BIKE

3rd NIST PQC Standardization Workshop

June 8th, 2021

Nicolas Aragon, University of Limoges, France
Paulo S. L. M. Barreto, University of Washington Tacoma, USA
Slim Bettaieb, Worldline, France
Loïc Bidoux, Worldline, France
Olivier Blazy, University of Limoges, France
Jean-Christophe Deneuville, ENAC, Federal University of Toulouse, France
Philippe Gaborit, University of Limoges, France
Santosh Ghosh, Intel, USA
Shay Gueron, University of Haifa, and Amazon Web Services, Israel & USA

Tim Güneysu, Ruhr-Universität Bochum & DFKI, Germany
Carlos Aguilar Melchor, University of Toulouse, France
Rafael Misoczki, Google, USA
Edoardo Persichetti, Florida Atlantic University, USA
Jan Richter-Brockmann, Ruhr-Universität Bochum, Germany
Nicolas Sendrier, INRIA, France
Jean-Pierre Tillich, INRIA, France
Valentin Vasseur, INRIA, France
Gilles Zémor, IMB, University of Bordeaux, France

https://bikesuite.org

Agenda

- BIKE recap
- A hardware-friendly tweak
- BIKE adoption
- New team member - Jan Richter-Brockmann

BIKE Recap

- Niederreiter-based KEM instantiated with QC-MDPC codes
- Leverage Fujisaki-Okamoto CCA Transform[1]
- State-of-the-art QC-MDPC Decoding Failure Rate analysis[2]
- Black-Gray-Flip Decoder implemented in constant time[3]

1: For an updated analysis of the FO transform applied to BIKE, see: Drucker, N., Gueron, S., Kostic, D., & Persichetti, E. (2021). On the applicability of the Fujisaki-Okamoto transformation to the BIKE KEM. Intl. Journal of Computer Mathematics: Computer Systems Theory.
2: For a comprehensive discussion on Decoding Failure Rate of BIKE decoders, see: Valentin Vasseur's PhD thesis "Post-quantum cryptography: study on the decoding of QC-MDPC codes", 2021, available at: https://who.rocq.inria.fr/Valentin.Vasseur/phd-defence/
3: See BIKE's Additional Implementation available at: https://github.com/awslabs/bike-kem and paper by N. Drucker, S, Gueron, D. Kostic "QC-MDPC Decoders with Several Shades of Gray". PQCrypto 2020: 35-50

BIKE Recap - Spec

KeyGen : $() \mapsto (h_0, h_1, \sigma), h$

Output: $(h_0, h_1, \sigma) \in \mathcal{H}_w \times \mathcal{M},\ h \in \mathcal{R}$

$(h_0, h_1) \xleftarrow{\$} \mathcal{H}_w$
$h \leftarrow h_1 h_0^{-1}$
$\sigma \xleftarrow{\$} \mathcal{M}$

Encaps : $h \mapsto K, c$

Input: $h \in \mathcal{R}$

Output: $K \in \mathcal{K},\ c \in \mathcal{R} \times \mathcal{M}$

$m \xleftarrow{\$} \mathcal{M}$
$(e_0, e_1) \leftarrow \mathbf{H}(m)$
$c \leftarrow (e_0 + e_1 h, m \oplus \mathbf{L}(e_0, e_1))$
$K \leftarrow \mathbf{K}(m, c)$

Decaps : $(h_0, h_1, \sigma), c \mapsto K$

Input: $((h_0, h_1), \sigma) \in \mathcal{H}_w \times \mathcal{M},\ c = (c_0, c_1) \in \mathcal{R} \times \mathcal{M}$

Output: $K \in \mathcal{K}$

$e' \leftarrow \mathtt{decoder}(c_0 h_0, h_0, h_1)$ ▷ $e' \in \mathcal{R}^2 \cup \{\perp\}$
$m' \leftarrow c_1 \oplus \mathbf{L}(e')$ ▷ with the convention $\perp = (0, 0)$
if $e' = \mathbf{H}(m')$ **then** $K \leftarrow \mathbf{K}(m', c)$ **else** $K \leftarrow \mathbf{K}(\sigma, c)$

Notation	
$\mathbb{F}_2$:	Binary finite field.
$\mathcal{R}$:	Cyclic polynomial ring $\mathbb{F}_2[X]/(X^r - 1)$.
$\mathcal{H}_w$:	Private key space $\{(h_0, h_1) \in \mathcal{R}^2 \mid \lvert h_0 \rvert = \lvert h_1 \rvert = w/2\}$
$\mathcal{E}_t$:	Error space $\{(e_0, e_1) \in \mathcal{R}^2 \mid \lvert e_0 \rvert + \lvert e_1 \rvert = t\}$
$\lvert g \rvert$:	Hamming weight of a binary polynomial $g \in \mathcal{R}$.
$u \xleftarrow{\$} U$:	Variable u is sampled uniformly at random from the set U.
$\oplus$:	exclusive or of two bits, componentwise with vectors

- $\mathbf{H} : \mathcal{M} \to \mathcal{E}_t$.
- $\mathbf{K} : \mathcal{M} \times \mathcal{R} \times \mathcal{M} \to \mathcal{K}$.
- $\mathbf{L} : \mathcal{R}^2 \to \mathcal{M}$.

BIKE Recap - Performance

	AVX2	AVX512	VPCLMUL
KeyGen	600	585	470
Encaps	220	205	195
Decaps	2220	1356	1280

Latency cost for BIKE Level 1 in kilocycles
(Additional Implementation)

Message Flow	Message	Size	Level 1	Level 3	Level 5
Init. $\rightarrow$ Resp.	h	r	12,323	24,659	40,973
Resp. $\rightarrow$ Init.	C	$r + \ell$	12,579	24,915	41,229

Communication cost in bits

Random Oracles in BIKE Specification

Algorithm 1: Encapsulation.

Input : Public key h.
Output: Encapsulated key K and ciphertext $C = (c_0, c_1)$.

Generate $m \xleftarrow{\$} \{0,1\}^{\ell}$ uniformly at random.
Compute $(e_0, e_1) \leftarrow \mathbf{H}(m)$.
Compute $C = (c_0, c_1) \leftarrow (e_0 + e_1 h, m \oplus \mathbf{L}(e_0, e_1))$.
Compute $K \leftarrow \mathbf{K}(m, C)$.
Return (C, K).

Algorithm 1: Decapsulation.

Input : Private key (h_0, h_1, σ) and ciphertext $C = (c_0, c_1)$.
Output: Decapsulated key K.

Compute syndrome $s \leftarrow c_0 h_0$.
Compute $\{(e_0', e_1'), \perp\} \leftarrow \mathtt{decoder}(s, h_0, h_1)$.
Compute $m' \leftarrow c_1 \oplus \mathbf{L}(e_0', e_1')$.
if $\mathbf{H}(m') \neq (e_0', e_1')$ **then**
  Compute $K \leftarrow \mathbf{K}(\sigma, C)$.
else
  Compute $K \leftarrow \mathbf{K}(m', C)$.
Return K.

Random Oracles: H, K, and L

$$\mathbf{H}: \{0,1\}^{l} \rightarrow \{0,1\}^{2r}_{[t]}$$
$$\mathbf{K}: \{0,1\}^{r+2l} \rightarrow \{0,1\}^{l}$$
$$\mathbf{L}: \{0,1\}^{2r} \rightarrow \{0,1\}^{l}$$

Replaced the underlying symmetric cryptographic primitives

New

Implementation of our Random Oracles

Function	Old	New
H	AES-256	SHAKE-256
K	SHA2-384	SHA3-384
L	SHA2-384	SHA3-384
PRNG	AES-256	SHA3-384

All KECCAK-based

Only one cryptographic primitive is required instead of two

Software

	Spec 4.1 to 4.2 Slowdown
Key Generation	+1.79%
Encapsulation	+13.54%
Decapsulation	+3.21%

Clock cycles difference for Level 1 on a machine with an Intel Xeon CPU E5-1660 3.2 GHz, 128 GB RAM (Reference Implementation)

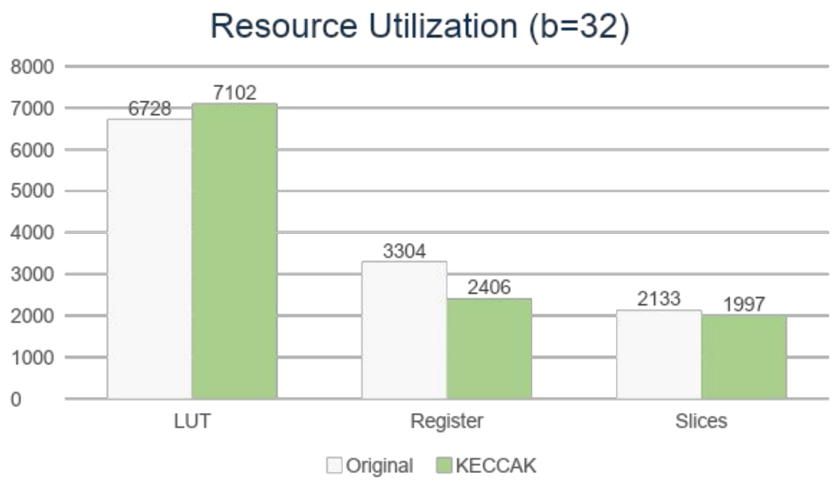
Smaller and faster hardware implementation

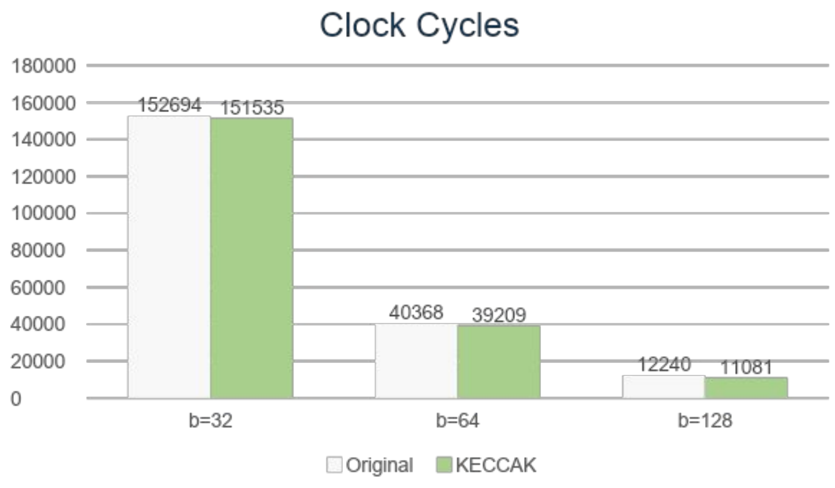
at the cost of a slightly slower software implementation

Obs: Recall that Encaps is by far the fastest BIKE step (~200 kcycles Additional implementation), thus a ~13% penalty is in practice minor

Hardware

Encapsulation





BIKE Adoption - Status Update

AWS Security Blog

Post-quantum TLS now supported in AWS KMS

by Andrew Hopkins | on 04 NOV 2019 | in Advanced (300), AWS Key Management Service, Security, Identity, & Compliance | Permalink | Comments | Share

```
Internet Engineering Task Force                              M. Campagna
Internet-Draft                                               E. Crockett
Intended status: Experimental                                        AWS
Expires: September 10, 2021                               March 9, 2021

  Hybrid Post-Quantum Key Encapsulation Methods (PQ KEM) for Transport
                      Layer Security 1.2 (TLS)
                draft-campagna-tls-bike-sike-hybrid-06

Abstract

   Hybrid key exchange refers to executing two independent key exchanges
   and feeding the two resulting shared secrets into a Pseudo Random
   Function (PRF), with the goal of deriving a secret which is as secure
   as the stronger of the two key exchanges.  This document describes
   new hybrid key exchange schemes for the Transport Layer Security 1.2
   (TLS) protocol.  The key exchange schemes are based on combining
   Elliptic Curve Diffie-Hellman (ECDH) with a post-quantum key
   encapsulation method (PQ KEM) using the existing TLS PRF.
```

The Open Quantum Safe (OQS) project is an open-source project that aims to support the development and prototyping of quantum-resistant cryptography.

New Team Member

- Jan Richter-Brockmann
  - PhD Candidate - Ruhr-Universität Bochum
  - Intern at Intel Labs
  - Area of expertise: efficient Hardware cryptographic implementations

Thank you

https://bikesuite.org