The Cost to Break SIKE

A Comparative Hardware-Based Analysis with AES and SHA-3

Patrick Longa Wen Wang Jakub Szefer

Microsoft Yale

CRYPTO 2021

Virtual Conference, August 2021

Summary: the problem

- SIKE is an alternate candidate in the 3rd round of the NIST post-quantum cryptography (PQC) standardization process
  - The only isogeny-based scheme in competition.
- Despite its solid security history, SIKE is penalized because its parameters are chosen using a very conservative Random Access Memory (RAM) model.

In this work:

- We analyze SIKE, AES and SHA-3 using a more realistic budget-based cost model
  - The model considers both **computing** and **memory** costs, together with historical price data and the use of a conservative projection of future costs.
- We propose an efficient architecture for ASICs for the isogeny computation
  - This is used to model an ASIC-powered instance of the von Oorschot-Wiener (vOW) parallel collision finding algorithm.

Summary: the results

- ❑ As conjectured, current SIKE parameters are conservative and offer a wide margin of (quantum and classical) security.
- ❑ We propose **new parameters** that fit more closely the NIST targets
  - The new parameters are still chosen conservatively.
  - Our software implementation of SIKE shows significant gains in execution time and bandwidth.

Post-quantum key exchange from supersingular isogenies: basics

Supersingular isogeny key exchange

Supersingular isogeny Diffie-Hellman key exchange (SIDH)

- ❑ Proposed by Jao and De Feo in 2011.
- ❑ Very solid security history for its young age
  - Best known attack is classical von Oorschot-Wiener (vOW) parallel collision finding algorithm.
  - Exponential complexity of best classical and quantum attacks.

Supersingular isogeny key encapsulation (SIKE)

- ❑ Designed by Costello–De Feo–Jao–L–Naehrig–Renes in 2017.
- ❑ IND-CCA secure key encapsulation protocol based on SIDH.
- ❑ Alternate candidate in the 3rd round of the NIST PQC standardization process.

Elliptic curves and isogenies

- Let E_1 and E_2 be elliptic curves defined over an extension field L.
- An isogeny is a (non-constant) rational map

$$\phi: E_1 \to E_2$$

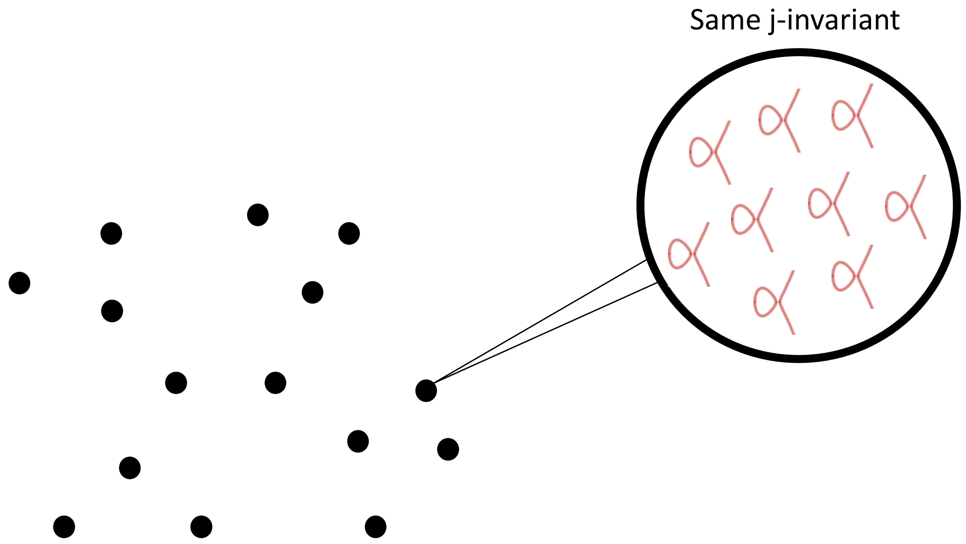
that preserves identity, i.e., $\phi(\mathcal{O}_{E_1}) \to \mathcal{O}_{E_2}$.

Some relevant properties:

- Isogenies are group homomorphisms.
- For every finite subgroup $\langle G \rangle \subseteq E_1$, there is a unique curve E_2 (up to isomorphism) and isogeny $\phi: E_1 \to E_2$ with kernel $\langle G \rangle$. Write $E_2 = \phi(E_1) = E_1/\langle G \rangle$.
- **Isomorphism classes** are determined by the ***j*-invariant**: $j(E) = 1728 \cdot \frac{4a^3}{4a^3+27b^2}$
- There are $\sim\lfloor \boldsymbol{p/12} \rfloor$ **isomorphism classes** of supersingular curves.

Supersingular isogeny graphs

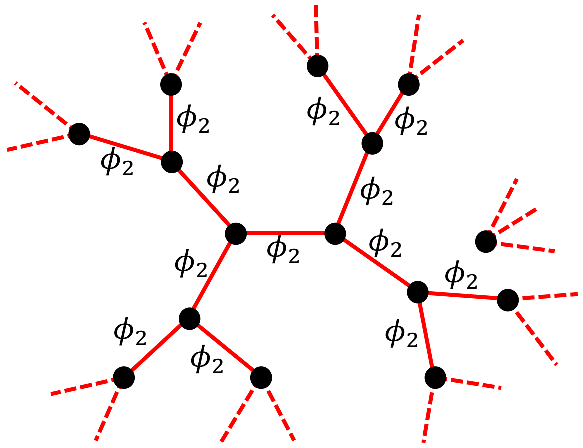
- Vertices: the $\sim\lfloor p/12 \rfloor$ isomorphism classes of supersingular curves over $\mathbb{F}_{p^2}$.



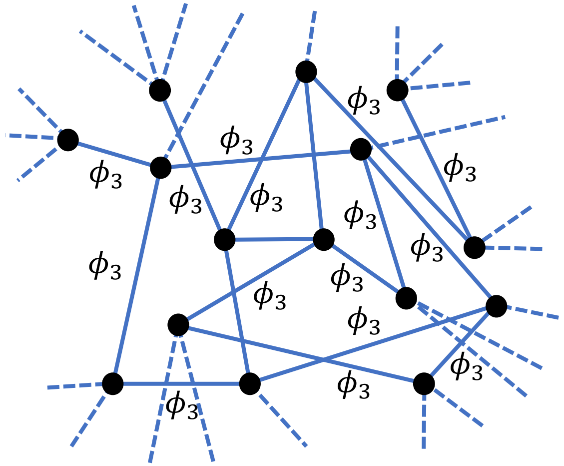
Supersingular isogeny graphs

- Vertices: the $\sim\lfloor p/12 \rceil$ isomorphism classes of supersingular curves over $\mathbb{F}_{p^2}$.
- Edges: isogenies of a fixed prime degree $\ell \nmid p$

For any prime $\ell \nmid p$, there exist $(\ell + 1)$ isogenies of degree ℓ originating from every supersingular curve.



$\ell = 2$



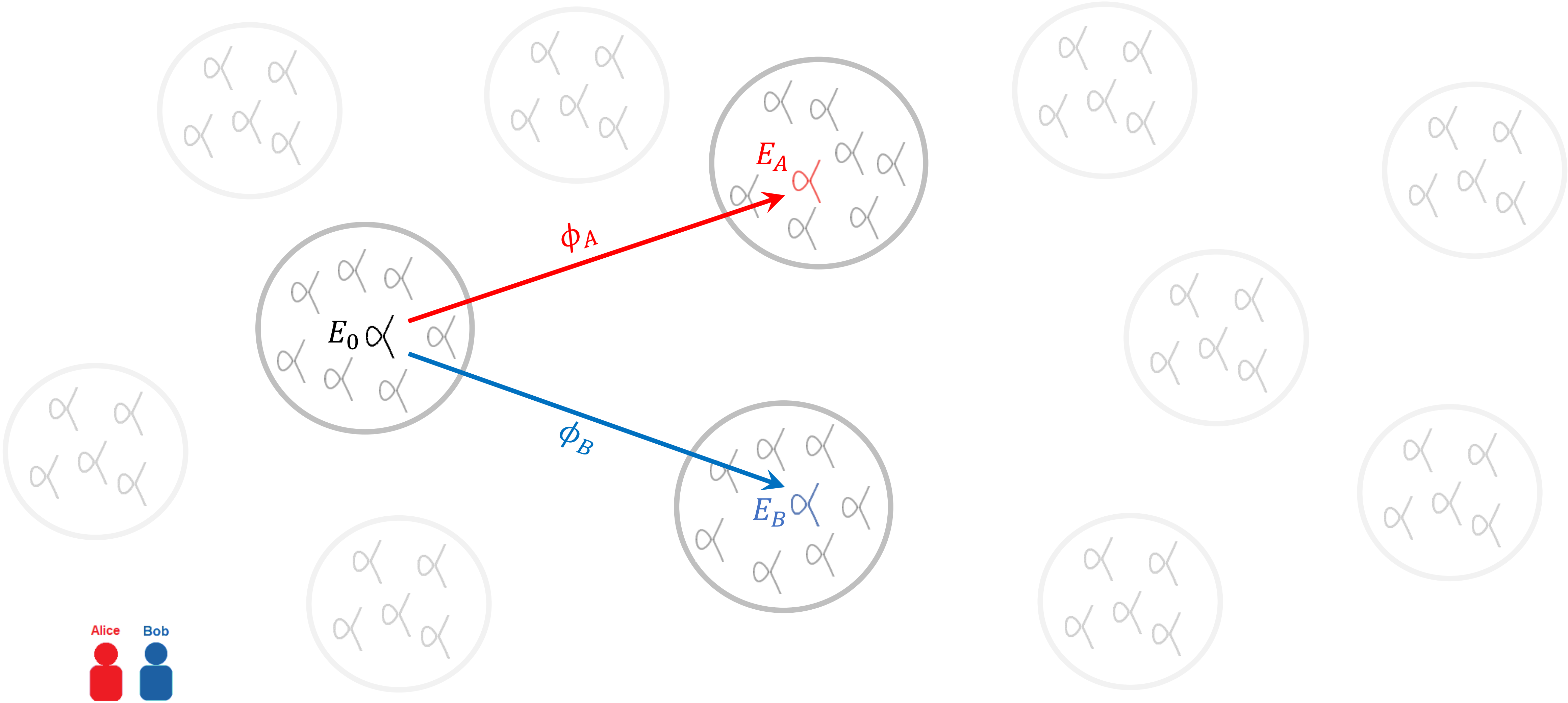
$\ell = 3$

SIDH and SIKE

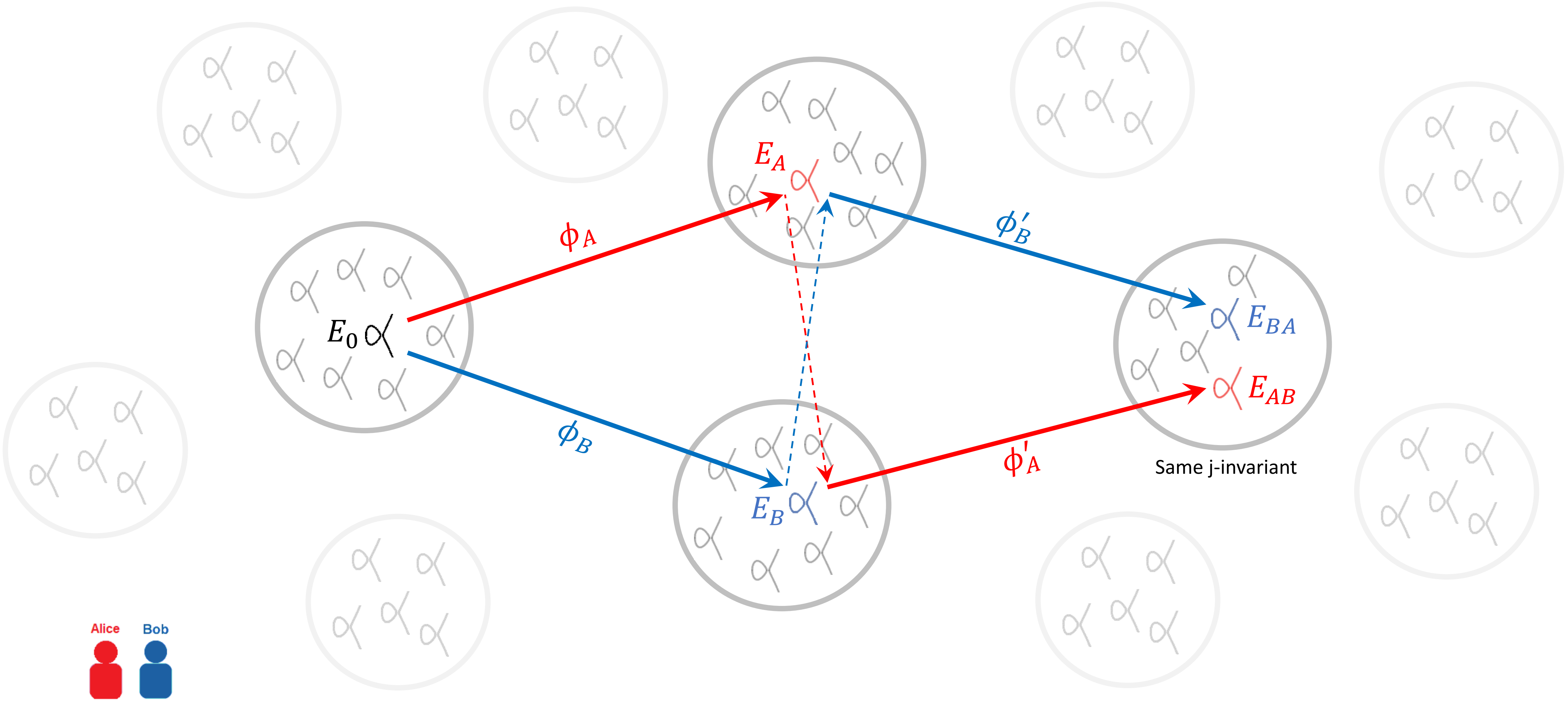
SIDH in a nutshell

SIDH in a nutshell

SIDH in a nutshell



SIDH in a nutshell



From SIDH to SIKE

- ❑ SIDH is not secure when keys are reused (Galbraith-Petit-Shani-Ti 2016)
  - Only recommended in **ephemeral mode.**
- ❑ SIKE uses a variant of Hofheinz–Hövelmanns–Kiltz (HHK) transform:

IND-CPA PKE → IND-CCA KEM

- ❑ HHK transform is secure in both the classical and quantum ROM models.

Supersingular isogeny key encapsulation (SIKE)

KeyGen

1. $s_B \in_R [0, 2^{\lfloor \log_2 3^{e_B} \rfloor})$
2. Set $ker(\phi_B) = \langle P_B + [s_B]Q_B \rangle$
3. $\text{pk}_B = \{\phi_B(E_0), \phi_B(P_A), \phi_B(Q_A)\}$
4. $s \in_R \{0,1\}^n$
5. **keypair:** $\{\text{sk}_B = (s, s_B), \text{pk}_B\}$

Decaps

1. $j' = j(E_{BA}) = j(\phi'_B(\phi_A(E_0)))$
2. $m' = F(j') \oplus c[2]$
3. $r' = G(m', \text{pk}_B) \bmod 2^{e_A}$
4. Set $ker(\phi_A) = \langle P_A + [r']Q_A \rangle$
5. $\text{pk}'_A = \{\phi_A(E_0), \phi_A(P_B), \phi_A(Q_B)\}$
6. If $\text{pk}'_A = c[1]$ then
   Shared key: $ss = H(m', c)$
7. Else $ss = H(s, c)$

decryption (steps 1–2)

partial re-encryption (steps 4–5)

pk_B →

← $c = (\text{pk}_A, F(j) \oplus m)$

Encaps

1. message $m \in_R \{0,1\}^n$
2. $r = G(m, \text{pk}_B) \bmod 2^{e_A}$
3. Set $ker(\phi_A) = \langle P_A + [r]Q_A \rangle$
4. $\text{pk}_A = \{\phi_A(E_0), \phi_A(P_B), \phi_A(Q_B)\}$
5. $j = j(E_{AB}) = j(\phi'_A(\phi_B(E_0)))$
6. **Shared key:** $ss = H(m, c)$

encryption (steps 3–5)

F, G, H instantiated with SHAKE256.

SIDH/SIKE security

Setting: supersingular curves $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$, a large prime p, and isogeny $\phi: E_1 \rightarrow E_2$ with fixed, smooth, public degree.

Computational supersingular isogeny (CSSI) problem: given $P, Q \in E_1$ and $\phi(P), \phi(Q) \in E_2$, compute ϕ.

Parameter selection

How is security strength estimated?

A simplistic but conservative approach:

- ❑ Use query complexity of cryptanalysis
  - E.g., a brute force attack on AES128 takes $\sim 2^{128}$ AES execution calls.
- ❑ Use some variant of the Random Access Memory (RAM) model
  - Could use gate/instruction/cycle count to estimate attack iteration cost.
  - E.g., AES128 security is estimated as $2^{128} \times 2^{15} = 2^{143}$ classical gates, since AES128 can be implemented in $\sim 2^{15}$ gates according to NIST [1].
- ❑ Cons: communication and memory costs are ignored

Crucially, cryptosystems whose cryptanalysis incurs high memory costs are penalized

[1] NIST PQC Call for Proposals (2017), https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/Call-for-Proposals

SIKE parameter selection

The number of order-$\ell^{e/2}$ subgroups of $E[\ell^e] = |S| \approx p^{1/4}$

- SIKE team (Round 1):
  - CSSI, modeled as black-box claw finding problem, can be solved classically using MitM in time and space complexity $\mathcal{O}(p^{1/4})$.
  - E.g., $\lceil \log_2 p \rceil \approx 512$ is expected to match the 128-bit security of AES128 (NIST level 1).

SIKE parameter selection

- Adj–Cervantes-Vázquez–Chi-Domínguez–Menezes–Rodríguez-Henríquez (2018):
  - The amount of memory required by the claw finding problem is unrealistic.
  - If the cost of memory is considered, the *best classical attack* is the **vOW algorithm.**

Assume storage $w = 2^{80}$. vOW on SIKE runs in time

$$\mathcal{O}\left(\frac{|S|^{\frac{3}{2}}}{\sqrt{w}}\right)$$

- E.g., $\lceil \log_2 p \rceil \approx 448$ is expected to match the 128-bit security of AES128 (NIST level 1).

SIKE parameters: from Round 1 to Round 2

	Round 1	Round 2
Level 1	SIKEp503	SIKEp434 (New)
Level 2	--	SIKEp503
Level 3	SIKEp751	SIKEp640 (New)
Level 5	~~SIKEp976~~	SIKEp751

- Parameter set name: SIKEp + $\lceil \log_2 p \rceil$

SIKE parameters (Round 2 and 3)

Parameter set	Security level	NIST classical gate requirements	SIKE: Total time ($w = 2^{80}$)	SIKE: x64 instructions ($w = 2^{80}$)
SIKEp434	1	143	128	143
SIKEp503	2	146	152	169
SIKEp610	3	207	189	210
SIKEp751	5	272	-	262

- "… although their times fall slightly below NIST's required gate counts, the corresponding conversion to gate counts would see these parameters comfortably exceed NIST's requirements." (SIKE specification, accessed on 07.29.2021)
- No fundamental reason to fix $w = 2^{80}$.

A budget-based cost model

Budget-based cost model

1. Fix a budget for the attacker.
2. Distribute $$$ to get computing and storage resources
   - Goal: to minimize time to break algorithm.
3. The scheme's security is estimated as the time it takes to break it.

❑ Similar approaches can be found in the literature, e.g., van Oorschot–Wiener 1994 and 1999.

❑ We improve it further by:

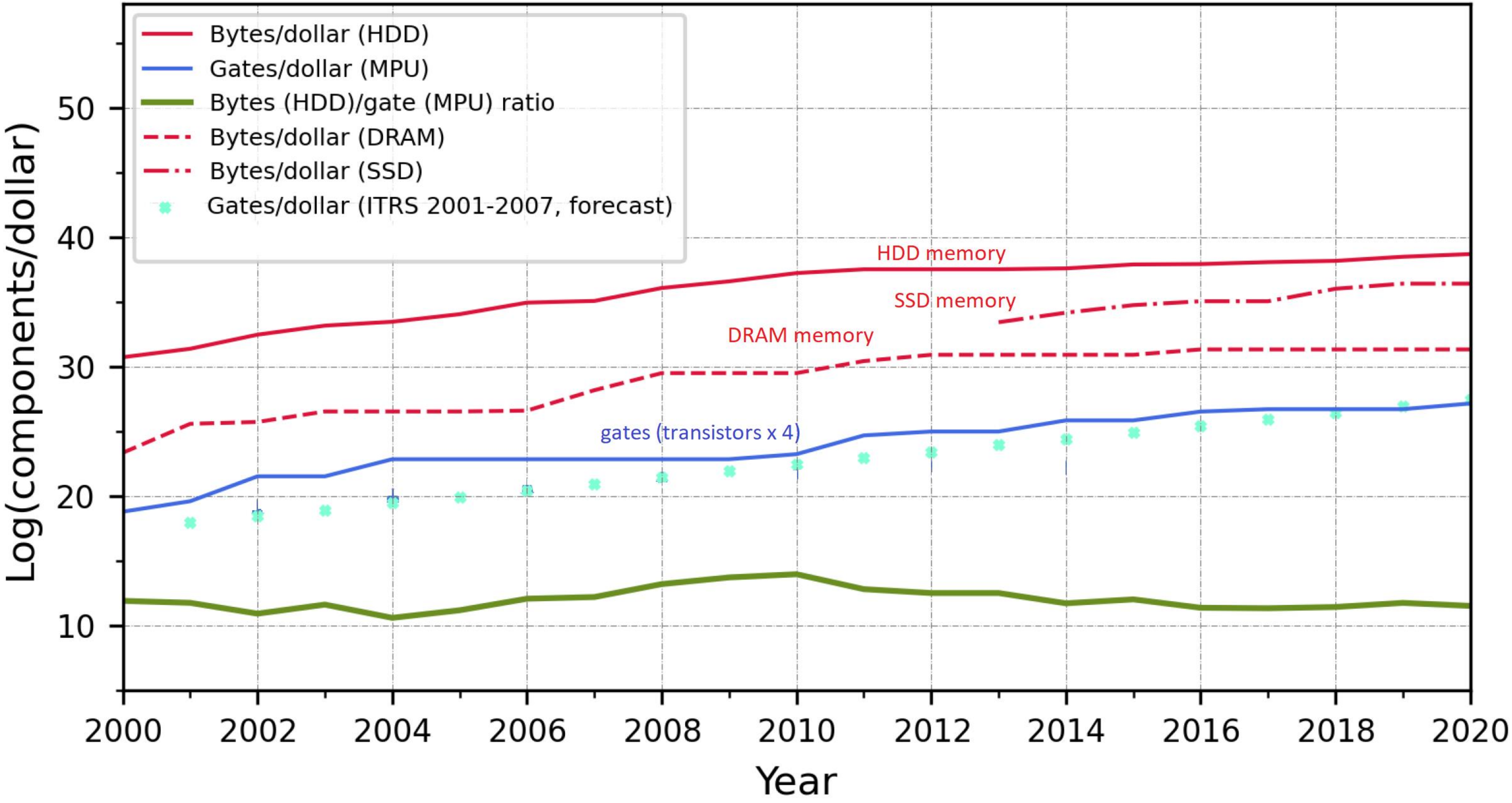
- Analyzing historical price information of semiconductors and memory components.
- Using simplified projections to estimate costs in the future.

Determining component costs

- We compiled public release prices of microprocessor units (MPUs) from Intel and AMD for the years from 2000 to 2020
  - Used public transistor counts and the standard assumption that a gate-equivalent (GE) consists of four transistors (as in a 2-input CMOS NAND gate).
  - Used a scaling factor to approximate release prices to large-scale production costs, closely following the estimation procedure by Khan and Mann [1].
- A similar approach was applied to estimate memory (HDD, DRAM and SSD) costs.
- Results were validated with data from the International Technology Roadmap for Semiconductors (ITRS).

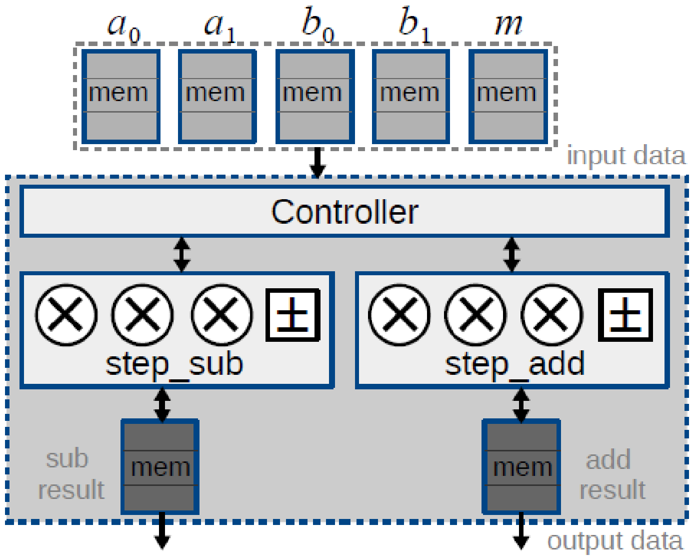
[1] S.M. Khan and A. Mann, “AI chips: What they are and why they matter” (2020).

Historical cost ratio

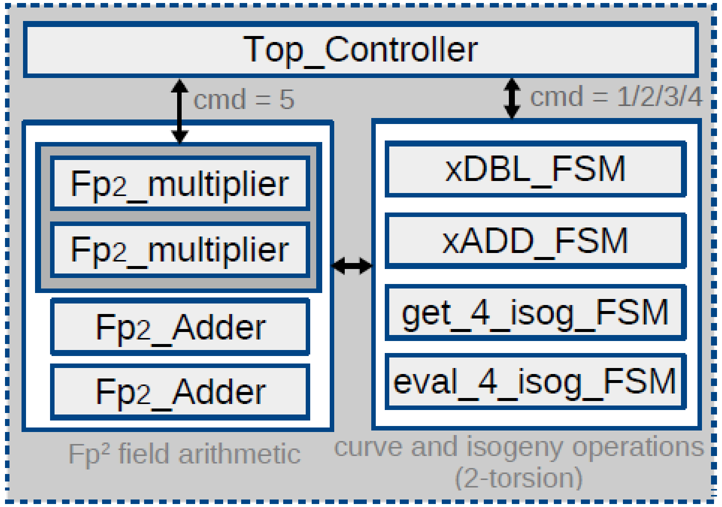


Estimating the cost to implement vOW on SIKE

- We designed a large-degree isogeny HW accelerator for ASIC
  - Goal: to minimize the area-time (AT) product (compared to previous works).
  - Implemented all the main computations: point addition (ADD), point doubling (DBL), isogeny evaluation (4-eval) and isogeny computation (4-get).
  - Based on a novel unified multiplier over $\mathbb{F}_{p^2}$ that enables 'optimal' parallelization.



Multiplier core



Isogeny HW accelerator

Estimating the cost to implement vOW on SIKE

	Radix	Kernel	Tree traversal		
		ADD	DBL	4-get	4-eval
SIKEp434	2^{32}	874	841	598	1105
SIKEp503	2^{32}	1088	1051	742	1383
SIKEp610	2^{64}	518	496	360	649
SIKEp751	2^{64}	684	658	472	863
SIKEp377	2^{32}	684	655	470	859
SIKEp546	2^{32}	1326	1288	904	1697
SIKEp697	2^{64}	634	610	438	800

Cycle counts

	Radix	Area (kGE)	Freq (MHz)	Period (nsec)	Speed	
					cycles	msec
SIKEp434	2^{32}	372.2	167.5	5.97	544930	3.253
SIKEp503	2^{32}	409.5	167.8	5.96	807659	4.814
SIKEp610	2^{64}	748.0	83.75	11.94	485977	5.803
SIKEp751	2^{64}	822.3	84.32	11.86	818106	9.703
SIKEp377	2^{32}	341.3	156.5	6.39	367239	2.347
SIKEp546	2^{32}	441.1	155.8	6.42	1105117	7.095
SIKEp697	2^{64}	798.9	83.68	11.95	719288	8.595

Area and time results

- The time to execute a half-degree isogeny was estimated using the operation counts and cycle results
  - The controller computation and data communication overheads are ignored.
- Area synthesis results were obtained using NanGate 45nm technology
  - Control circuitry to implement the rest of the vOW algorithm and other several components (e.g., the hash function) are ignored.

The cost of breaking AES

- ❑ Assume the use of the efficient parallel attack based on rainbow chains by Oechslin.
- ❑ The time to find a given b-bit key k with m parallel engines and probability close to 1 is

$$\frac{2^b}{m} \cdot t_{\text{AES}}$$

- ❑ Area and time information are obtained from Ueno et al. [1]
  - To our knowledge, the AES implementation with the lowest A/T product on ASIC in the literature.
  - Based on the same NanGate 45nm library used to evaluated SIKE.

[1] Ueno et al., “High throughput/gate AES hardware architectures based on datapath compression” (2020).

The cost of breaking SHA-3

- Assume the use of the vOW parallel collision finding algorithm.
- Let θ be the proportion of points of the search space S that are distinguished. The time to find a collision for a hash function using m parallel engines is

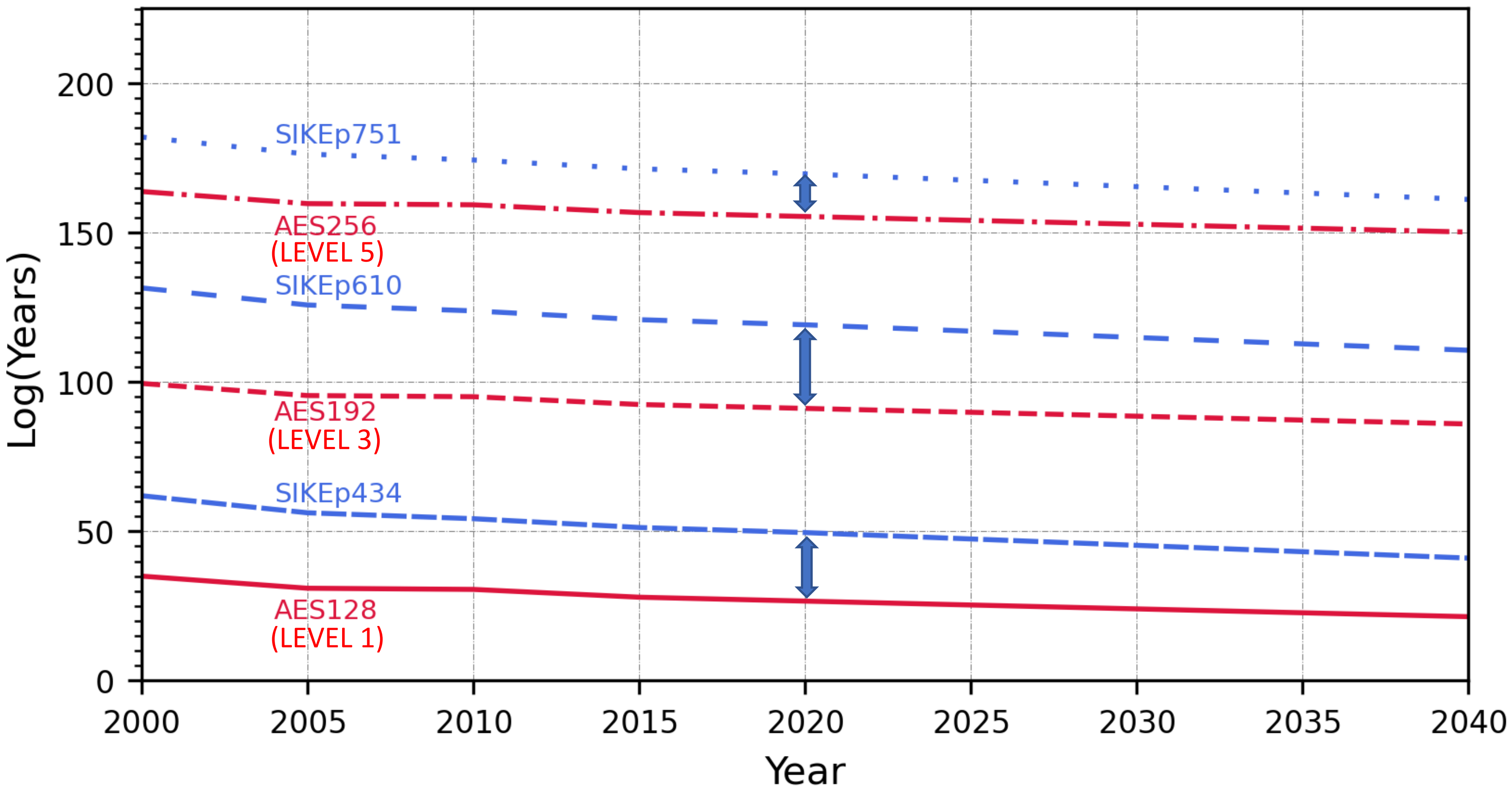
$$\left(\frac{1}{m}\sqrt{\pi|S|/2}+\frac{2.5}{\theta}\right)\cdot t_{\mathrm{SHA-3}}$$

- Area and time information are obtained from Akin et al. [1]
  - To our knowledge, the Keccak implementation with the lowest A/T product on ASIC in the literature.
  - We applied some scaling to convert 90nm results to 45nm, and to account for the extra area required by full SHA-3.

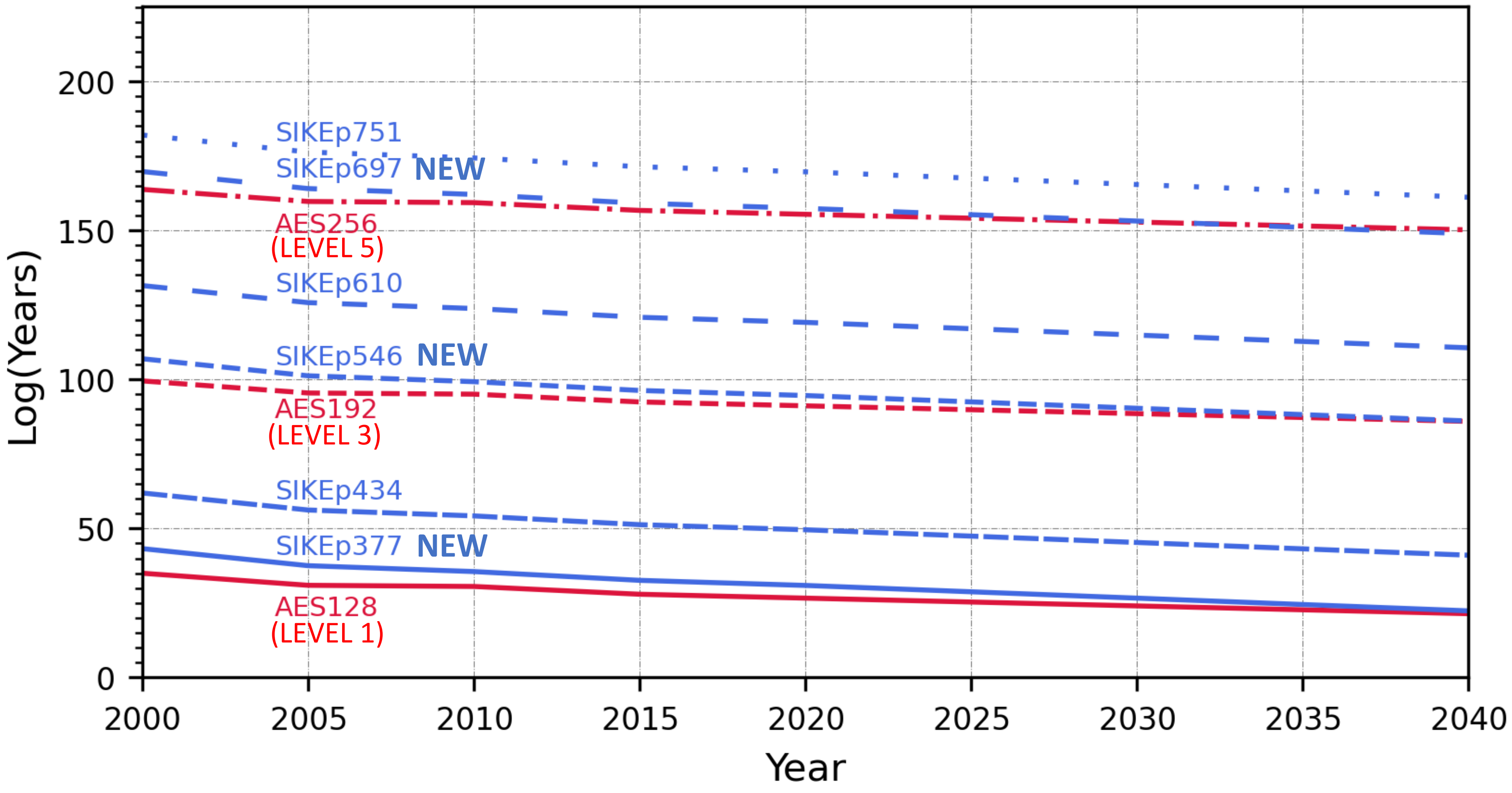
[1] A. Akin, A. Aysu, O. Can Ulusel, and E. Savas. "Efficient hardware implementations of high throughput SHA-3 candidates Keccak, Lua and Blue Midnight Wish for single- and multi-message hashing" (2010).

Analysis results

Analysis example: US$100 billion budget



Analysis example: US$100 billion budget



Implementation results

Round 3 vs. alternative parameters

- Performance on an x64 Intel Skylake CPU @3.4GHz.

Security level	Parameter set	Round 3		Parameter set	This work	
		pk	Encaps+Decaps ($\times 10^6$ cycles)		pk	Encaps+Decaps ($\times 10^6$ cycles)
1	**SIKEp434**	330 B	20.0	**SIKEp377**	288 B	14.5
2	**SIKEp503**	378 B	27.9	**--**	--	--
3	**SIKEp610**	462 B	54.7	**SIKEp546**	414 B	39.8
5	**SIKEp751**	564 B	84.6	**SIKEp697**	528 B	68.3

- E.g., SIKEp377 executes in **4.3 msec.**, $\sim 1.4 \times$ faster than SIKEp434 (Level 1).

The Cost to Break SIKE

A Comparative Hardware-Based Analysis with AES and SHA-3

Patrick Longa Wen Wang Jakub Szefer

https://eprint.iacr.org/2020/1457

https://github.com/microsoft/vOW4SIKE_on_HW