SIKE

Supersingular Isogeny Key Encapsulation

David Jao, UW
Craig Costello, MSR
Aaron Hutchinson, UW
Brian Koziel, TI
Michael Naehrig, MSR
Vladimir Soukharev, ISG
Reza Azarderakhsh, FAU
Luca De Feo, IBM
Amir Jalali, LinkedIn
Brian LaMacchia, MSR
Geovandro Pereira, UW
David Urbanik, UofT
Matthew Campagna, Amazon
Basil Hess, IBM
Koray Karabina, NRC
Patrick Longa, MSR
Joost Renes, NXP

`https://sike.org/`

June 8, 2021, Third PQC Standardization Conference

SIKE

Supersingular **I**sogeny **K**ey **E**ncapsulation (SIKE)

- IND-CCA2 KEM
- Based on **S**upersingular **I**sogeny **D**iffie-**H**ellman (SIDH)
- Uses Hofheinz et al. transformation (TCC 2017) on SIDH to achieve CCA security

The SIKE protocol specifies:

- Parameter sets
- Key/ciphertext formats
- Encapsulation/decapsulation mechanisms
- Choice of symmetric primitives (hash functions, etc.)

SIDH: simplified overview

0. Starting supersingular curve
   $E \; : \; y^2 = x^3 + 6x^2 + x$ over $\mathbb{F}_{p^2}$;

1. Alice chooses a kernel $A \subset E(\mathbb{F}_{p^2})$
   and sends E/A to Bob;

2. Bob chooses a kernel $B \subset E(\mathbb{F}_{p^2})$
   and sends E/B to Alice;

3. Shared secret: $E/\langle A, B\rangle = (E/A)/\phi_A(B) = (E/B)/\phi_B(A)$.

$$\begin{tikzcd}
g \arrow[r] \arrow[d] \arrow[dr, phantom, "\text{DH}"] & g^a \arrow[d] \\
g^b \arrow[r] & g^{ab}
\end{tikzcd}
\qquad
\begin{tikzcd}
E \arrow[r, "\phi_A"] \arrow[d, "\phi_B"'] \arrow[dr, phantom, "\text{SIDH}"] & E/A \arrow[d] \\
E/B \arrow[r] & E/\langle A, B\rangle
\end{tikzcd}$$

SIDH: detailed overview

Curves represented by public triplets of *torsion points*

"Alice's" *torsion basis*:

$$E^A := (P_A, Q_A, R_A := P_A - Q_A)$$

"Bob's" *torsion basis*:

$$E^B := (P_B, Q_B, R_B := P_B - Q_B)$$

Alice | **Bob**

$$A \subset \langle P_A, Q_A \rangle \updownarrow \phi_A \qquad B \subset \langle P_B, Q_B \rangle \updownarrow \phi_B$$

$$\xrightarrow{E^B/A := (\phi_A(P_B), \phi_A(Q_B), \phi_A(R_B))}$$

$$\xleftarrow{E^A/B := (\phi_B(P_A), \phi_B(Q_A), \phi_B(R_A))}$$

Shared secret:

$$j(E/\langle A, B\rangle) = j((E^B/A)/\phi_A(B)) = j((E^A/B)/\phi_B(A)).$$

Changes for SIKE in third round

- New optimized ARMv8, Cortex M4, and VHDL implementations.
- Key compression:
  - Changed format of compressed ciphertexts (12.5% larger than in round 2).
  - Major improvements in speed and memory usage.

Changes for SIKE in second round

- New parameter sets: SIKEp434, SIKEp503, SIKEp610, SIKEp751, ~~SIKEp964~~;
- Updated security analysis.
- Starting curve changed;
- Introduced key compression: $\approx 40\%$ smaller public keys and ciphertexts;

Parameter sets

Scheme	prime p	$\log_2 p$	Security level
SIKEp434	$2^{216}3^{137} - 1$	434	NIST 1
SIKEp503	$2^{250}3^{159} - 1$	503	NIST 2
SIKEp610	$2^{305}3^{192} - 1$	610	NIST 3
SIKEp751	$2^{372}3^{239} - 1$	751	NIST 5

Performance

Scheme	Public key	ciphertext	Encaps	Decaps
	bytes		10^6 cycles (x86_64 asm)	
SIKEp434	330	346	9.7	10.3
SIKEp434_compressed	197	236	15.1	11.0
SIKEp503	378	402	13.6	14.4
SIKEp503_compressed	225	280	21.2	15.7
SIKEp610	462	486	27.3	27.4
SIKEp610_compressed	274	336	37.5	29.2
SIKEp751	564	596	40.7	43.9
SIKEp751_compressed	335	410	63.3	46.6

Memory footprint of compression 3–10$\times$ smaller compared to Round 2.

Additional implementations

	Scheme	Cortex M4 (ARMv7)[1]		Cortex A72 (ARMv8)	
		Encaps	Decaps	Encaps	Decaps
ARM implementations (10^6 cycles)	SIKEp434	69	74	28	30
	SIKEp503	97	104	40	42
	SIKEp610	198	199	90	91
	SIKEp751	299	321	136	146

	Scheme	Xilinx Artix-7		Xilinx Kintex UltraScale+	
		Encaps	Decaps	Encaps	Decaps
VHDL implementation (FPGA, ms)	SIKEp434	7.01	7.42	3.09	3.28
	SIKEp503	8.81	9.25	3.75	3.93
	SIKEp610	14.43	14.22	6.02	5.94
	SIKEp751	17.37	18.39	7.43	7.87

[1] M. Anastasova, R. Azarderakhsh, M. Mozaffari Kermani, "Fast Strategies for the Implementation of SIKERound 3 on ARM Cortex-M4", `https://ia.cr/2021/115`.

Recent developments

- SIKE's speed has greatly improved over the last 10 years.
- Improvements, especially in software, become harder to come by.
- [BI'21] applies a Polynomial Modular Number System (PMNS) representation to finite fields in SIKE:
  - Does not appear to be competitive for SIKE's proposed parameters;
  - Suggests new level 5 parameter, `p736`, which is $1.17\times$ faster.
  - [TWLLWG'20] had explored similar ideas previously, but had not demonstrated a speed-up.

[1] [BI'21] C. Bouvier, L. Imbert. "An Alternative Approach for SIDH Arithmetic", PKC 2021, https://ia.cr/2020/1385.

[2] [TWLLWG'20] J. Tian, P. Wang, Z. Liu, J. Lin, Z. Wang, J. Großschädl "Faster Software Implementation of the SIKE Protocol Based on a New Data Representation", https://ia.cr/2020/660.

Recent developments

- Best attack is the generic van Oorschot-Wiener (vOW) parallel collision finding algorithm.
- Current parameter selection penalizes SIKE: **memory is assumed to be free**.
- [LWS'21] uses a budget-based cost model to derive a more realistic security estimation:
  - Takes into account processing (ASICs) and memory costs needed for cryptanalysis,
  - Suggests new smaller parameters fit NIST levels more closely,

NIST level	SIKE round 3				[LWS'21]			
	$\log p$	public key bytes	Encaps 10^6 cycles	Decaps 10^6 cycles	$\log p$	public key bytes	Encaps 10^6 cycles	Decaps 10^6 cycles
1	434	330	9.7	10.3	377	288 B	7.3	7.2
3	610	462	27.3	27.4	546	414 B	19.9	19.9
5	751	564	40.7	43.9	697	528 B	33.3	35.0

[1] [LWS'21] P. Longa, W. Wang, J. Szefer: "The Cost to Break SIKE: A Comparative Hardware-Based Analysis with AES and SHA-3", CRYPTO 2021. `https://eprint.iacr.org/2020/1457`.

Summary

SIKE advantages:

- Smallest public key size. Key compression has become almost free.
- Straightforward parameter selection.
- No decryption error, Gaussians, rejection sampling, etc.
- Generic attacks are well understood.
- Only KEM proposal not based on lattices / codes / LW[ER].

SIKE disadvantages:

- Slow.
- Non-generic attacks may one day pose a threat (they are currently far from it).

Work in progress:

- Side channel attacks, cryptanalysis.
- Do not miss Craig Costello's talk tomorrow!