richter-Brockmann, J., Sendrier, N., Tillich, J.P., Vasseur, V., Zémor, G.: BIKE: Bit Flipping Key Encapsulation v5.1 (oct 2022), https://bikesuite.org/files/v5.0/BIKE_Spec.2022.10.10.1.pdf
4. Drucker, N., Gueron, S.: A toolbox for software optimization of QC-MDPC code-based cryptosystems. Journal of Cryptographic Engineering pp. 1–17 (jan 2019). https://doi.org/10.1007/s13389-018-00200-4
5. Drucker, N., Gueron, S., Dusan, K.: Additional implementation of BIKE (Bit Flipping Key Encapsulation), commit 40519b8338ebe1f7bcd0efd8419a180642d94aa4. https://github.com/awslabs/bike-kem (2022)
6. Drucker, N., Gueron, S., Dusan, K.: Isochronous implementation of the errors-vector generation of BIKE, commit 46d757bfdb359e38847d355ad69306d3ea66259a. https://github.com/awslabs/bike-kem/blob/master/BIKE_Rejection_Sampling.pdf (2022)
7. Drucker, N., Gueron, S., Kostic, D.: Fast Polynomial Inversion for Post Quantum QC-MDPC Cryptography. In: Dolev, S., Kolesnikov, V., Lodha, S., Weiss, G. (eds.) Cyber Security Cryptography and Machine Learning. pp. 110–127. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-49785-9_8
8. Drucker, N., Gueron, S., Kostic, D.: On Constant-Time QC-MDPC Decoders with Negligible Failure Rate. In: Baldi, M., Persichetti, E., Santini, P. (eds.) Code-Based Cryptography. pp. 50–79. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-54074-6_4
9. Drucker, N., Gueron, S., Kostic, D.: QC-MDPC Decoders with Several Shades of Gray. In: Ding, J., Tillich, J.P. (eds.) Post-Quantum Cryptography. pp. 35–50. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-44223-1_3
10. Drucker, N., Gueron, S., Kostic, D.: Binding BIKE Errors to a Key Pair. In: Dolev, S., Margalit, O., Pinkas, B., Schwarzmann, A. (eds.) Cyber Security Cryptography and Machine Learning. pp. 275–281. Springer International Publishing, Cham (2021). https://doi.org/978-3-030-78086-9_21
11. Drucker, N., Gueron, S., Kostic, D.: To Reject or Not Reject: That Is the Question. The Case of BIKE Post Quantum KEM. In: Latifi, S. (ed.) ITNG 2023 20th International Conference on Information Technology-New Generations. pp. 125–131. Springer International Publishing, Cham (2023). https://doi.org/978-3-031-28332-1_15
12. Drucker, N., Gueron, S., Kostic, D., Persichetti, E.: On the applicability of the Fujisaki-Okamoto transformation to the BIKE KEM. Int. J. Comput. Math. Comput. Syst. Theory **6**(4), 364–374 (2021). https://doi.org/10.1080/23799927.2021.1930176
13. Guo, Q., Hlauschek, C., Johansson, T., Lahr, N., Nilsson, A., Schröder, R.L.: Don't Reject This: Key-Recovery Timing Attacks Due to Rejection-Sampling in HQC and BIKE. IACR Transactions on Cryptographic Hardware and Embedded Systems **2022**(3), 223–263 (Jun 2022), https://doi.org/10.46586/tches.v2022.i3.223-263

14. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A Modular Analysis of the Fujisaki-Okamoto Transformation. In: Kalai, Y., Reyzin, L. (eds.) Theory of Cryptography. pp. 341–371. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-70500-2_12
15. Sendrier, N.: Secure sampling of constant-weight words – application to bike. Cryptology ePrint Archive, Paper 2021/1631 (2021), https://eprint.iacr.org/archive/2021/1631/20220927:074053
16. Sendrier, N., Vasseur, V.: On the Decoding Failure Rate of QC-MDPC Bit-Flipping Decoders. In: Ding, J., Steinwandt, R. (eds.) Post-Quantum Cryptography. vol. 2, pp. 404–416. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-25510-7
17. Vasseur, V.: Post-quantum cryptography: a study of the decoding of QC-MDPC codes. Theses, Université de Paris (Mar 2021), https://theses.hal.science/tel-03254461
18. Vasseur, V.: Qc-mdpc codes dfr and the ind-cca security of bike. Cryptology ePrint Archive, Paper 2021/1458 (2021), https://eprint.iacr.org/2021/1458
19. Wang, T., Wang, A., Wang, X.: Exploring decryption failures of bike: New class of weak keys and key recovery attacks. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023. pp. 70–100. Springer Nature Switzerland, Cham (2023). https://doi.org/978-3-031-38548-3_3