POST-QUANTUM SECURE CRYPTOGRAPHIC IMPLEMENTATIONS FOR EMBEDDED DEVICES

Joppe Bos

SEPTEMBER 2023

SECURE CONNECTIONS
FOR A SMARTER WORLD

PUBLIC

Agenda

- Who am I?
- Quantum Threat → Post-Quantum → New Standards
- Examples: Applied PQC Innovation
  - PQC Side-Channel Analysis
  - PQC Hardware Re-use
- PQC Use-Cases
  - Low-memory Dilithium
  - PQC in Automotive

Goal: Look at PQC from an industry perspective.

What research is important and needed?

SECURE CONNECTIONS FOR A SMARTER WORLD

OUR DIGITALLY ENHANCED WORLD IS EVOLVING TO ANTICIPATE AND AUTOMATE

NXP Semiconductors N.V. (NASDAQ: NXPI) is a global semiconductor company creating solutions that enable secure connections and infrastructure for a smarter world. NXP focuses on research, development and innovation in its target markets.

AUTOMOTIVE

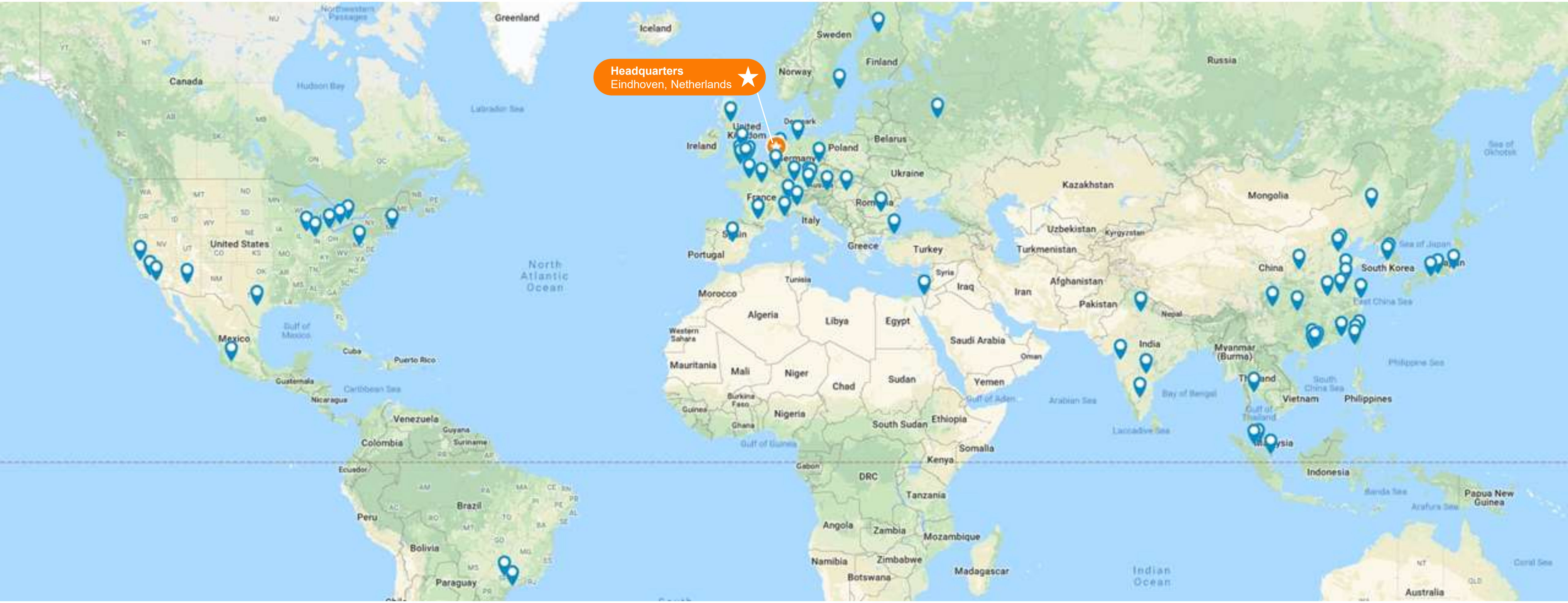
INDUSTRIAL & IOT

MOBILE

COMMUNICATION INFRASTRUCTURE

NXP LOCATIONS

~34,000 employees with operations in more than 30 countries



WHOAMI

Joppe W. Bos

Cryptographic Researcher and Technical Director at NXP Semiconductors

Secretary of the IACR (2017-2019, 2020-2022)

Editor of the Cryptology ePrint Archive (2019-today)

Editor-in-Chief of the IACR Communications in Cryptology

WHOAMI

- Cryptographic Researcher & Technical Director @ NXP
  - Competence Center Crypto & Security in Leuven, Belgium
  - Technical lead of the PQC project
  - Manager of the Crypto Concepts team
  - Head security + crypto funded projects & university relations

- Post-doc
  - Cryptography Research Group at Microsoft Research, Redmond, USA.
- PhD in Cryptology
  - EPFL, Lausanne, Switzerland
- Bachelor / Master in Computer Science
  - University of Amsterdam

BREAKING ECC

Main PhD project:

112-bit ECDLP solved using 224 PlayStation 3 game consoles.

Bos, Kaihara, Kleinjung, Lenstra, Montgomery: Solving a 112-bit Prime Elliptic Curve Discrete Logarithm Problem on Game Consoles using Sloppy Reduction. International Journal of Applied Cryptography, 2012.

QUANTUM THREAT

→

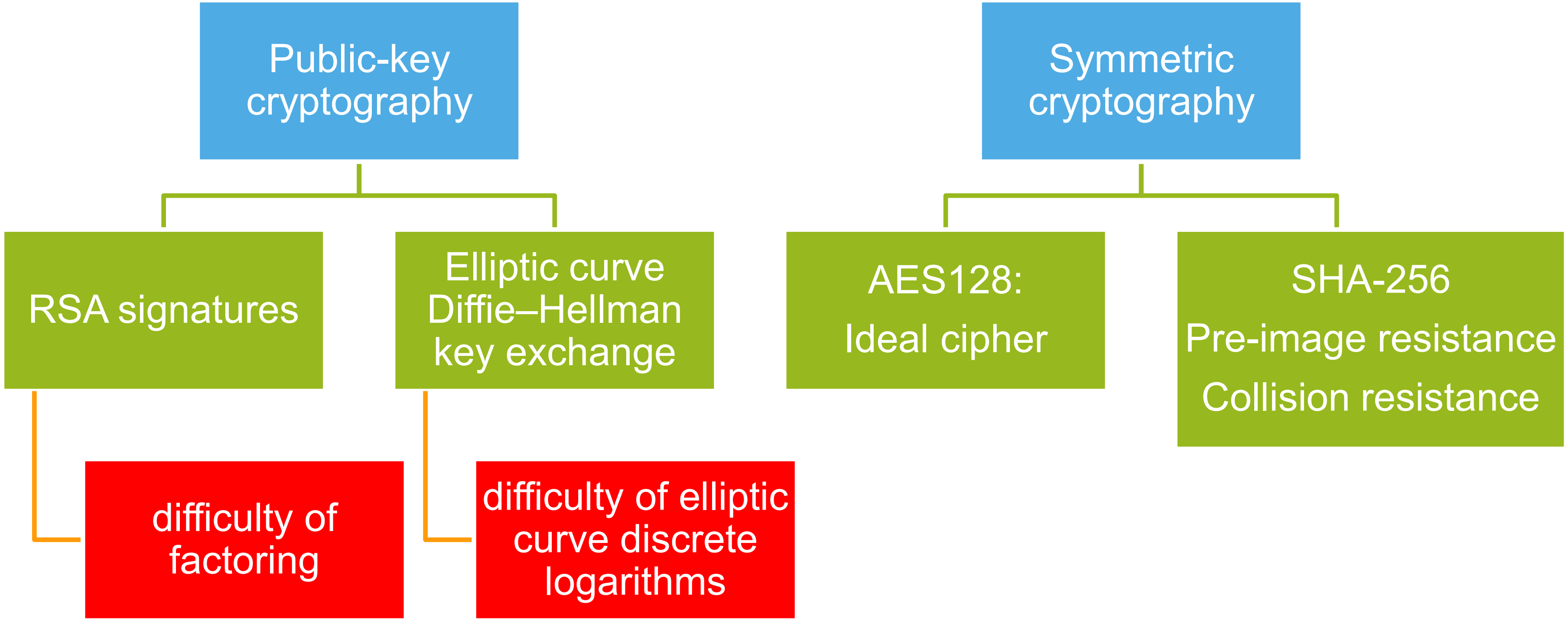
POST-QUANTUM

→

NEW STANDARDS

CONTEMPORARY CRYPTOGRAPHY

TLS-ECDHE-RSA-AES128-GCM-SHA256



Public-key cryptography

RSA signatures

difficulty of factoring

Elliptic curve Diffie–Hellman key exchange

difficulty of elliptic curve discrete logarithms

Symmetric cryptography

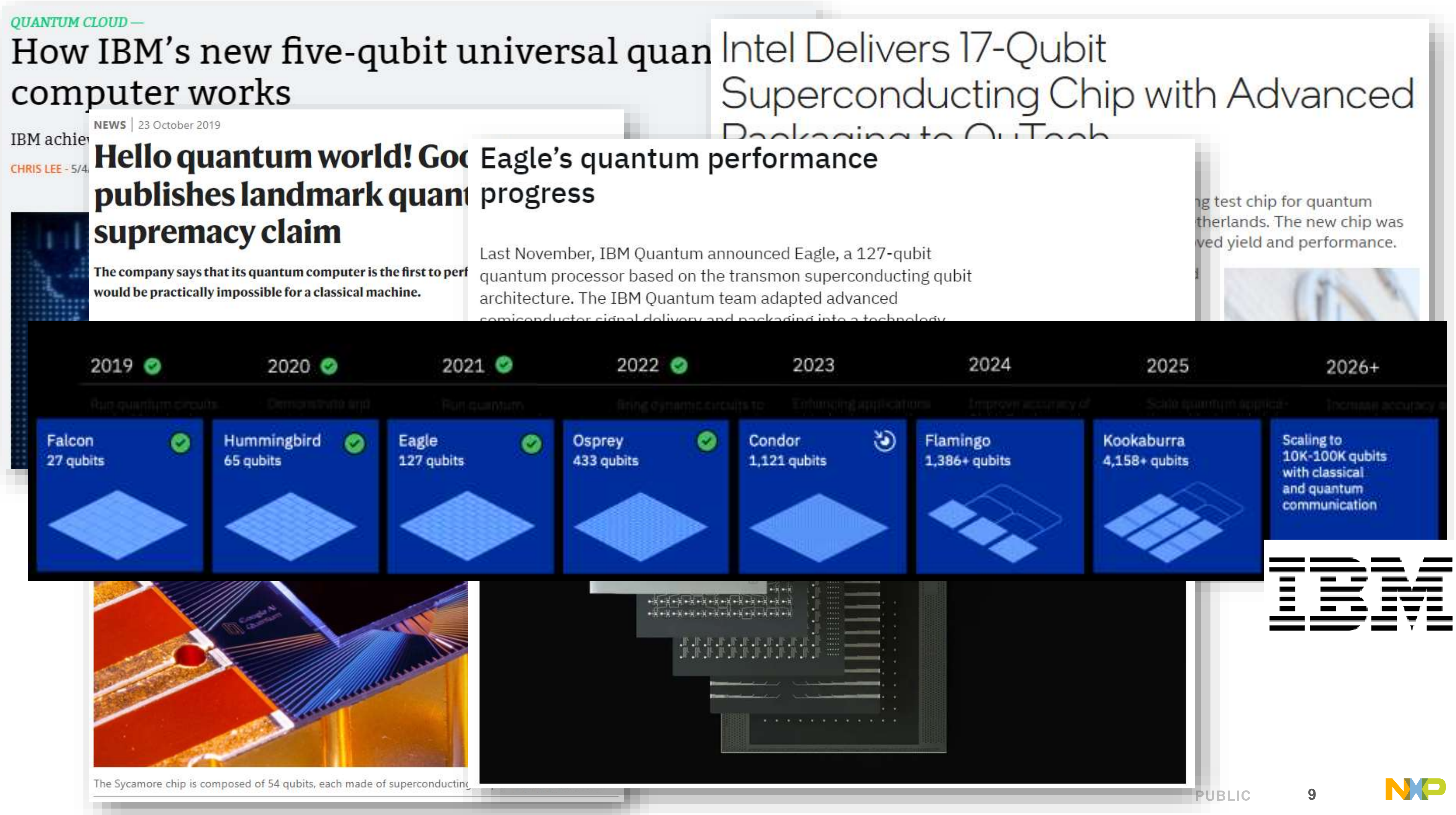
AES128:

Ideal cipher

SHA-256

Pre-image resistance

Collision resistance



QUANTUM CLOUD —

How IBM's new five-qubit universal quan computer works

IBM achie

CHRIS LEE - 5/4

NEWS | 23 October 2019

Hello quantum world! Goo publishes landmark quant supremacy claim

The company says that its quantum computer is the first to perf would be practically impossible for a classical machine.

The Sycamore chip is composed of 54 qubits, each made of superconducting

Intel Delivers 17-Qubit Superconducting Chip with Advanced Packaging to QuTech

ng test chip for quantum therlands. The new chip was ved yield and performance.

Eagle's quantum performance progress

Last November, IBM Quantum announced Eagle, a 127-qubit quantum processor based on the transmon superconducting qubit architecture. The IBM Quantum team adapted advanced semiconductor signal delivery and packaging into a technology

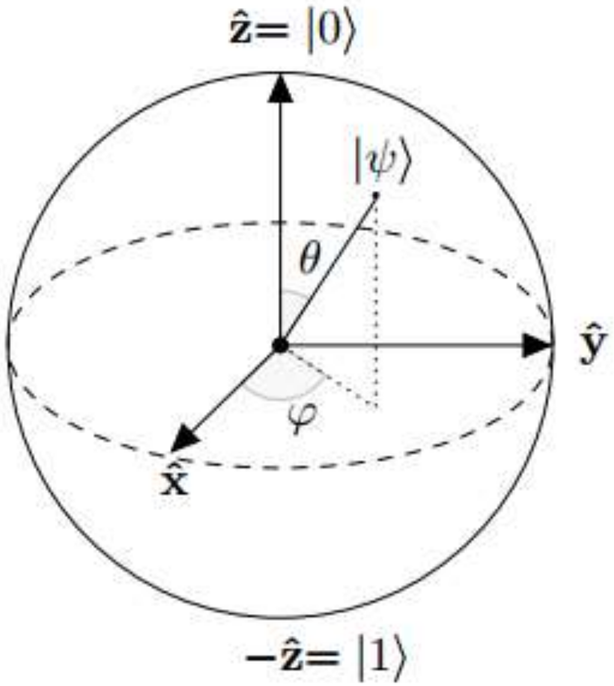
2019 ✓	2020 ✓	2021 ✓	2022 ✓	2023	2024	2025	2026+
Run quantum circuits	Demonstrate and	Run quantum	Bring dynamic circuits to	Enhancing applications	Improve accuracy of	Scale quantum applica-	Increase accuracy a
Falcon ✓ 27 qubits	Hummingbird ✓ 65 qubits	Eagle ✓ 127 qubits	Osprey ✓ 433 qubits	Condor 1,121 qubits	Flamingo 1,386+ qubits	Kookaburra 4,158+ qubits	Scaling to 10K-100K qubits with classical and quantum communication

IBM

QUANTUM COMPUTING

Computer systems and algorithms based on principles of quantum mechanics

- Superposition
- Interference
- Entanglement



- A classical bit can only be in the state corresponding to 0 or the state corresponding to 1
- A qubit may be in a superposition of both states
  → when measured it is always 0 or 1

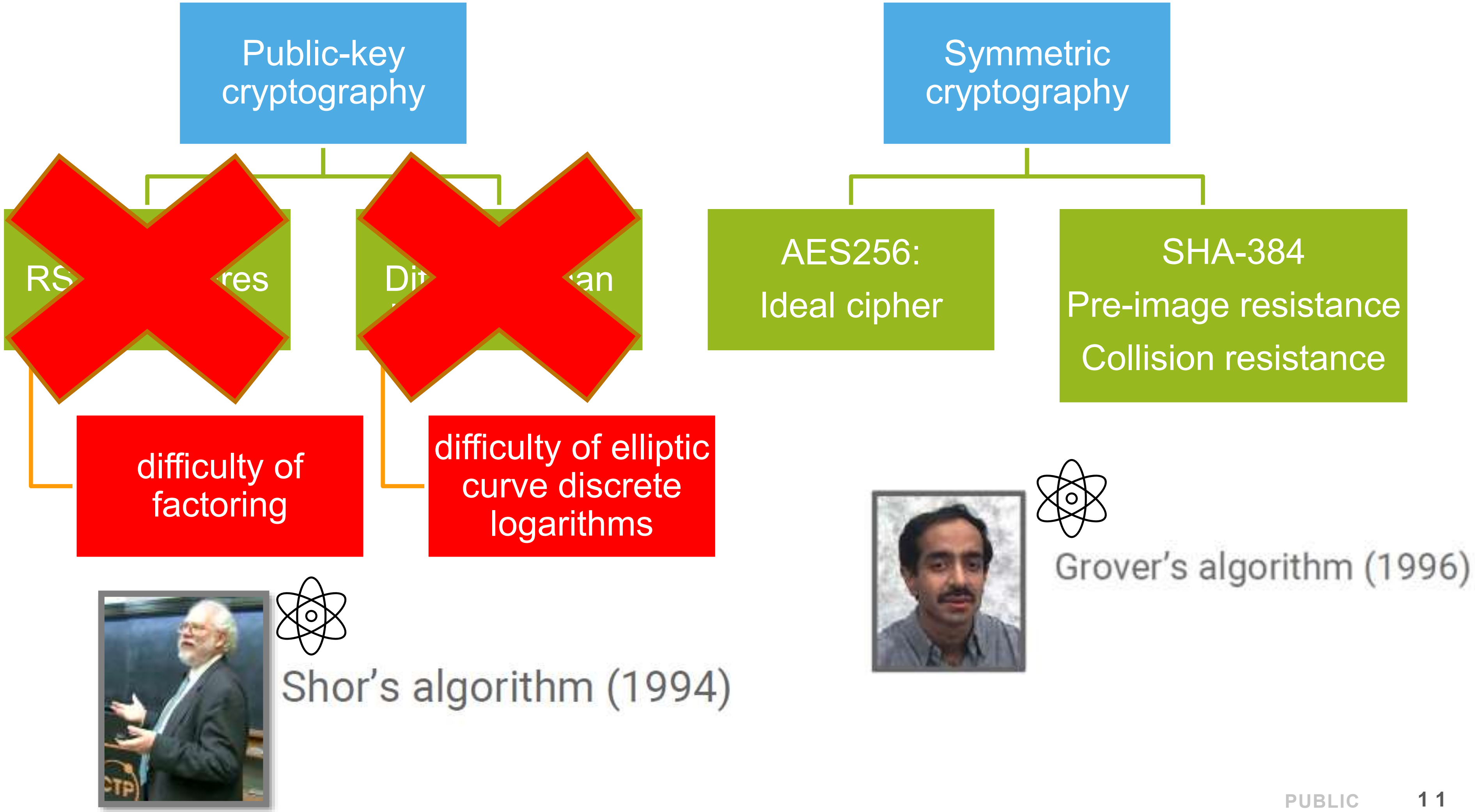
State-of-the-art.
IBM’s 127-Qubit Quantum Processor
Break RSA-3072:
~10,000 qubits are needed

Shor’s quantum algorithm (1994).
Polynomial time algorithm to factor integers.
Impact. If we assume the availability of a large quantum computer, then one can break RSA instantly.

CONTEMPORARY CRYPTOGRAPHY

TLS-~~ECDHE-RSA~~-AES256-GCM-SHA384

“Double” the key sizes



Public-key cryptography

RS res

Di an

difficulty of factoring

difficulty of elliptic curve discrete logarithms

Shor’s algorithm (1994)

Symmetric cryptography

AES256:
Ideal cipher

SHA-384
Pre-image resistance
Collision resistance

Grover’s algorithm (1996)

Quantum Potential to Destroy Security as we know it

Confidential email messages, private documents, and financial transactions
Secure today but could be compromised in the future, even if encrypted

Firmware update mechanisms in vehicles
Could be circumvented and allow dangerous modifications

Critical industrial and public service infrastructure (for healthcare, utilities, and transportation using internet and virtual private networks)
Could become exposed – potentially destabilize cities

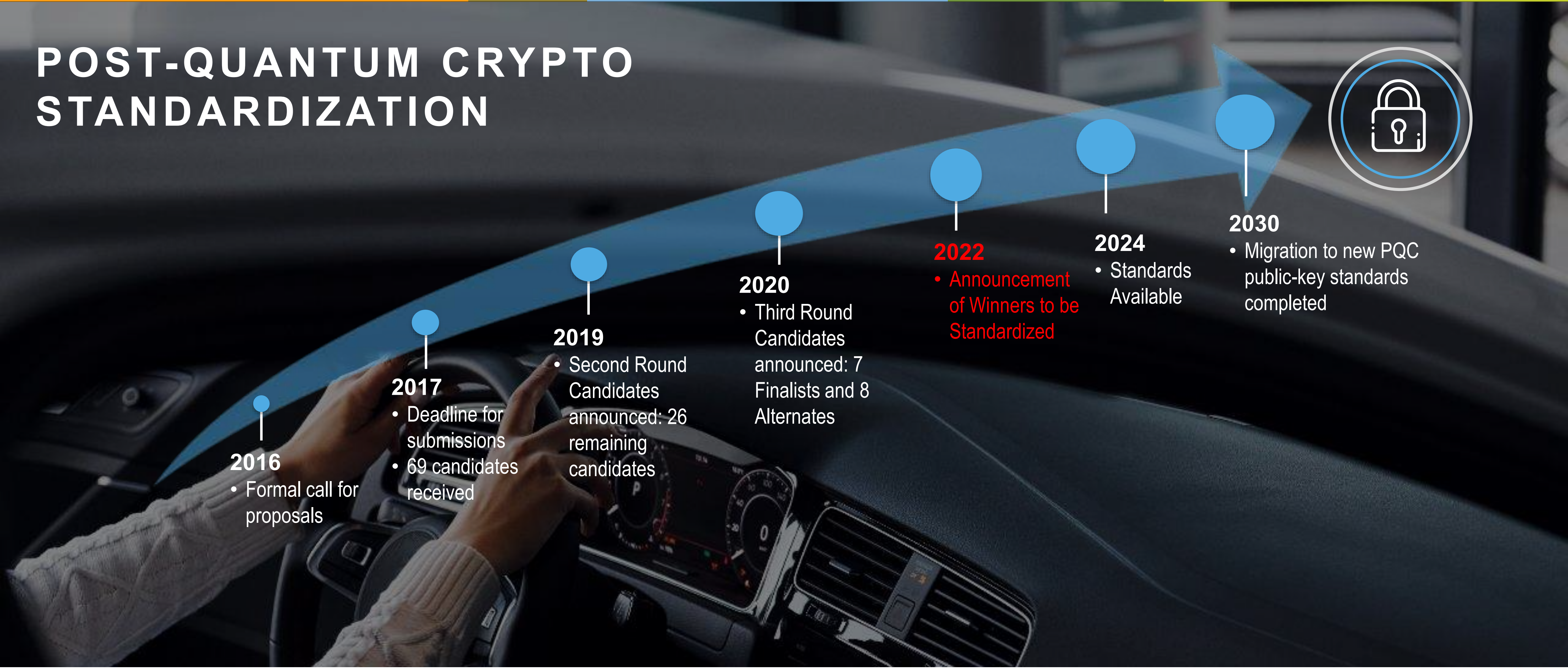
Audit trails and digitally signed documents associated with safety (auto certification and pharmaceutical authorizations)
Could be retrospectively modified

The integrity of blockchains
Could be retrospectively compromised - could include fraudulent manipulation of ledger and cryptocurrency transactions

POST-QUANTUM CRYPTO STANDARDS ARE COMING

IT DOESN’T MATTER IF YOU BELIEVE IN QUANTUM COMPUTERS OR NOT

POST-QUANTUM CRYPTO STANDARDIZATION



2016
- Formal call for proposals

2017
- Deadline for submissions
- 69 candidates received

2019
- Second Round Candidates announced: 26 remaining candidates

2020
- Third Round Candidates announced: 7 Finalists and 8 Alternates

2022
- Announcement of Winners to be Standardized

2024
- Standards Available

2030
- Migration to new PQC public-key standards completed

NIST

National Institute of Standards and Technology

HOW TO PREPARE FOR ~~HURRICANE~~ SEASON
Quantum

MAKE A PLAN

Airmen should create an emergency plan and/or checklist

- obtain supplies
- update personal documents
- secure household
- research evacuation options/routes
- update prescriptions

CREATE A GO-BAG

Prepare supplies ahead of a hurricane. These can include:

- Food/water
- Additional clothes
- Personal documents
- Travel supplies
- Prescriptions

KNOW YOUR WING GUIDANCE

Whether preparing for a hurricane or evacuating know your wing or installation's guidance. Routinely check for updates from leadership and maintain communication with your chain of command.

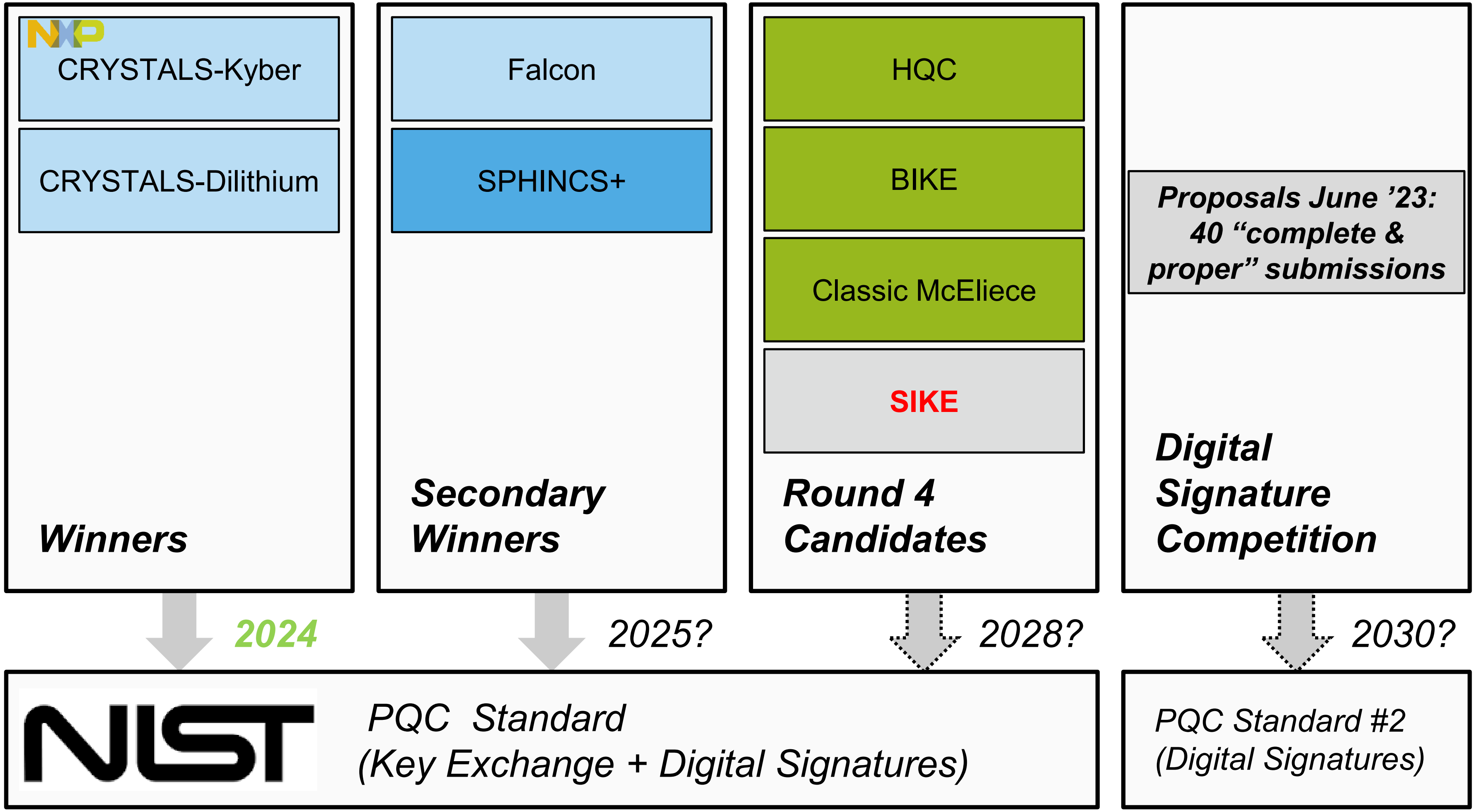
RECOGNIZE WARNINGS & ALERTS

Have several ways to receive alerts. Download real-time alert apps. Sign up for community alerts in your area and be aware of the Emergency Alert System (EAS) and Wireless Emergency Alert (WEA)- which requires no-sign up.

STAY SAFE

Practice good hygiene and safety measures during any part of a hurricane evacuation or impact. Keep family considerations in mind and don't be afraid to contact leadership for guidance.

PQC STANDARDS – NIST



CRYSTALS-Kyber

CRYSTALS-Dilithium

Winners

Falcon

SPHINCS+

Secondary Winners

HQC

BIKE

Classic McEliece

SIKE

Round 4 Candidates

Proposals June '23: 40 "complete & proper" submissions

Digital Signature Competition

2024

2025?

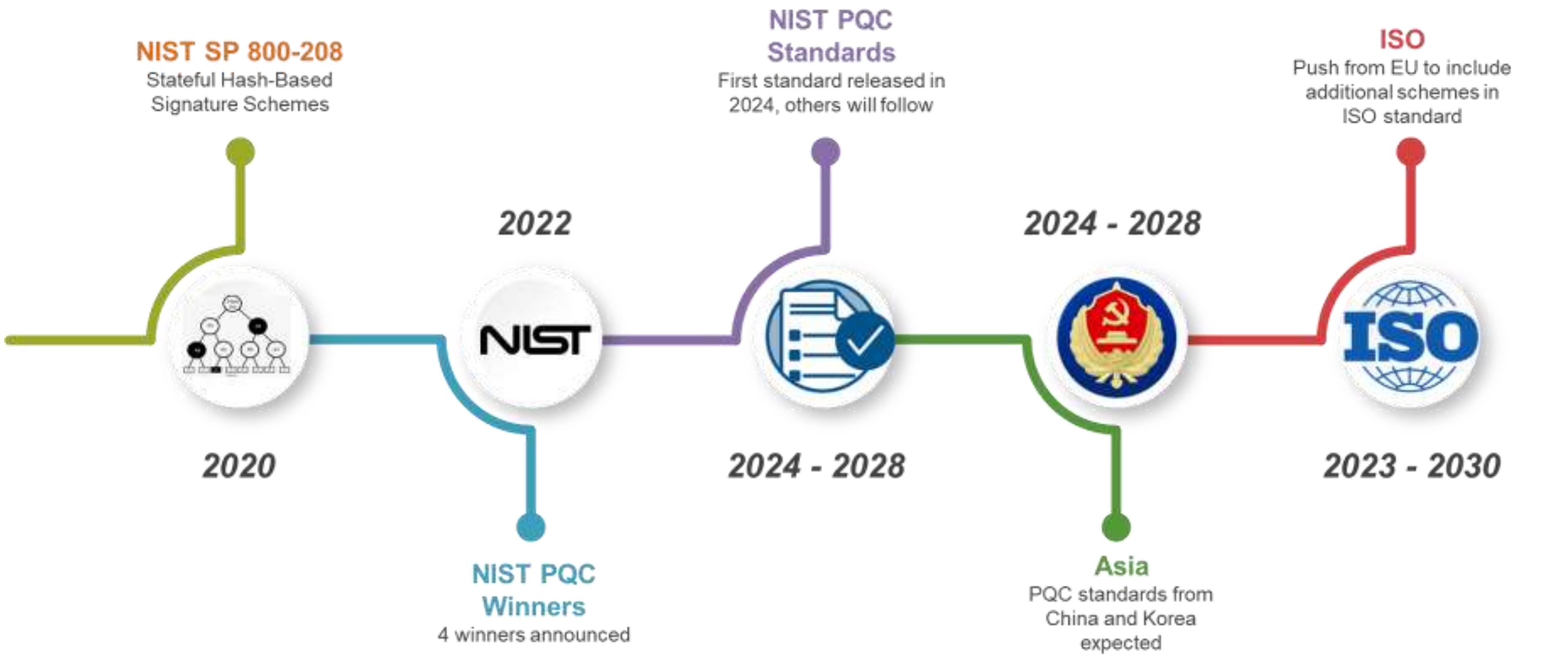
2028?

2030?

PQC Standard (Key Exchange + Digital Signatures)

PQC Standard #2 (Digital Signatures)

Color key: Mathematical approach | Lattice | Hash | Code



NIST SP 800-208
Stateful Hash-Based
Signature Schemes

2020

2022

NIST PQC Winners
4 winners announced

NIST PQC Standards
First standard released in
2024, others will follow

2024 - 2028

2024 - 2028

Asia
PQC standards from
China and Korea
expected

ISO
Push from EU to include
additional schemes in
ISO standard

2023 - 2030

National Standards

- **USA.** NIST announces standards release of 4 PQC schemes (‘24 – ’25). Additional standards to follow.
- **EU.** Push from BSI (help from NXP) for adding schemes to international standard. April ‘23: ISO to amend ISO/IEC 18033-2.
- **ASIA.** Selection of new schemes ongoing in both China/Korea.

Protocol Standards

- **IETF:** TLS, OpenPGP, hybrid keys, key serialization, encoding for signatures
- ISO/TC 68/SC 2/WG 11 (Encryption algorithms used in banking applications)
- ISO/IEC JTC1/SC 17/WG 4 (Cards and security devices for personal identification)

PQC MIGRATION GUIDANCE BY GOVERNMENTS

USA (NIST/NSA)

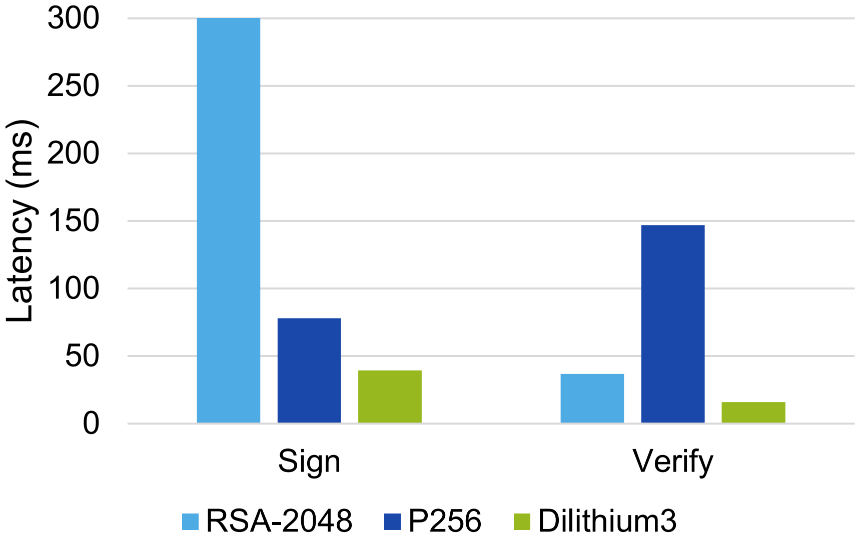
- NIST/NSA recommendation available
  - Commercial National Security Algorithm Suite 2.0
- PQC FW signature recommended for new products after 2025
- PQC transition complete by 2030 using SW update

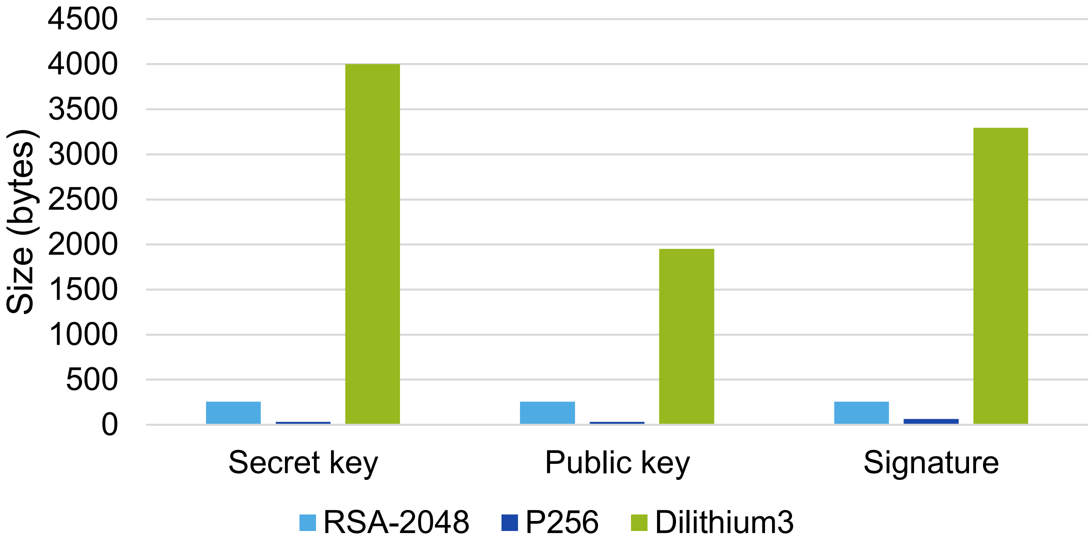
Germany (BSI)

- BSI first recommendation (English)
- BSI considerations (German)
- Expectation is that beginning of 2030s, a relevant quantum computer is available to be a threat for high-secure applications
- Quantum security: considers both PQC + QKD

France (ANSSI)

- PQC for security products “as soon as possible” when long-lasting (until 2030) protection is required
- Others to migrate to classic-PQC hybrid in 2025 – 2030
- Switch to PQC-only expected by 2030





DILITHIUM IMPACT

- Measurements on Cortex-M4 from pqm4 framework
- Functional implementation only (not hardened)
- Large trade-offs between stack and efficiency
- **80 ~ 90 percent** of run-time in SHA-3

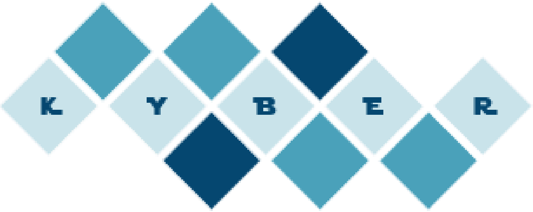
PQC SIGNATURE MIGRATION (EMBEDDED PERSPECTIVE)

Algorithm (Level 3)	PQ Secure?	Standard?	Efficient Signing?	Stateful?	Efficient Verify?	Need hybrid?	PK (Bytes)	Sig (Bytes)
ECC	No	FIPS 186	Yes	No	Yes	N/A	32 B	64 B
Dilithium	Yes	PQC (2024)	Yes	No	Yes	Yes	1952 B	3293 B
Falcon (L5)	Yes	PQC (2024)	No	No	Yes	Yes	1793 B	1280 B
SPHINCS+	Yes	PQC (2024)	No	No	Yes	No	48 B	16224 B
LMS / XMSS	Yes	SP 800-208	Yes?	Yes	Yes	No	60 B	1744 B

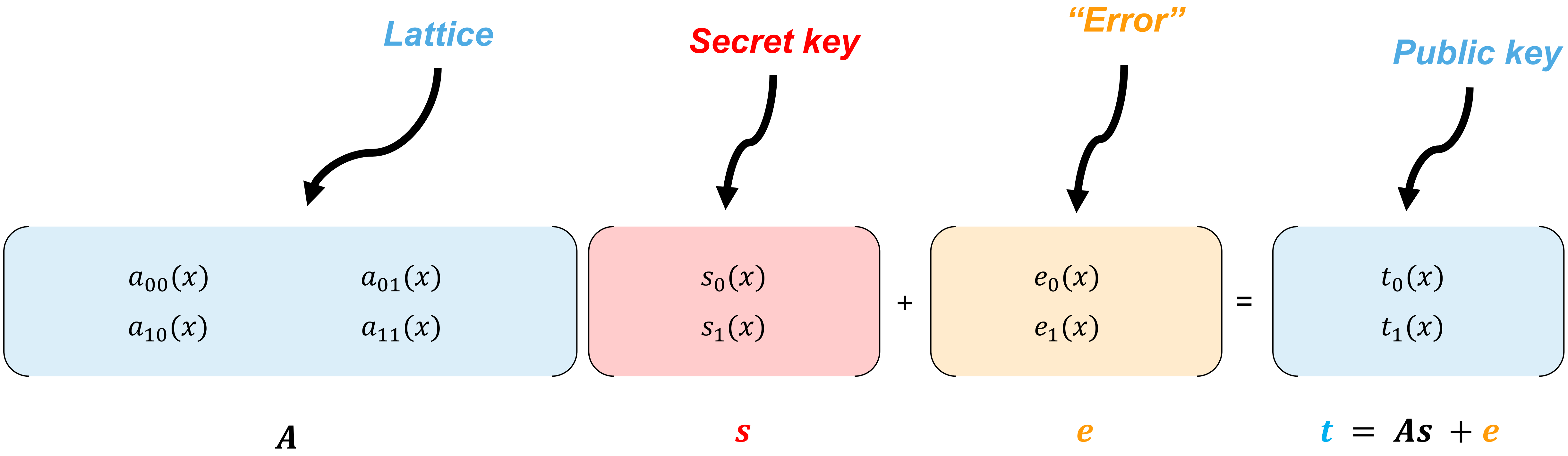
MODULE LWE 101

CRYPTOGRAPHIC SUITE FOR ALGEBRAIC LATTICES (CRYSTALS)

- The Cryptographic Suite for Algebraic Lattices (CRYSTALS) encompasses
  - Kyber – Key Encapsulation Mechanism (KEM)
  - Dilithium – Digital Signatures
- **Theory**: same building blocks
  - Module Learning with Errors
  - Number-Theoretic Transformations



MODULE (RING) LEARNING WITH ERRORS



Lattice

Secret key

“Error”

Public key

$$\begin{pmatrix} a_{00}(x) & a_{01}(x) \\ a_{10}(x) & a_{11}(x) \end{pmatrix} \begin{pmatrix} s_0(x) \\ s_1(x) \end{pmatrix} + \begin{pmatrix} e_0(x) \\ e_1(x) \end{pmatrix} = \begin{pmatrix} t_0(x) \\ t_1(x) \end{pmatrix}$$

$\boldsymbol{A}$ $\boldsymbol{s}$ $\boldsymbol{e}$ $\boldsymbol{t} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}$

Given **blue**, find **red** or **yellow**

PUBLIC-KEY ENCRYPTION (DLOG DIFFIE-HELLMAN)

Exchange

Generate keypair $(r, u = rP)$

Generate shared secret $\kappa = rt$

Key generation

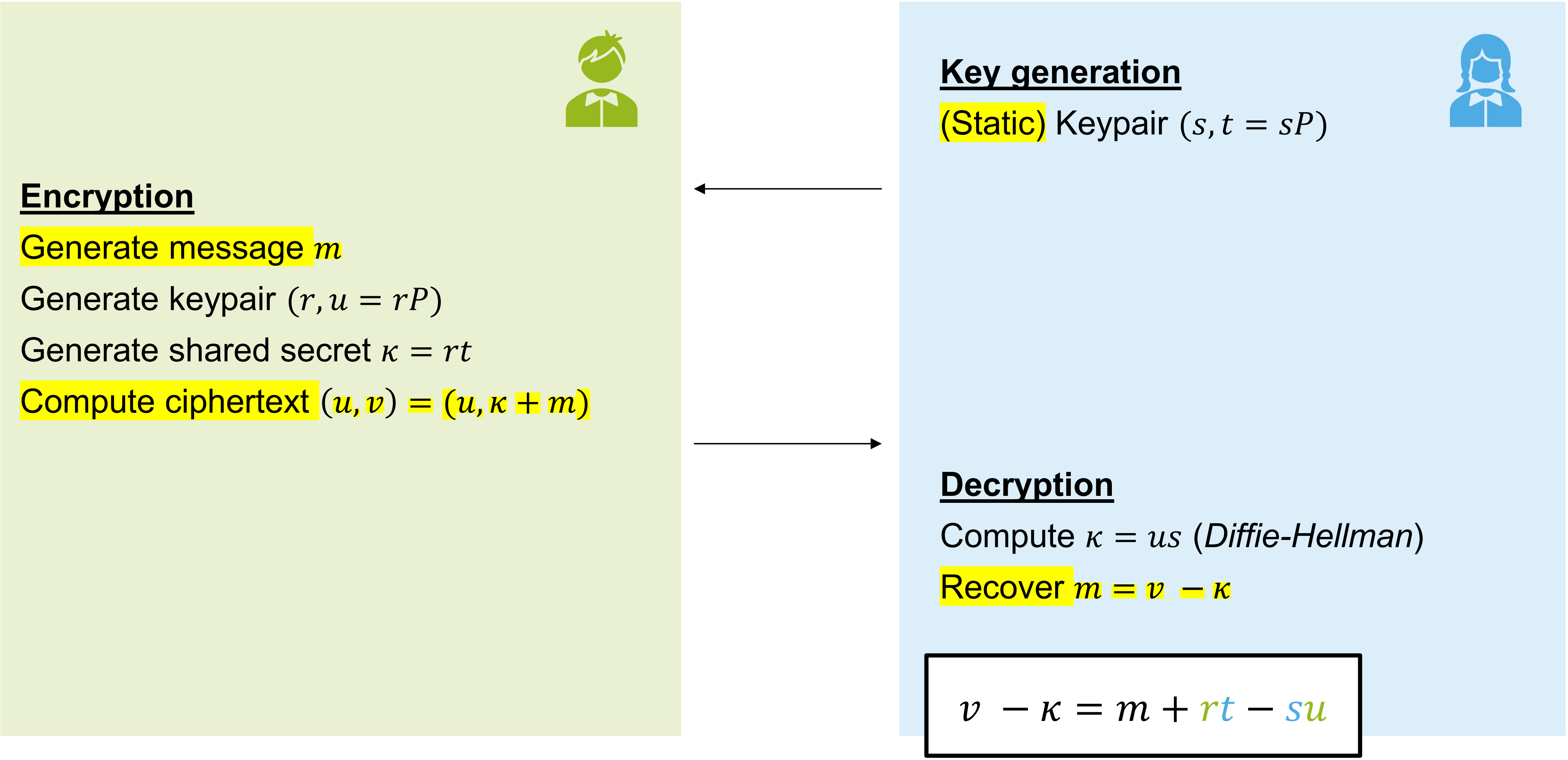
Keypair $(s, t = sP)$

Exchange

Compute $\kappa = us$ (*Diffie-Hellman*)

$$\boxed{rt = r(sP) = s(rP) = su}$$

PUBLIC-KEY ENCRYPTION (DLOG DIFFIE-HELLMAN + EL GAMAL)



Key generation

(Static) Keypair $(s, t = sP)$

Encryption

Generate message m

Generate keypair $(r, u = rP)$

Generate shared secret $\kappa = rt$

Compute ciphertext $(u, v) = (u, \kappa + m)$

Decryption

Compute $\kappa = us$ (*Diffie-Hellman*)

Recover $m = v - \kappa$

$$\boxed{v - \kappa = m + rt - su}$$

PUBLIC-KEY ENCRYPTION (“APPROXIMATE” EL GAMAL)

Key generation

(Static) Keypair $(s, t = As + e)$

Encryption

Generate message m

Generate keypair $(r, u = rA + e_1)$

Generate shared secret $\kappa = rt$

Compute ciphertext $(u, v) = (u, \kappa + m + e_2)$

Decryption

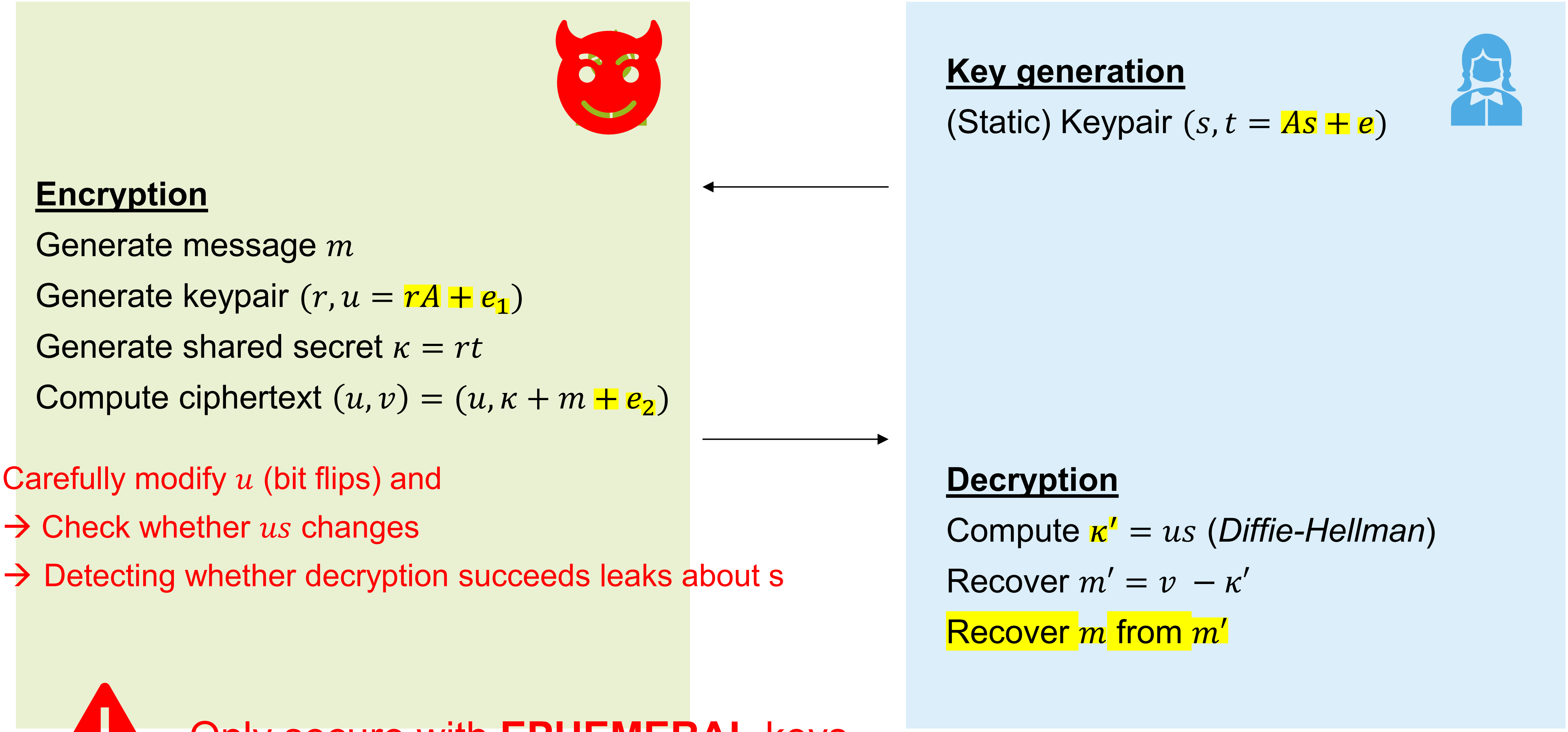
Compute $\kappa' = us$ (*Diffie-Hellman*)

Recover $m' = v - \kappa'$

Recover m from m'

$$\boxed{v - \kappa' = m + e_2 - e^T r - s^T e_1}$$

PUBLIC-KEY ENCRYPTION (LATTICE-BASED, IND-CPA)



Key generation

(Static) Keypair $(s, t = As + e)$

Encryption

Generate message m

Generate keypair $(r, u = rA + e_1)$

Generate shared secret $\kappa = rt$

Compute ciphertext $(u, v) = (u, \kappa + m + e_2)$

Decryption

Compute $\kappa' = us$ (*Diffie-Hellman*)

Recover $m' = v - \kappa'$

Recover m from m'

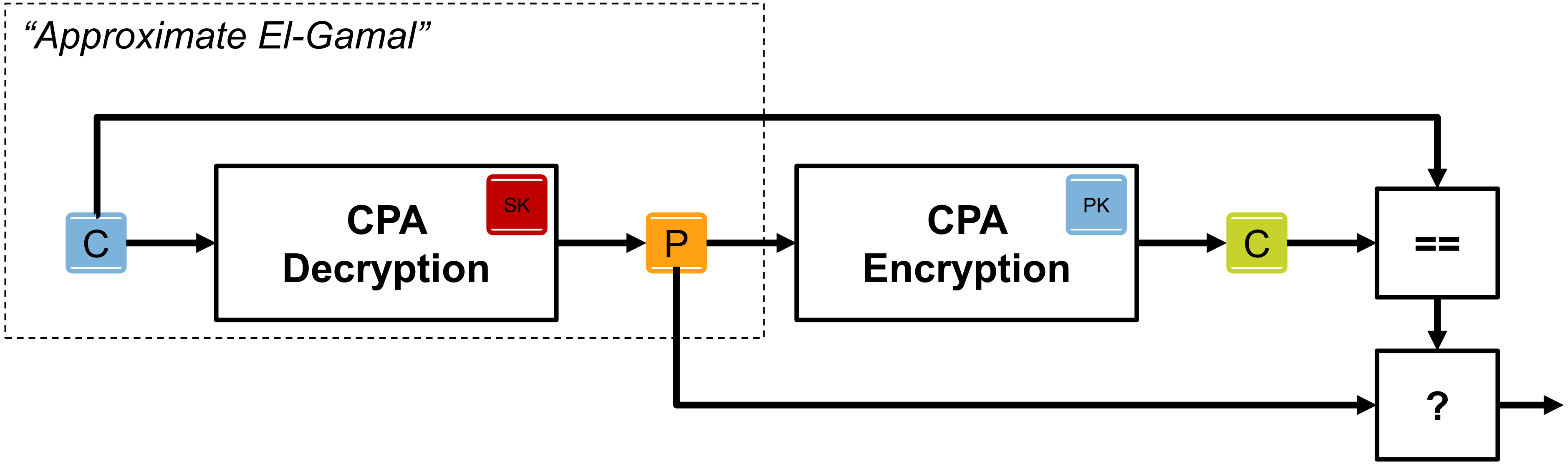
Carefully modify u (bit flips) and

→ Check whether us changes

→ Detecting whether decryption succeeds leaks about s

Only secure with **EPHEMERAL** keys

FUJISAKI OKAMOTO TRANSFORM

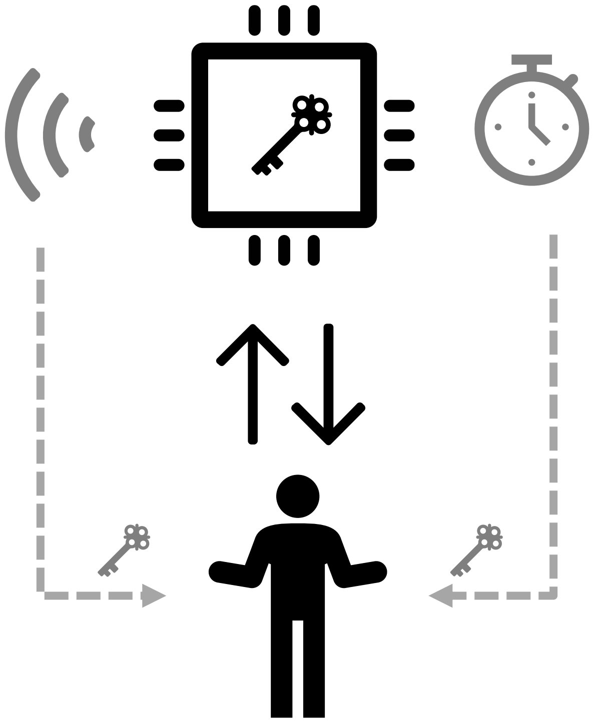


Transform a scheme which achieves IND-CPA ("chosen plaintext attack") security to reach IND-CCA ("indistinguishability against chosen-ciphertext attacks") security

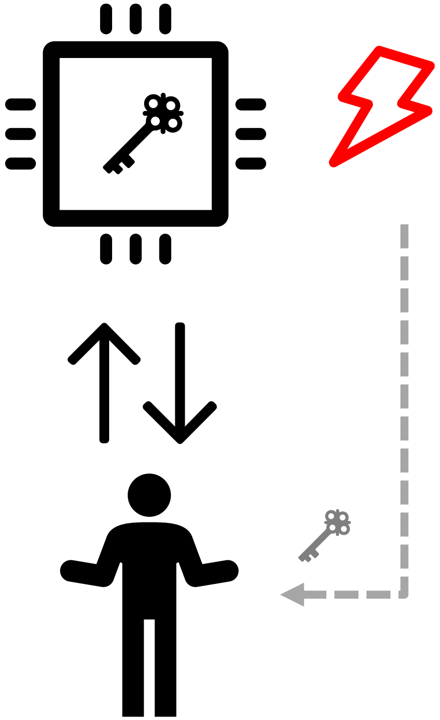
- Fujisaki, E. and Okamoto T., Secure integration of asymmetric and symmetric encryption schemes, CRYPTO 1999 and JoC 2013

PQC & SCA

EMBEDDED CRYPTOGRAPHY AND IMPLEMENTATION ATTACKS



Side-Channel Attacks (SCA)



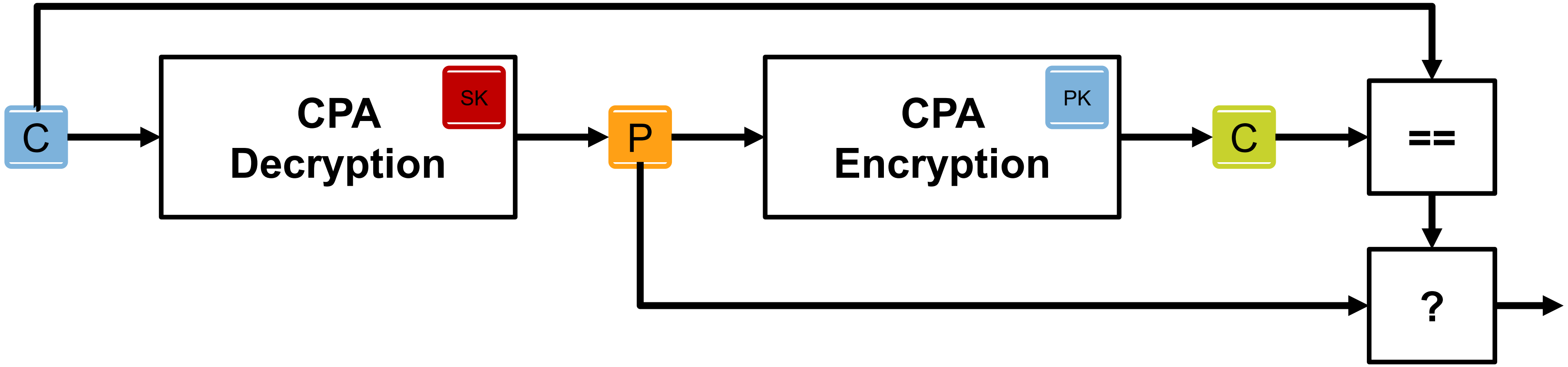
Fault Attacks (FA)

CHALLENGES IN THE EMBEDDED WORLD

Attacks		Current Cryptography		Countermeasures
Deep understanding in both academia and industry.	✓	AES 3DES DSA … ECDSA RSA ECC	✓	Practically secure and certified implementations.
Active research area resulting in increasingly powerful attacks.	?	NTRU HBS Saber … Kyber Dilithium	?	Early stage of academic research. Limited industrial results.

What does it mean to secure PQC implementations in “practice”?

FUJISAKI OKAMOTO TRANSFORM



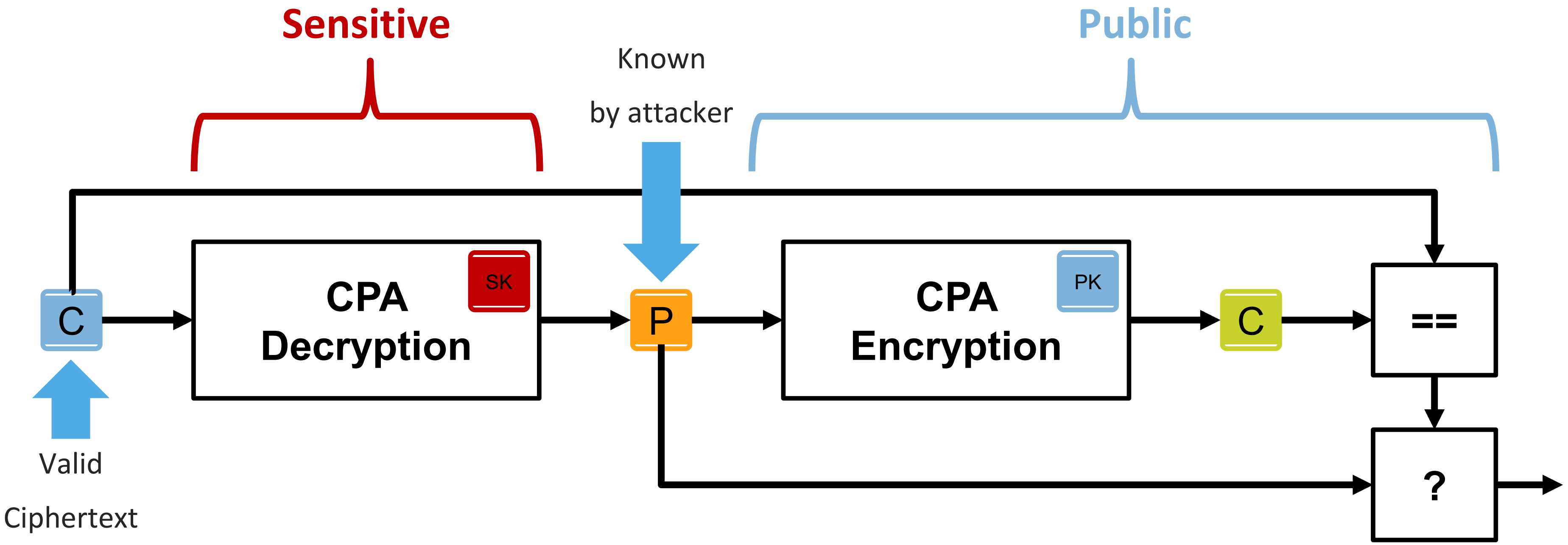
Transform a scheme which achieves IND-CPA ("chosen plaintext attack") security to reach IND-CCA ("indistinguishability against chosen-ciphertext attacks") security

- Fujisaki, E. and Okamoto T., Secure integration of asymmetric and symmetric encryption schemes, CRYPTO 1999 and JoC 2013

THE SCA PROBLEM OF THE FO-TRANSFORM

Attack 1: Chosen Plaintext

- Attacker inputs only valid ciphertexts
- Attack focuses on **CPA Decryption**, everything after (and including) P is public
- Only need to protect **CPA Decryption**

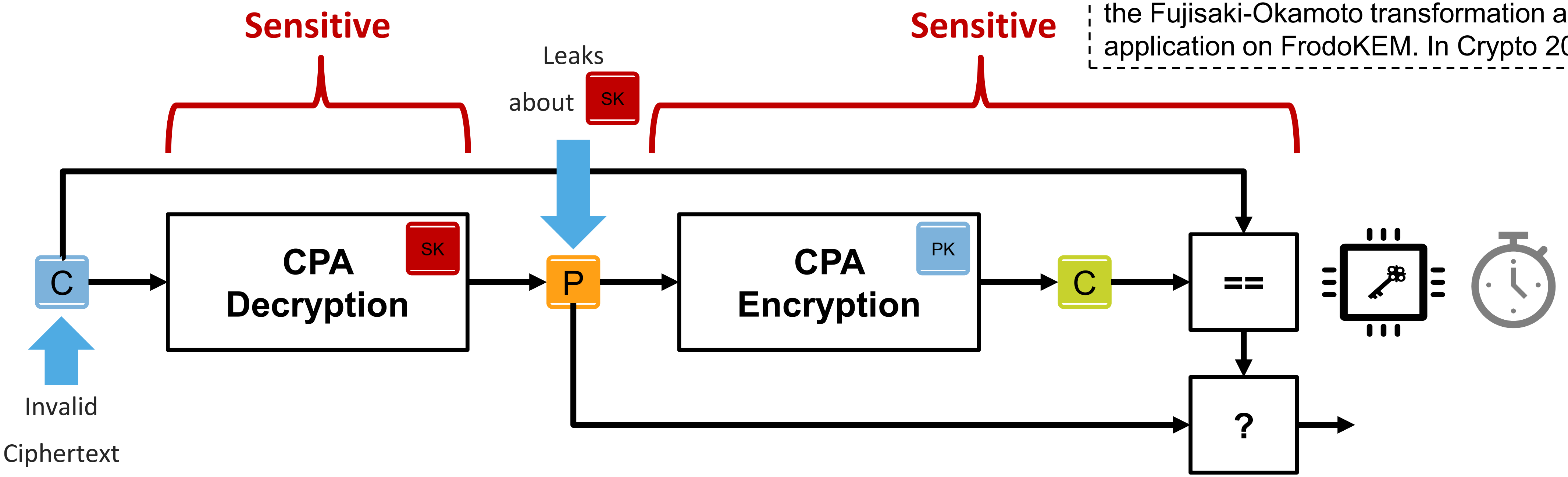


THE SCA PROBLEM OF THE FO-TRANSFORM

Attack 2: Chosen Ciphertext

- Attacker inputs specially-crafted invalid ciphertexts
- Attack focuses on **CPA Decryption** + everything after (and including) P is potentially sensitive
- Potentially all (or most) modules need to be hardened

Guo, Johansson, Nilsson. A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM. In Crypto 2020.



THE SCA PROBLEM OF THE FO-TRANSFORM

? **Why is it bad?**

☑ Millions of Points of Interest (PoI)

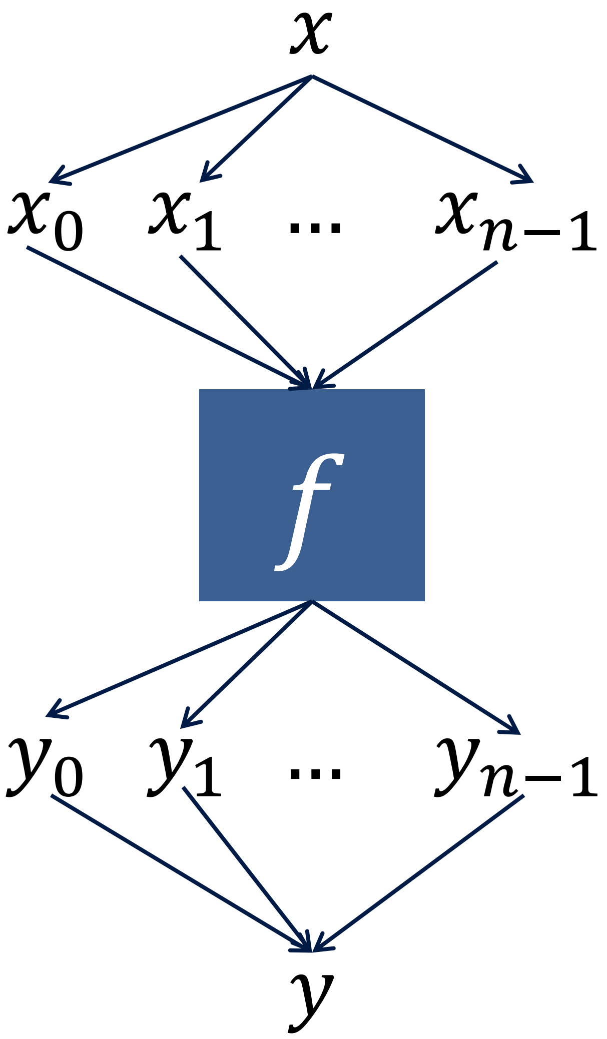
☑ Low number of leakage classes (worst case = 2)

☑ Easy to build templates

SIDE-CHANNEL ATTACKS ON THE FO-TRANSFORM

- *Ravi et al. "Generic Side-channel attacks on CCA-secure lattice-based PKE and KEMs" TCHES 2020*
- *Xu et al. "Magnifying Side-Channel Leakage of Lattice-Based Cryptosystems with Chosen Ciphertexts: The Case Study of Kyber" IEEE Transactions on Computers, 2021*
- *Qin et al. "A Systematic Approach and Analysis of Key Mismatch Attacks on Lattice-Based NIST Candidate KEMs" ASIACRYPT 2021*
- *Ngo et al. "A Side-Channel Attack on a Masked IND-CCA Secure Saber KEM Implementation" TCHES 2021*
- *Ravi et al. "Will You Cross the Threshold for Me? - Generic Side-Channel Assisted Chosen-Ciphertext Attacks on NTRU-based KEMs" TCHES 2022*
- *Ueno et al. "Curse of Re-encryption: A Generic Power/EM Analysis on Post-Quantum KEMs" TCHES 2022*
- *Shen et al. "Find the Bad Apples: An efficient method for perfect key recovery under imperfect SCA oracles – A case study of Kyber" IACR ePrint archive 2022*
- *Ngo et al. "Side-Channel Attacks on Lattice-Based KEMs Are Not Prevented by Higher-Order Masking" IACR ePrint archive 2022*
- *Rajedran et al. "Pushing the Limits of Generic Side-Channel Attacks on LWE-based KEMs - Parallel PC Oracle Attacks on Kyber KEM and Beyond" IACR ePrint archive 2022*
- …

MASKING AGAINST SIDE-CHANNEL ATTACKS



- Encode sensitive variables into shares
- Compute securely on shares
- Decode at end to recover result

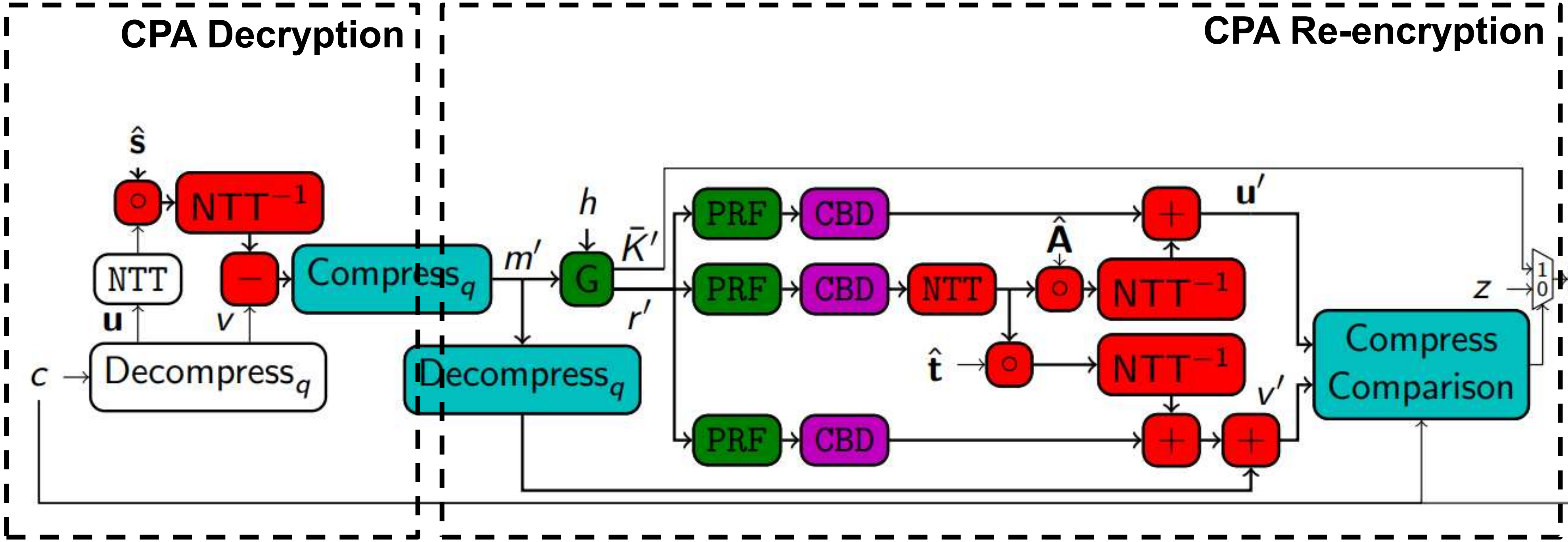
Masking if implemented **correctly** increases the attack complexity **exponentially** in the number of shares.

(assuming sufficient noise)

$x = x_0 + x_1 \bmod q$ (***arithmetic*** *masking*)

$x = x_0 \oplus x_1 \oplus x_2 \oplus x_3$ (***Boolean*** *masking*)

MASKING KYBER



Poly. arithmetic (🟥):

- Arith. masking.
- Linear overheads.

Hash functions (🟩):

- Boolean masking.
- Quadratic overheads.

Poly. sampl. (🟪) & compress. (🟦):

- Boolean & arith. masking.
- Quadratic overheads.

Bronchain, O., Cassiers, G. “Bitslicing Arithmetic/Boolean Masking Conversions for Fun and Profit with Application to Lattice-Based KEMs“. TCHES 2022.

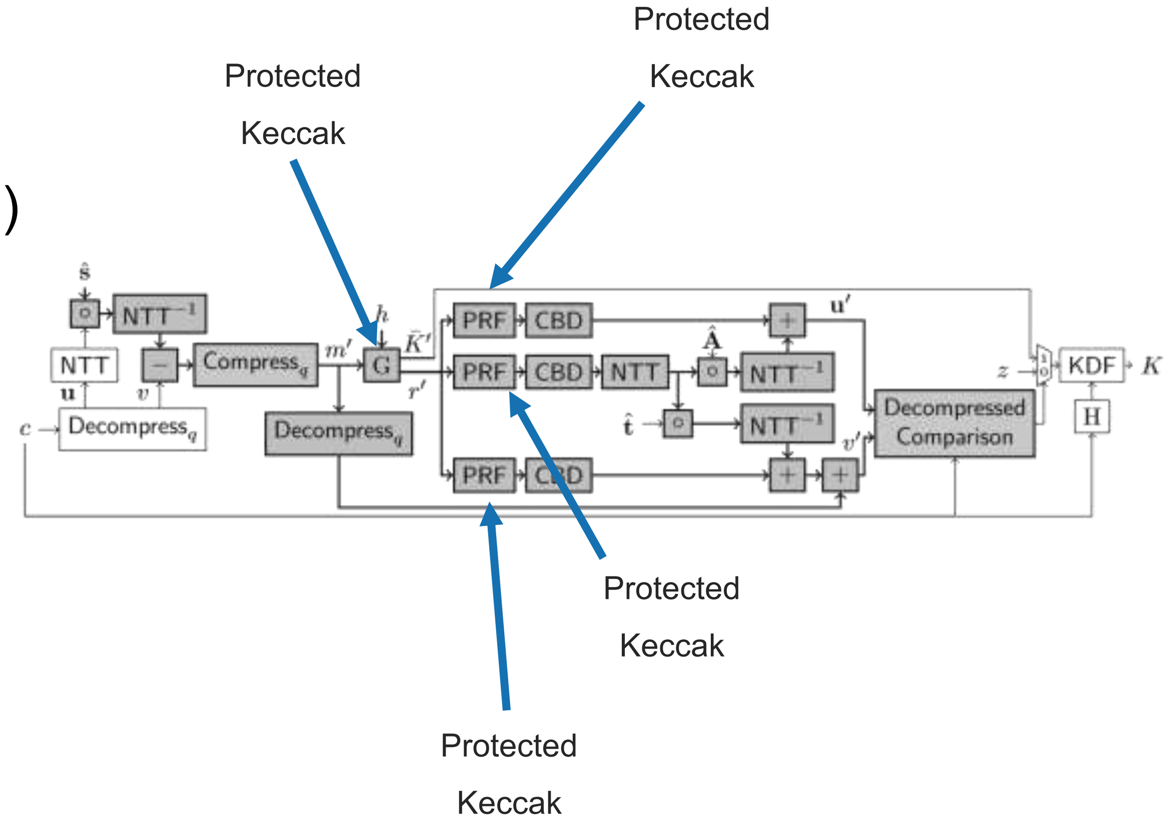
MASKING KYBER

What is the bottleneck for masking Kyber?

Latest Performance Numbers from *[BG22]*:

- Bitsliced masked Kyber (pure SW, ARM Cortex-M4)
- Performance values for 3 shares:

Masked Decapsulation	**16.7 M Cycles (100%)**
Keccak	**7.22 M Cycles (43%)**
B2A Conversion	5.02 M Cycles (30%)
Rest	4.46 M Cycles (27%)



Protected Keccak

Protected Keccak

Protected Keccak

Protected Keccak

[BG22] Bronchain, O., Cassiers, G. “Bitslicing Arithmetic/Boolean Masking Conversions for Fun and Profit with Application to Lattice-Based KEMs“. TCHES 2022.

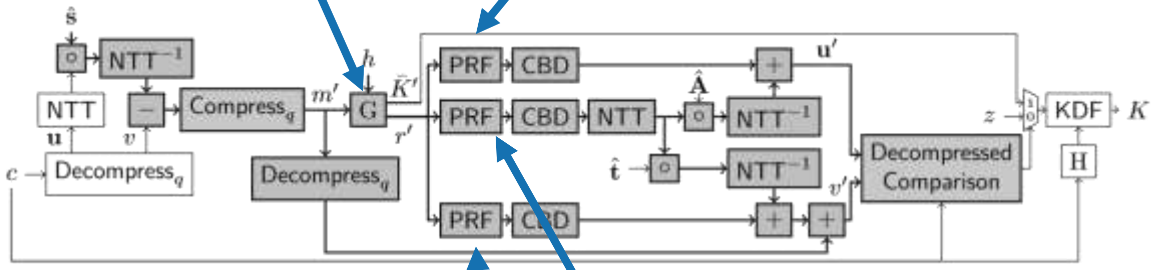
MASKING KYBER

? What is the bottleneck for masking Kyber?

Latest Performance Numbers from *[BG22]*:

- Bitsliced masked Kyber (pure SW, ARM Cortex-M4)
- Performance values for 3 shares:

Masked Decapsulation	**16.7 M Cycles (100%)**
Keccak	**7.22 M Cycles (43%)**
B2A Conversion	5.02 M Cycles (30%)
Rest	4.46 M Cycles (27%)



Protected Keccak

Protected Keccak

Protected Keccak

Protected Keccak

Most of the protected Keccak calls are in the re-encryption.

[BG22] Bronchain, O., Cassiers, G. “Bitslicing Arithmetic/Boolean Masking Conversions for Fun and Profit with Application to Lattice-Based KEMs“. TCHES 2022.

A CLOSER LOOK AT THE MASKED DECAPSULATION

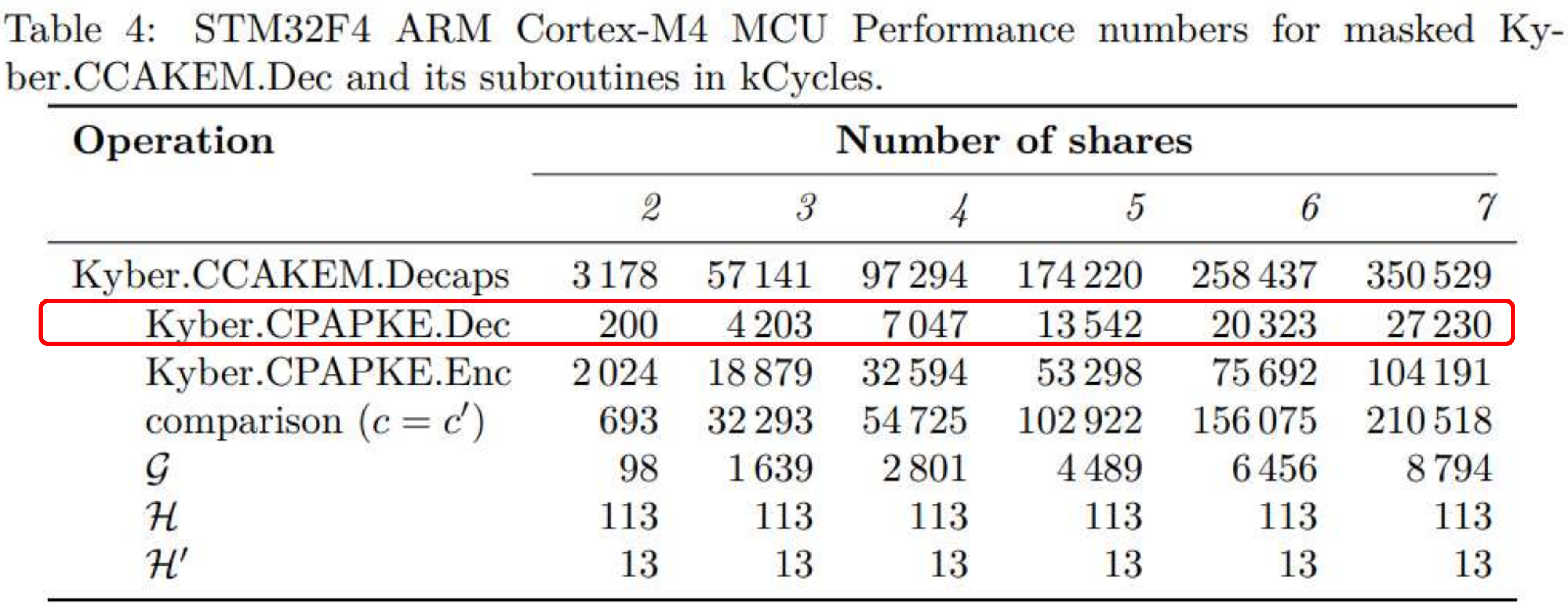


Table 4: STM32F4 ARM Cortex-M4 MCU Performance numbers for masked Kyber.CCAKEM.Dec and its subroutines in kCycles.

Operation	Number of shares					
	2	*3*	*4*	*5*	*6*	*7*
Kyber.CCAKEM.Decaps	3 178	57 141	97 294	174 220	258 437	350 529
Kyber.CPAPKE.Dec	200	4 203	7 047	13 542	20 323	27 230
Kyber.CPAPKE.Enc	2 024	18 879	32 594	53 298	75 692	104 191
comparison $(c = c')$	693	32 293	54 725	102 922	156 075	210 518
$\mathcal{G}$	98	1 639	2 801	4 489	6 456	8 794
$\mathcal{H}$	113	113	113	113	113	113
$\mathcal{H}'$	13	13	13	13	13	13

- Masked decryption is <10% of the cost of masked decapsulation
- Cost of masked decapsulation is dominated by the masked FO

Azouaoui, M., Kuzovkova, Y., Schneider, T., van Vredendaal, C. “Post-Quantum Authenticated Encryption against Chosen-Ciphertext Side-Channel Attacks“. TCHES 2022.

A VERY SIMPLE IDEA

Replace expensive FO by a signature verification of the ciphertext.

Signature verification only uses public data and does not require SCA protection.

→ Never decrypt untrusted ciphertexts.

– Based on the *Encrypt-then-Sign* ($\mathcal{E}t\mathcal{S}$) paradigm

– CCA security shown in *[ADR02]* in the outsider security model

– Post-quantum CCA security shown in *[CPPS20]*

- Y. Zheng. *Signcryption and its applications in efficient public key solutions*. ISW 1997.
- Azouaoui, M., Kuzovkova, Y., Schneider, T., van Vredendaal, C. *Post-Quantum Authenticated Encryption against Chosen-Ciphertext Side-Channel Attacks*. TCHES 2022.
- An, JH., Dodis, Y., Rabin, R. *On the Security of Joint Signature and Encryption*. EUROCRYPT 2002.
- Chatterjee, S., Pandit, T., Puria, SKP., Shah, A. *Signcryption in a Quantum World*. IACR ePrint Arch., 2020.

A VERY SIMPLE IDEA

Replace expensive FO by a signature verification of the ciphertext.

Signature verification only uses public data and does not require SCA protection.

→ Never decrypt untrusted ciphertexts.

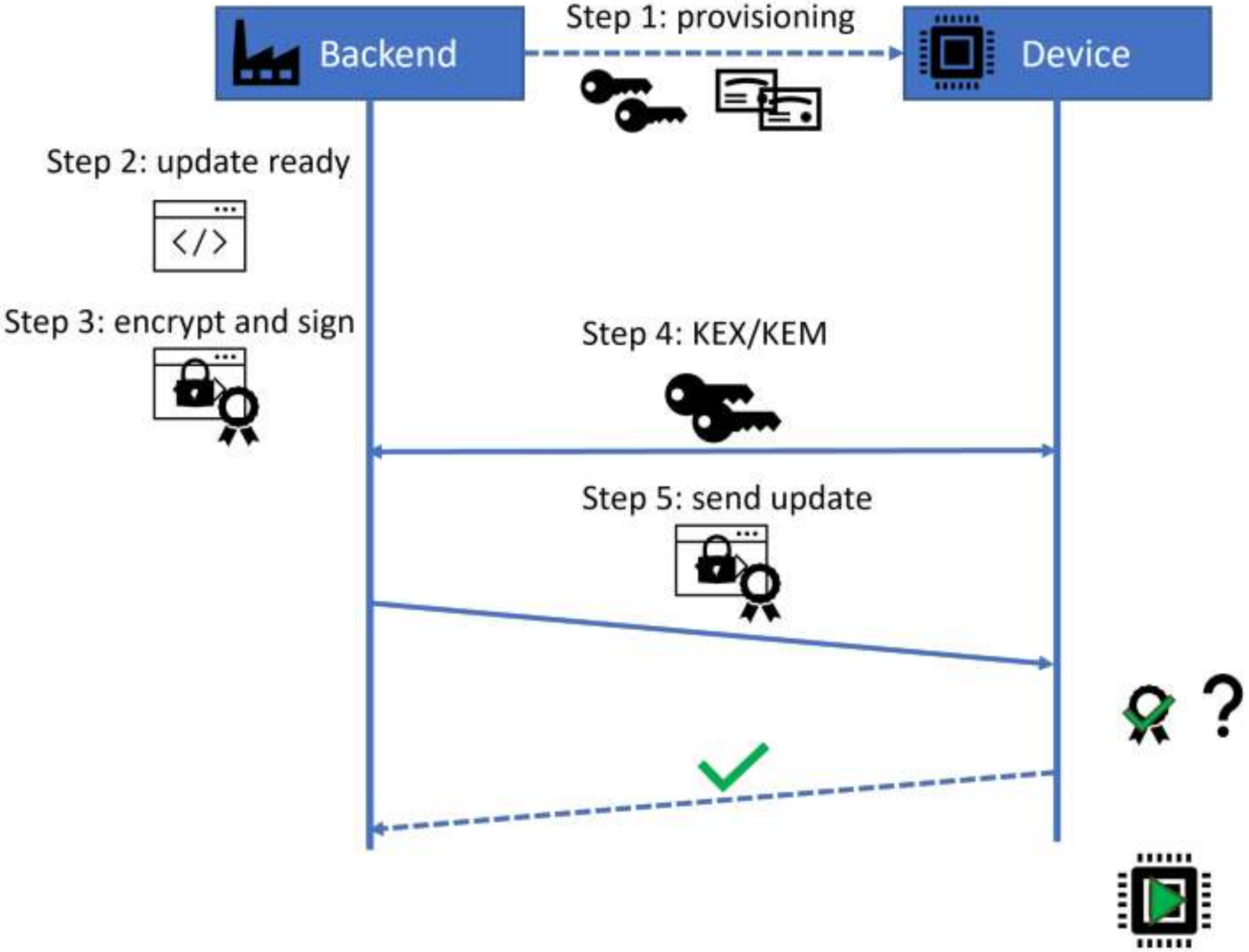
Adversary has only access to public material.

It is neither the sender nor the receiver.

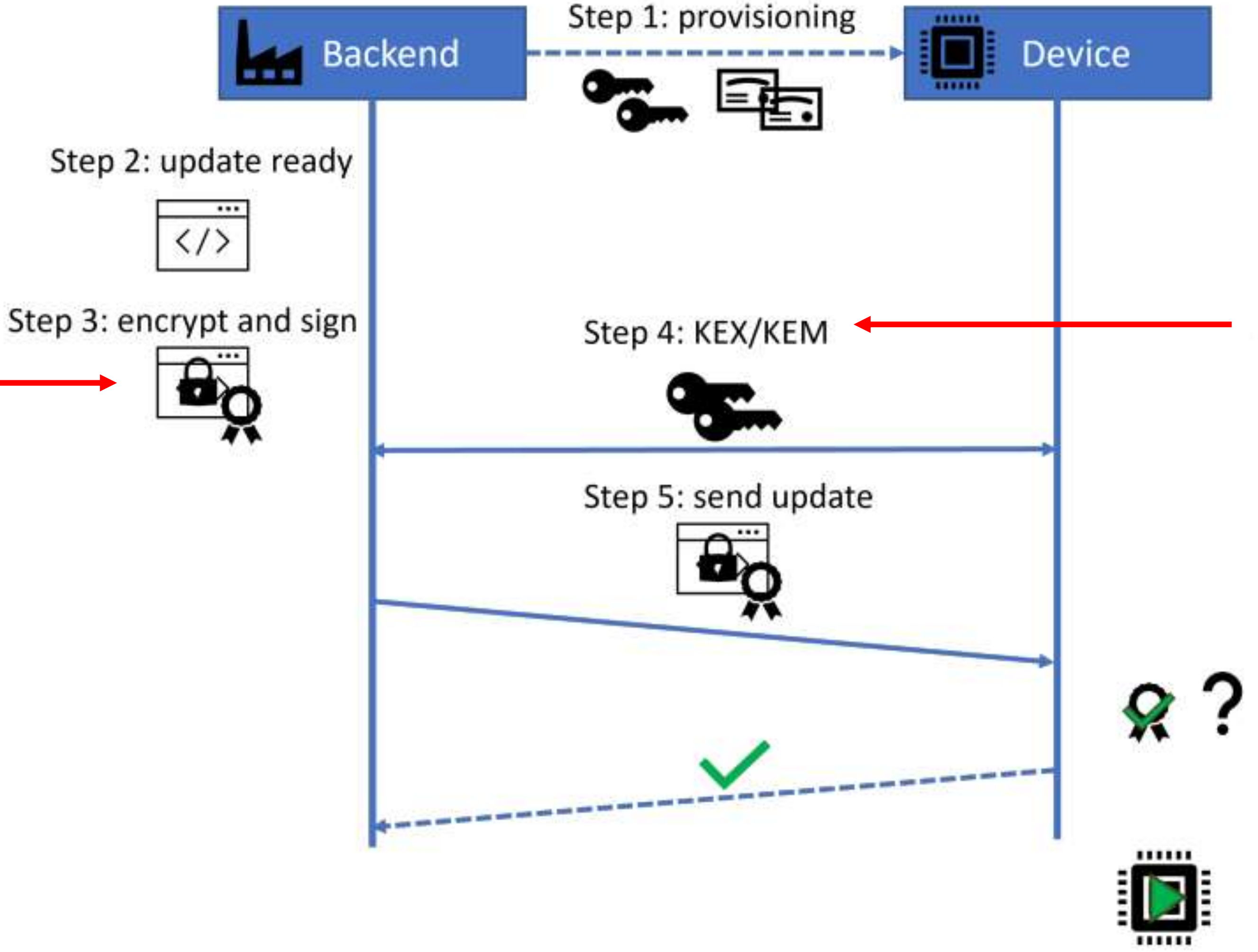
- Based on the *Encrypt-then-Sign* ($\mathcal{E}t\mathcal{S}$) paradigm
- CCA security shown in *[ADR02]* in the outsider security model
- Post-quantum CCA security shown in *[CPPS20]*

- Y. Zheng. *Signcryption and its applications in efficient public key solutions*. ISW 1997.
- Azouaoui, M., Kuzovkova, Y., Schneider, T., van Vredendaal, C. *Post-Quantum Authenticated Encryption against Chosen-Ciphertext Side-Channel Attacks*. TCHES 2022.
- An, JH., Dodis, Y., Rabin, R. *On the Security of Joint Signature and Encryption*. EUROCRYPT 2002.
- Chatterjee, S., Pandit, T., Puria, SKP., Shah, A. *Signcryption in a Quantum World*. IACR ePrint Arch., 2020.

THE $\mathcal{E}t\mathcal{S}$ KEM FOR SECURE UPDATE MECHANISM



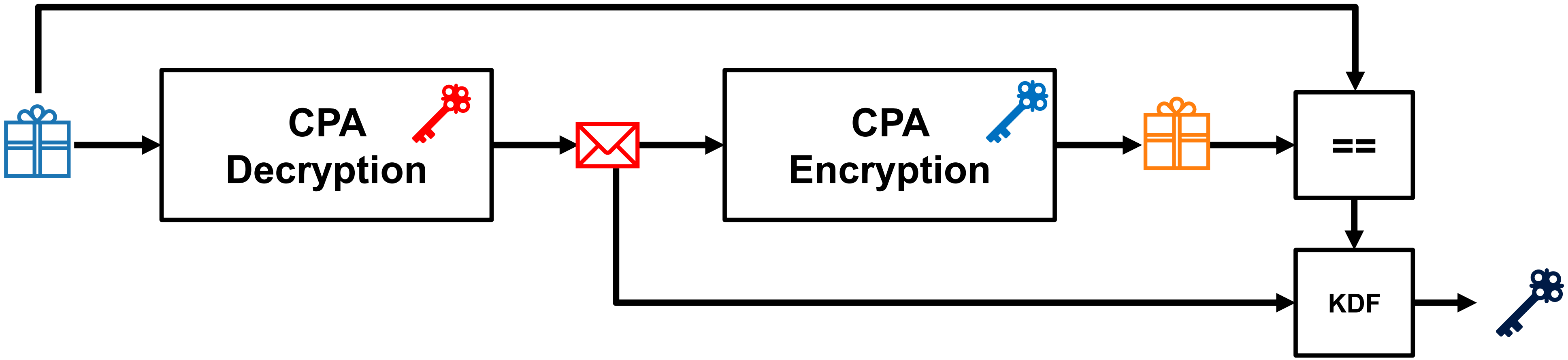
THE $\mathcal{E}t\mathcal{S}$ KEM FOR SECURE UPDATE MECHANISM



Add ciphertext before signing

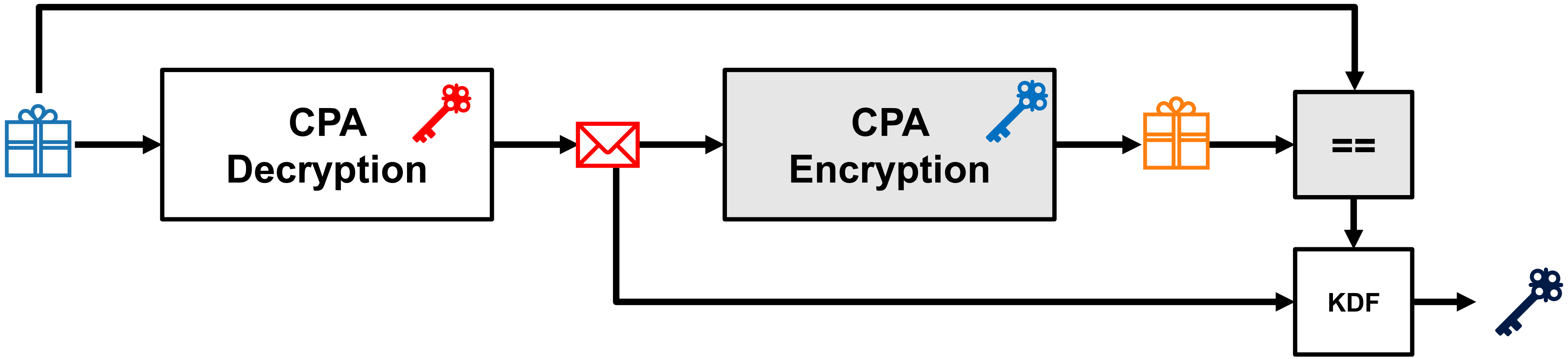
$\mathcal{E}t\mathcal{S}$ **KEM**

THE $\mathcal{E}t\mathcal{S}$ KEM VS. THE FO KEM



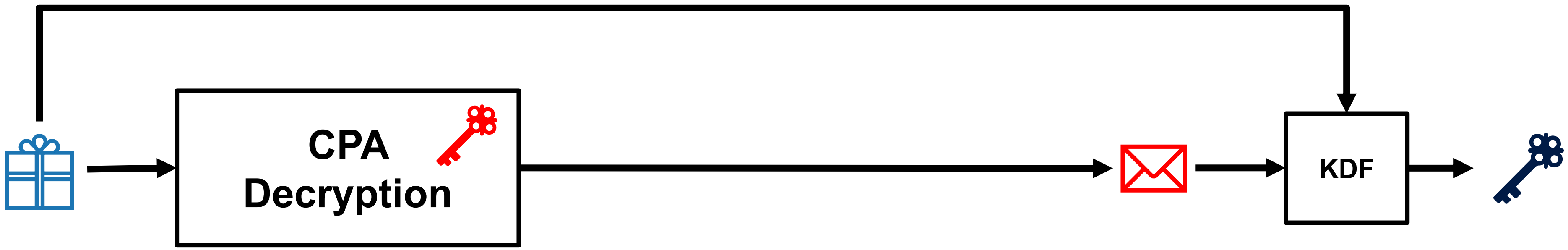
- CCA FO KEM Decapsulation -

THE $\mathcal{E}t\mathcal{S}$ KEM VS. THE FO KEM



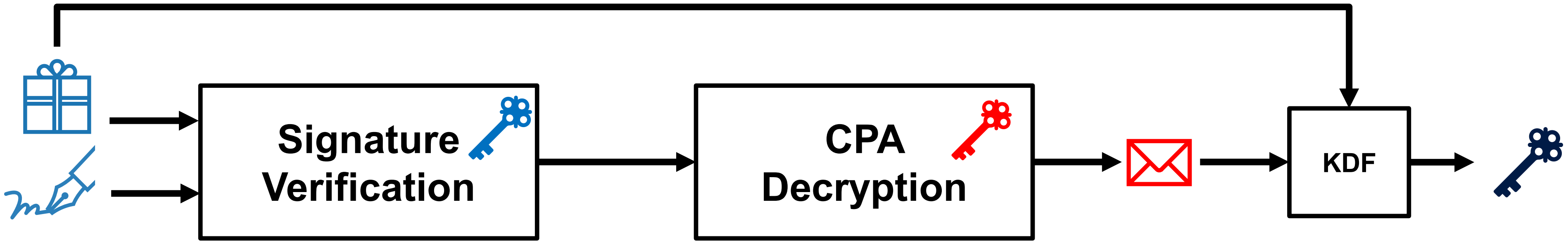
- CCA FO KEM Decapsulation -

THE $\mathcal{E}t\mathcal{S}$ KEM VS. THE FO KEM



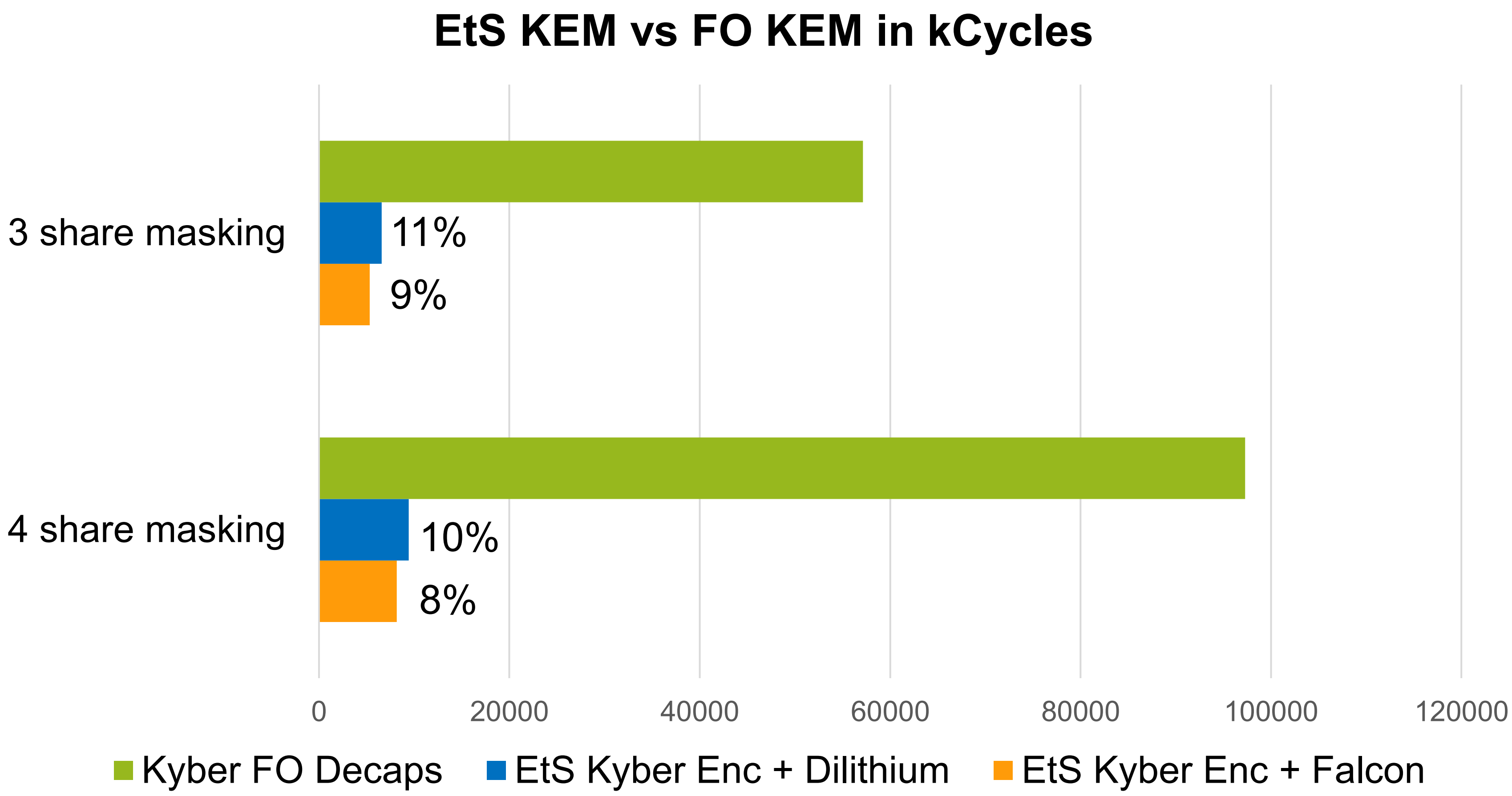
- CPA PKE Decryption -

THE $\mathcal{E}t\mathcal{S}$ KEM VS. THE FO KEM



- CCA $\mathcal{E}t\mathcal{S}$ KEM Decapsulation -

THE $\mathcal{E}t\mathcal{S}$ KEM VS. THE FO KEM

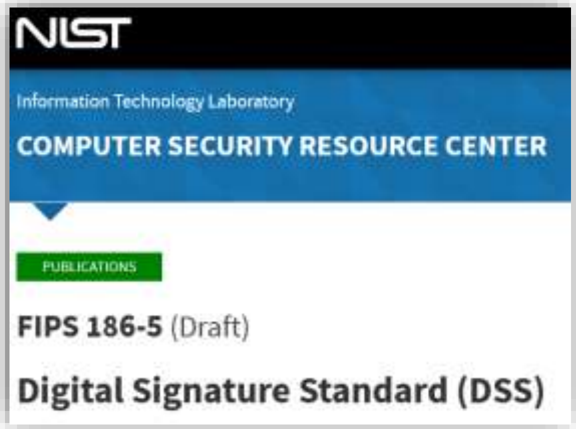


Ciphertext size	$1088\ bytes$	$4381\ bytes$	$2368\ bytes$

Azouaoui, M., Kuzovkova, Y., Schneider, T., van Vredendaal, C. “Post-Quantum Authenticated Encryption against Chosen-Ciphertext Side-Channel Attacks“. TCHES 2022.

PQC & HW RE-USE

IMPLEMENTING CLASSICAL CRYPTOGRAPHY



NIST

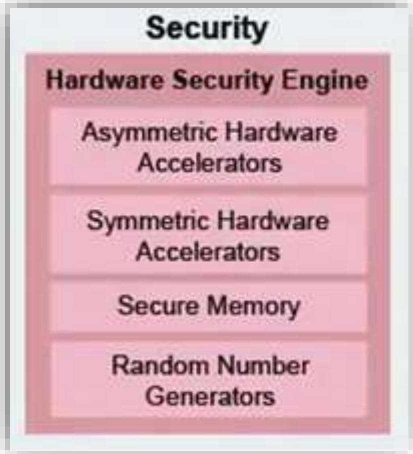
Information Technology Laboratory

COMPUTER SECURITY RESOURCE CENTER

PUBLICATIONS

FIPS 186-5 (Draft)

Digital Signature Standard (DSS)

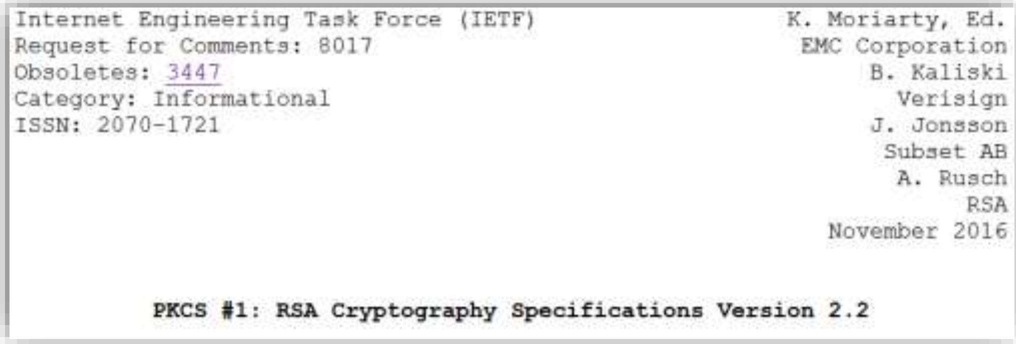


Security

Hardware Security Engine

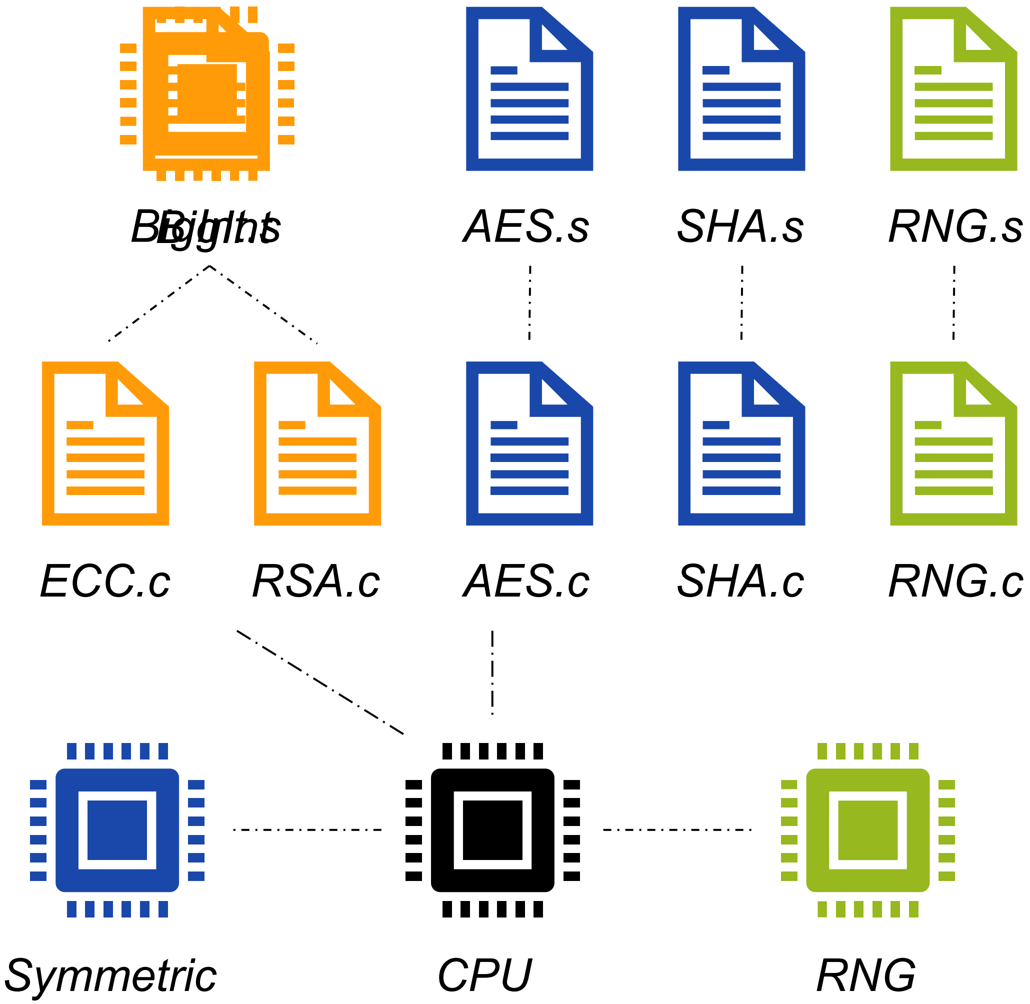
- Asymmetric Hardware Accelerators
- Symmetric Hardware Accelerators
- Secure Memory
- Random Number Generators

S32G2 automotive processor spec



```
Internet Engineering Task Force (IETF)                 K. Moriarty, Ed.
Request for Comments: 8017                              EMC Corporation
Obsoletes: 3447                                              B. Kaliski
Category: Informational                                        Verisign
ISSN: 2070-1721                                              J. Jonsson
                                                              Subset AB
                                                               A. Rusch
                                                                    RSA
                                                          November 2016


          PKCS #1: RSA Cryptography Specifications Version 2.2
```



1. https://www.nxp.com/products/processors-and-microcontrollers/arm-processors/s32g-vehicle-network-processors:S32G-PROCESSORS

IMPLEMENTING POST-QUANTUM CRYPTOGRAPHY

NIST

Information Technology Laboratory

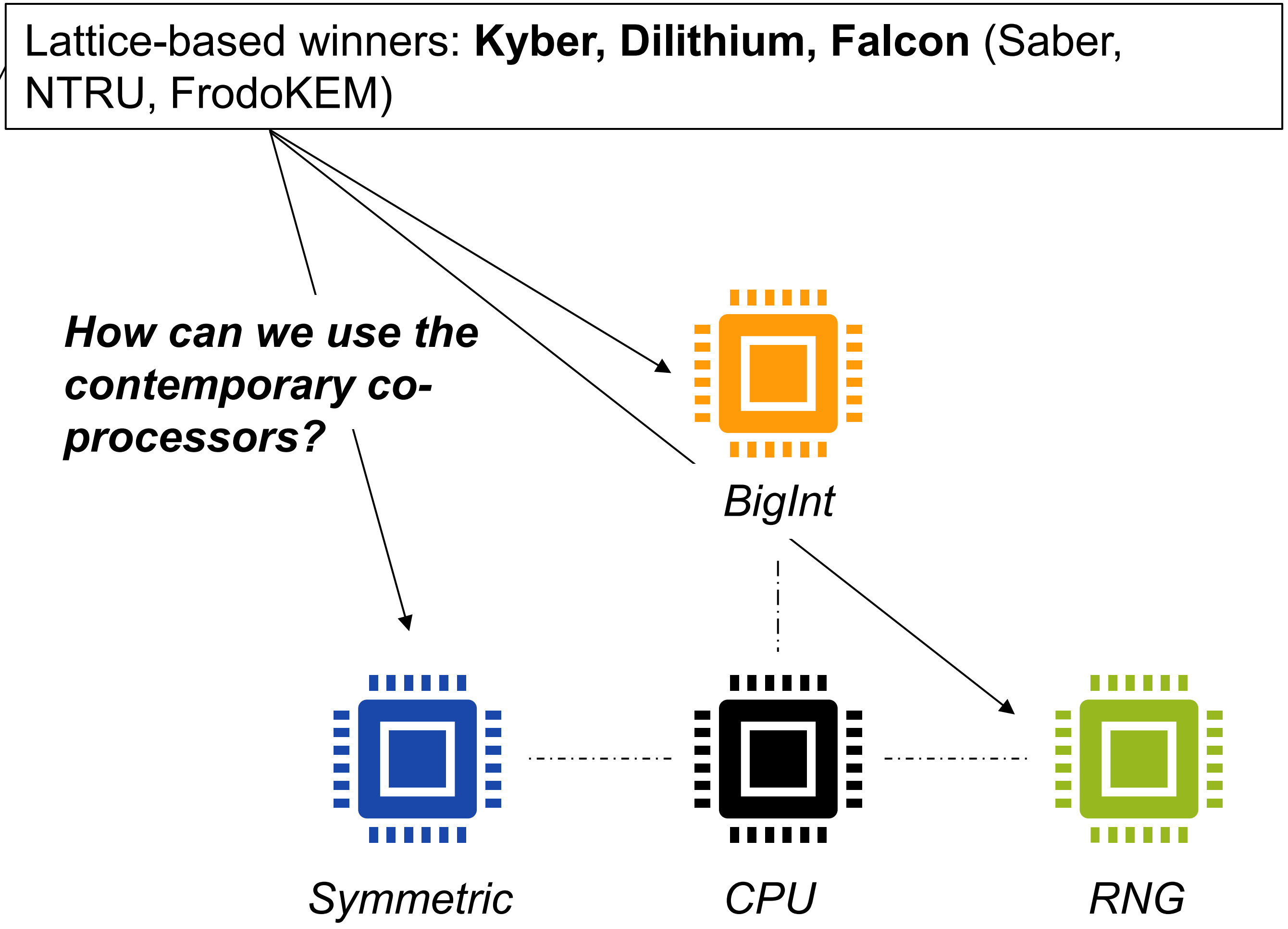
COMPUTER SECURITY RESOURCE CENTER

PROJECTS

Post-Quantum Cryptography PQC

Project Overview

NIST has initiated a process to solicit, evaluate, and standardize one or more quantum-resistant public-key cryptographic algorithms. **Full details can be found in the Post-Quantum Cryptography Standardization page.**



Lattice-based winners: **Kyber, Dilithium, Falcon** (Saber, NTRU, FrodoKEM)

How can we use the contemporary co-processors?

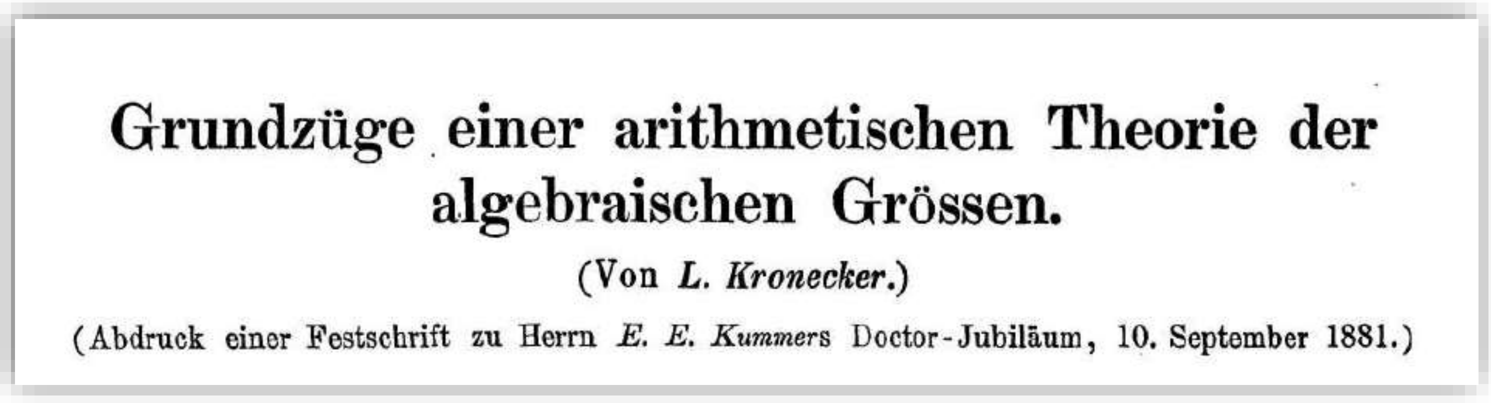
RE-USING EXISTING HW

Approach	Core	Structure	Size
RSA	Modular multiplication	$(\mathbb{Z}/n\mathbb{Z})^*$	n is 3072-bit
ECC	Elliptic curve scalar multiplication	$\mathrm{E}(\mathbb{F}_p)$	p is 256-bit
Lattice	Polynomial multiplication	$(\mathbb{Z}/q\mathbb{Z})[X]/(X^n+1)$	q is 16-bit n is 256

Co-pro present in chips

Can we use this?

KRONECKER SUBSTITUTION



Grundzüge einer arithmetischen Theorie der algebraischen Grössen.

(Von *L. Kronecker.*)

(Abdruck einer Festschrift zu Herrn *E. E. Kummers* Doctor-Jubiläum, 10. September 1881.)

Polynomial domain

$$f = 1 + 2x + 3x^2 + 4x^3$$

$$g = 5 + 6x + 7x^2 + 8x^3$$

$$\times$$

$$fg = \underline{5} + \underline{16}x + \underline{34}x^2 + \underline{60}x^3 + \underline{61}x^4 + \underline{52}x^5 + \underline{32}x^6$$

Kronecker domain (with evaluation point 100)

$$f(100) = 4030201$$

$$g(100) = 8070605$$

$$\times$$

$$fg(100) = \underline{3}\underline{25}\underline{26}\underline{16}\underline{03}\underline{41}\underline{6}05$$

POLYNOMIAL MULTIPLICATION TECHNIQUES

Kronecker evaluation at 2^{32}

Multiplication with a 256-bit multiplier

$$\begin{tikzcd}
\frac{\mathbb{Z}[X]}{(X^{256}+1)} \arrow[r, dashed, "Kronecker"] \arrow[d] \arrow[dd, dashed, bend right=60, "Nussbaumer"'] & \frac{\mathbb{Z}}{(2^{8192}+1)} \arrow[r, dashed, "Schön.\text{-}Strassen"] & \frac{\mathbb{Z}/(2^{544}+1)[X]}{(X^{32}+1)} \\
\frac{\mathbb{Z}[Y]/(Y^{8}+1)[X]}{(X^{32}-Y)} \arrow[r, dashed, "Kronecker"] \arrow[d] & \frac{\mathbb{Z}/(2^{256}+1)[X]}{(X^{32}-2^{32})} \arrow[r, dashed, "Twist"] & \frac{\mathbb{Z}/(2^{256}+1)[X]}{(X^{32}-1)} \\
\frac{\mathbb{Z}[Y]/(Y^{8}+1)[X]}{(X^{64}-1)} \arrow[r, dashed, "Kronecker"] & \frac{\mathbb{Z}/(2^{256}+1)[X]}{(X^{64}-1)} &
\end{tikzcd}$$

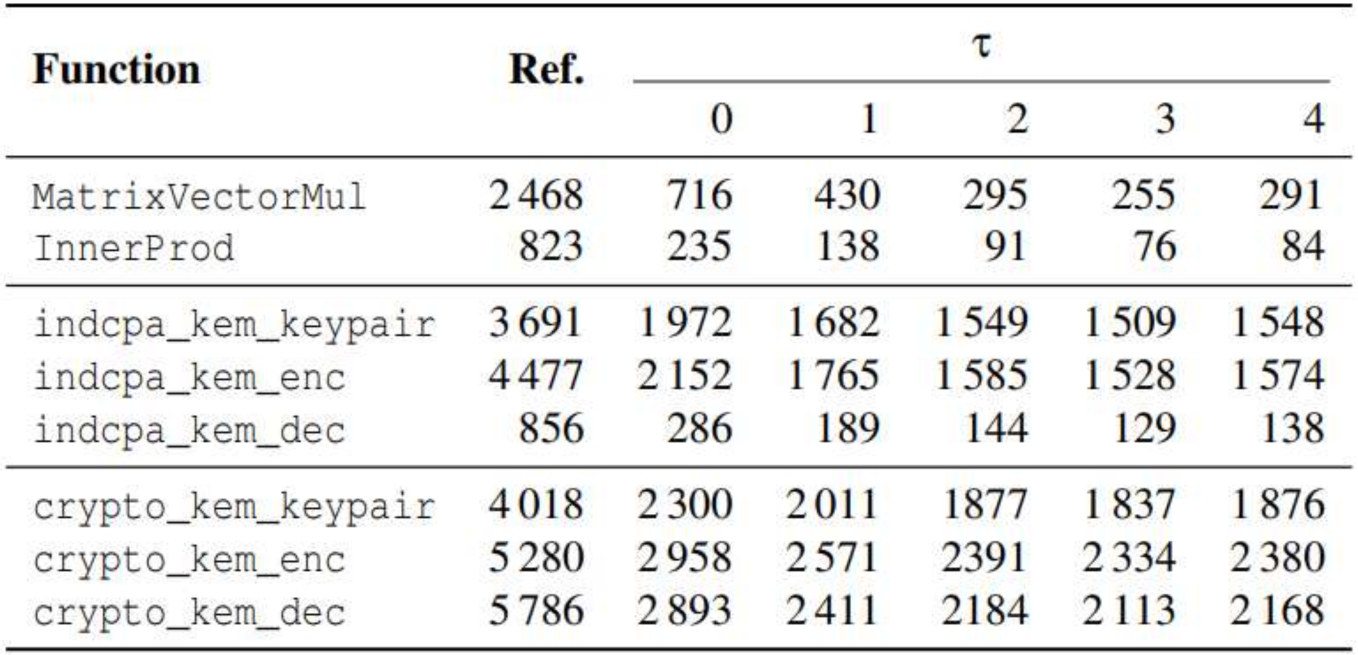
Kronecker+

Algorithm	# Muls	# Bits
Kron. + Schoolbook	1024	256
Kron. + Karatsuba	243	256
Kron. + Toom-Cook	63	256
Kron. + Schön.-Strassen	32	544
Nussbaumer + Kron.	64	256
Kronecker+	**32**	**256**

- Harvey. Faster polynomial multiplication via multipoint Kronecker substitution. J. of Sym. Comp. 2009.
- Albrecht, Hanser, Hoeller, Pöppelmann, Virdia, Wallner; Implementing RLWE-based schemes using an RSA co-processor. TCHES 2019
- Bos, Renes, van Vredendaal: Polynomial Multiplication with Contemporary Co-Processors: Beyond Kronecker, Schönhage-Strassen & Nussbaumer. USENIX 2022.

CAN WE USE EXISTING HARDWARE

- Works very well for Saber, ~8-10x faster for matrix / vector multiplication on RISC-V



Function	Ref.	τ				
		0	1	2	3	4
MatrixVectorMul	2 468	716	430	295	255	291
InnerProd	823	235	138	91	76	84
indcpa_kem_keypair	3 691	1 972	1 682	1 549	1 509	1 548
indcpa_kem_enc	4 477	2 152	1 765	1 585	1 528	1 574
indcpa_kem_dec	856	286	189	144	129	138
crypto_kem_keypair	4 018	2 300	2 011	1877	1 837	1 876
crypto_kem_enc	5 280	2 958	2 571	2391	2 334	2 380
crypto_kem_dec	5 786	2 893	2 411	2184	2 113	2 168

Cycle counts on RV32IMC in 1000s of cycles, rounded up

- CRYSTALS Design: Sample matrix elements directly in NTT domain

LOW-MEMORY PQC

SECURE ELEMENTS AND END-TO-END SERVICES

NXP propels today's on-the-go lifestyle with intelligent mobile solutions that safely connect consumers and their technology to the world around them.

SECURE ELEMENTS AND END-TO-END SERVICES

CUSTOM HIGH-PERFORMANCE INTERFACES

SMART VOICE, AUDIO, AND HAPTIC SOLUTIONS

EFFICIENT CHARGING SOLUTIONS

DEFINING WHAT'S NEXT FOR MOBILE PHONES

NXP has been driving the mobile wallet expansion, advancing analog and charging solutions add more capabilities to mobile phones, notebooks, and tablets.

- NFC, eSE, eSIM, and UWB solutions
- Advanced analog solutions for personal computing
- Fast charging with USB Type-C

WEARABLES

Thanks to secure mobile payments, advanced audio solutions and tailored MCUs, wearables naturally blend into our lives.

- NFC+eSE mobile wallet solutions
- Highly integrated Arm® based MPUs and MCUs
- MiGLO™ NFMI radios for wireless audio

ACCESSORIES

NXP's anti-counterfeiting technology, among others products, support charging cables, power adapters, and wireless charging pads for mobile phones to help OEMs protect their brand and provides safety to their customers by making trusted accessories.

INDUSTRIAL

- Fit-for-purpose Scalable Processors
- Functional Safety & Security
- Industrial Connectivity & Control
- Machine Learning & Vision
- Comprehensive Software

PQC ON EMBEDDED DEVICES

What is embedded?

- NIST has recommended a focus on the Arm Cortex-M4

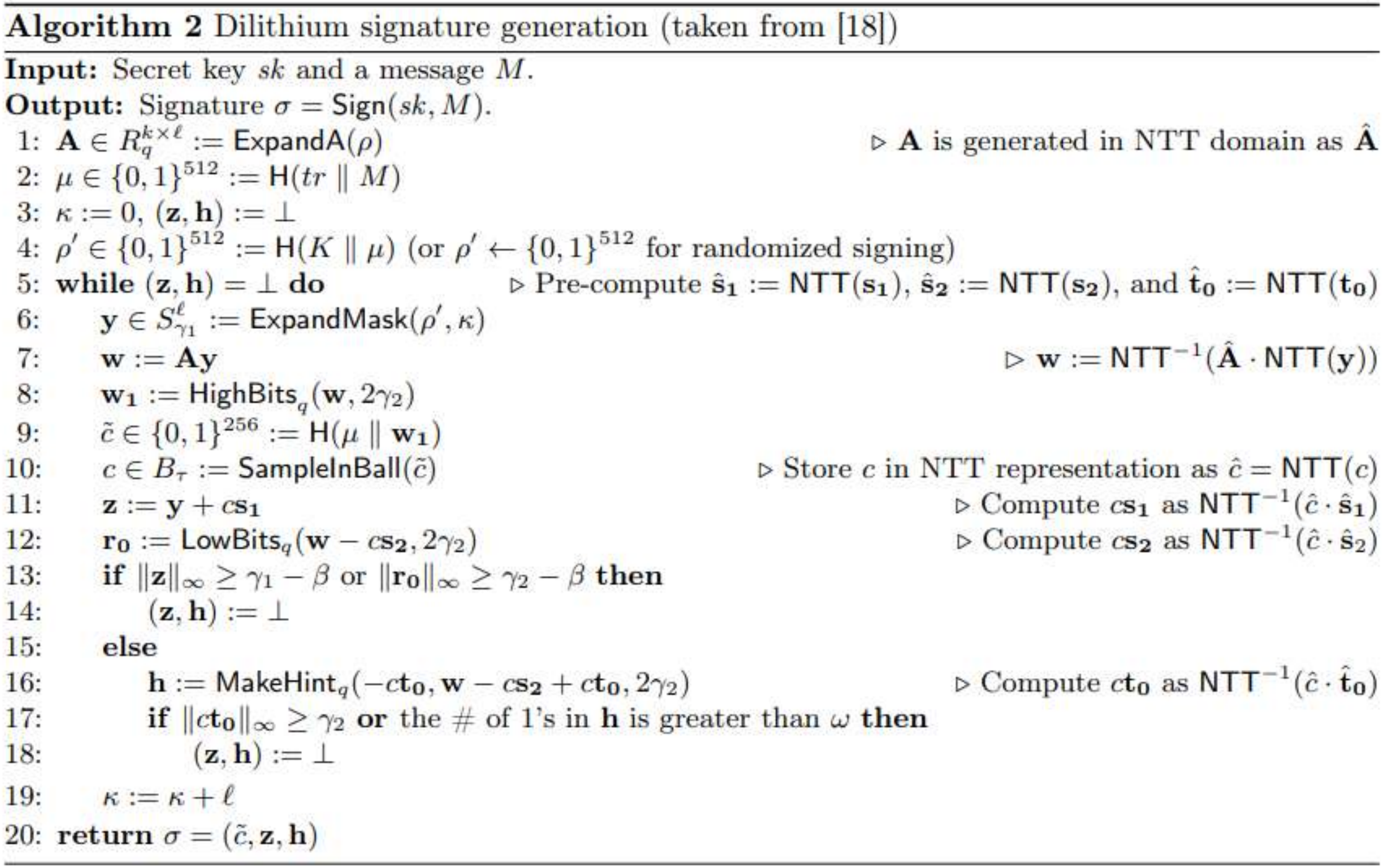
Pqm4: Post-quantum crypto library for the ARM Cortex-M4, STM32F4DISCOVERY
196 KiB of RAM and 1 MiB of Flash ROM

Low-power Edge computing: LPC800 Series

- 8 to 60 MHz Cortex-M0+ core
- $\{4, 8, 16\}$ KiB of SRAM
- $\{16, 32\}$ KiB Flash

The fastest implementations in pqm4 require ≈ 49, ≈ 80 and ≈ 116 KiB memory for Dilithium-{2,3,5}.

DILITHIUM SIGNATURE GENERATION



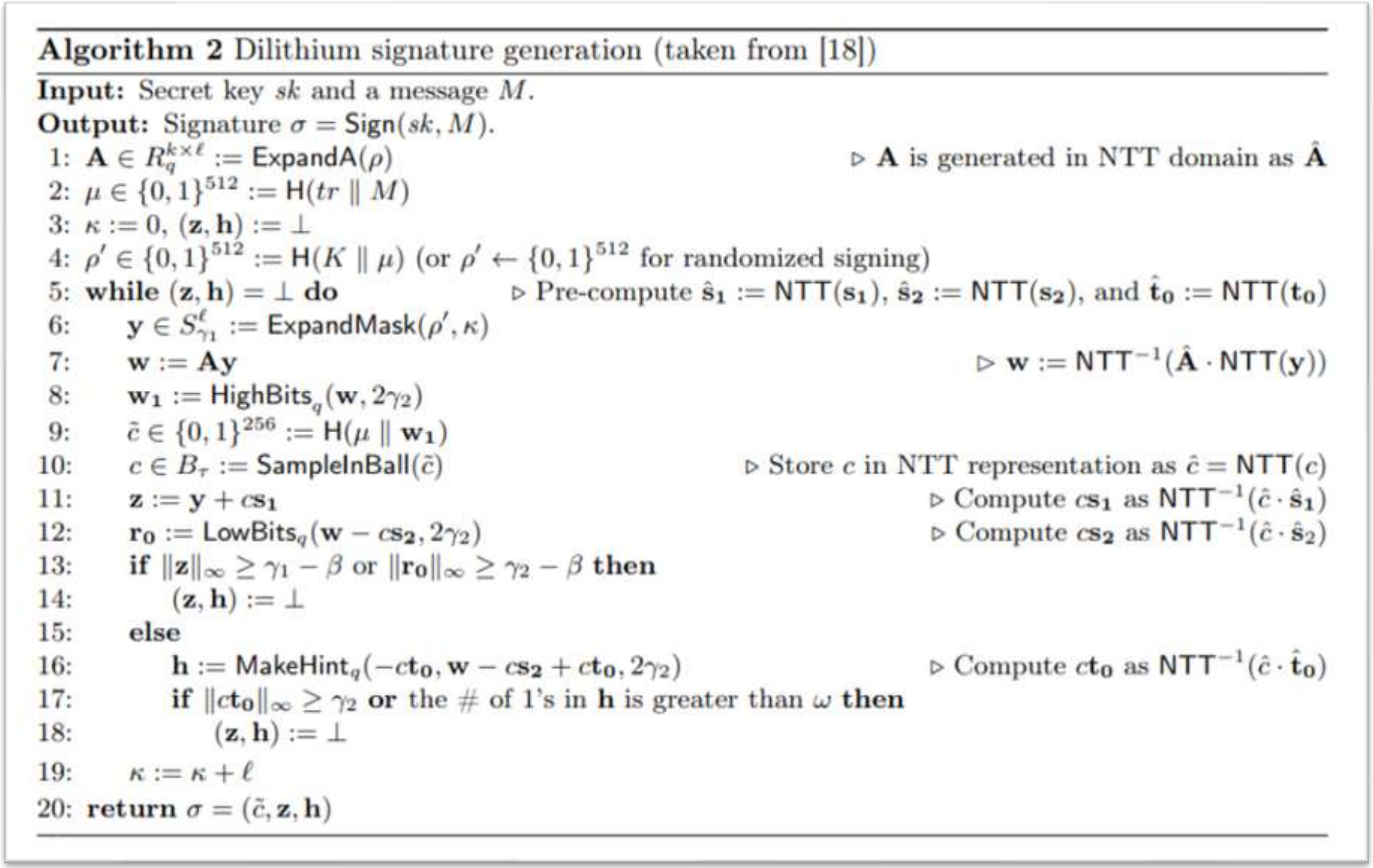
Algorithm 2 Dilithium signature generation (taken from [18])

Input: Secret key sk and a message M.

Output: Signature $\sigma = \mathsf{Sign}(sk, M)$.

```
$\mathbf{A} \in R_q^{k\times\ell} := \mathsf{ExpandA}(\rho)$    ▷ $\mathbf{A}$ is generated in NTT domain as $\hat{\mathbf{A}}$
$\mu \in \{0,1\}^{512} := \mathsf{H}(tr \parallel M)$
$\kappa := 0$, $(\mathbf{z}, \mathbf{h}) := \perp$
$\rho' \in \{0,1\}^{512} := \mathsf{H}(K \parallel \mu)$ (or $\rho' \leftarrow \{0,1\}^{512}$ for randomized signing)
while $(\mathbf{z}, \mathbf{h}) = \perp$ do    ▷ Pre-compute $\hat{\mathbf{s}}_1 := \mathsf{NTT}(\mathbf{s_1})$, $\hat{\mathbf{s}}_2 := \mathsf{NTT}(\mathbf{s_2})$, and $\hat{\mathbf{t}}_0 := \mathsf{NTT}(\mathbf{t_0})$
    $\mathbf{y} \in S_{\gamma_1}^{\ell} := \mathsf{ExpandMask}(\rho', \kappa)$
    $\mathbf{w} := \mathbf{A}\mathbf{y}$    ▷ $\mathbf{w} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \cdot \mathsf{NTT}(\mathbf{y}))$
    $\mathbf{w_1} := \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := \mathsf{H}(\mu \parallel \mathbf{w_1})$
    $c \in B_\tau := \mathsf{SampleInBall}(\tilde{c})$    ▷ Store $c$ in NTT representation as $\hat{c} = \mathsf{NTT}(c)$
    $\mathbf{z} := \mathbf{y} + c\mathbf{s_1}$    ▷ Compute $c\mathbf{s_1}$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_1)$
    $\mathbf{r_0} := \mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s_2}, 2\gamma_2)$    ▷ Compute $c\mathbf{s_2}$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_2)$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\mathbf{r_0}\|_\infty \geq \gamma_2 - \beta$ then
        $(\mathbf{z}, \mathbf{h}) := \perp$
    else
        $\mathbf{h} := \mathsf{MakeHint}_q(-c\mathbf{t_0}, \mathbf{w} - c\mathbf{s_2} + c\mathbf{t_0}, 2\gamma_2)$    ▷ Compute $c\mathbf{t_0}$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{t}}_0)$
        if $\|c\mathbf{t_0}\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than $\omega$ then
            $(\mathbf{z}, \mathbf{h}) := \perp$
    $\kappa := \kappa + \ell$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$
```

DILITHIUM SIGNATURE GENERATION



Algorithm 2 Dilithium signature generation (taken from [18])

Input: Secret key sk and a message M.

Output: Signature $\sigma = \mathsf{Sign}(sk, M)$.

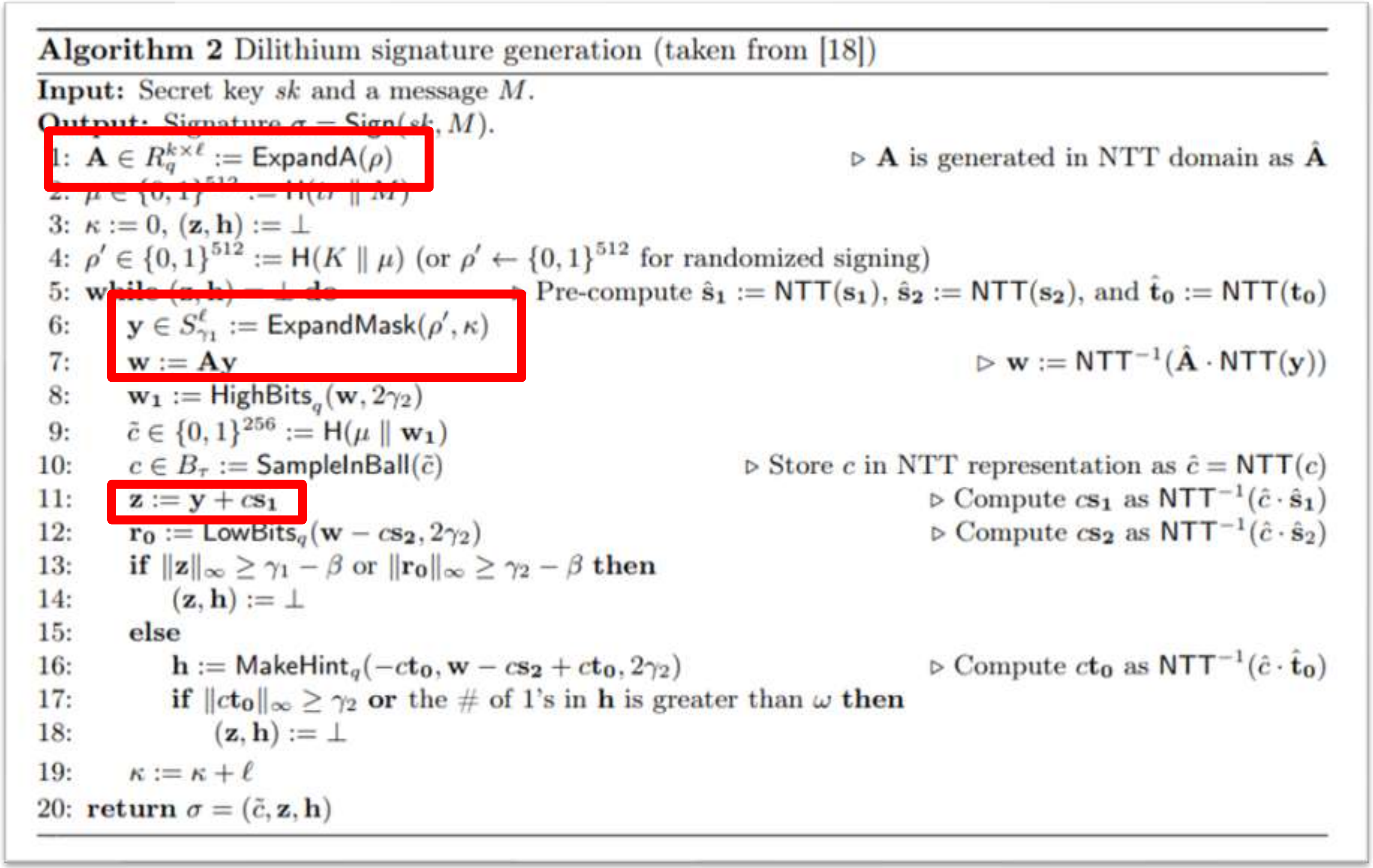
```
$\mathbf{A} \in R_q^{k \times \ell} := \mathsf{ExpandA}(\rho)$    ▷ $\mathbf{A}$ is generated in NTT domain as $\hat{\mathbf{A}}$
$\mu \in \{0,1\}^{512} := \mathsf{H}(tr \,\|\, M)$
$\kappa := 0$, $(\mathbf{z}, \mathbf{h}) := \perp$
$\rho' \in \{0,1\}^{512} := \mathsf{H}(K \,\|\, \mu)$ (or $\rho' \leftarrow \{0,1\}^{512}$ for randomized signing)
while $(\mathbf{z}, \mathbf{h}) = \perp$ do    ▷ Pre-compute $\hat{\mathbf{s}}_1 := \mathsf{NTT}(\mathbf{s}_1)$, $\hat{\mathbf{s}}_2 := \mathsf{NTT}(\mathbf{s}_2)$, and $\hat{\mathbf{t}}_0 := \mathsf{NTT}(\mathbf{t}_0)$
    $\mathbf{y} \in S_{\gamma_1}^{\ell} := \mathsf{ExpandMask}(\rho', \kappa)$
    $\mathbf{w} := \mathbf{A}\mathbf{y}$    ▷ $\mathbf{w} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \cdot \mathsf{NTT}(\mathbf{y}))$
    $\mathbf{w}_1 := \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := \mathsf{H}(\mu \,\|\, \mathbf{w}_1)$
    $c \in B_\tau := \mathsf{SampleInBall}(\tilde{c})$    ▷ Store $c$ in NTT representation as $\hat{c} = \mathsf{NTT}(c)$
    $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$    ▷ Compute $c\mathbf{s}_1$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_1)$
    $\mathbf{r}_0 := \mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)$    ▷ Compute $c\mathbf{s}_2$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_2)$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta$ then
        $(\mathbf{z}, \mathbf{h}) := \perp$
    else
        $\mathbf{h} := \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2)$    ▷ Compute $c\mathbf{t}_0$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{t}}_0)$
        if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than $\omega$ then
            $(\mathbf{z}, \mathbf{h}) := \perp$
    $\kappa := \kappa + \ell$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$
```

Polynomials from
$$R_q = \mathbb{Z}_q[X]/(X^{256} + 1)$$
where $q = 2^{23} - 2^{13} + 1$ and stored as 32-bit values.

→ One R_q elements needs **1KB**

Dilithium-3: $(k, \ell) = (6,5)$

DILITHIUM SIGNATURE GENERATION



Algorithm 2 Dilithium signature generation (taken from [18])

Input: Secret key sk and a message M.

Output: Signature $\sigma = \mathsf{Sign}(sk, M)$.

```
$\mathbf{A} \in R_q^{k\times\ell} := \mathsf{ExpandA}(\rho)$    ▷ $\mathbf{A}$ is generated in NTT domain as $\hat{\mathbf{A}}$
$\mu \in \{0,1\}^{512} := \mathsf{H}(tr \,\|\, M)$
$\kappa := 0$, $(\mathbf{z}, \mathbf{h}) := \perp$
$\rho' \in \{0,1\}^{512} := \mathsf{H}(K \,\|\, \mu)$ (or $\rho' \leftarrow \{0,1\}^{512}$ for randomized signing)
while $(\mathbf{z}, \mathbf{h}) = \perp$ do    ▷ Pre-compute $\hat{\mathbf{s}}_1 := \mathsf{NTT}(\mathbf{s}_1)$, $\hat{\mathbf{s}}_2 := \mathsf{NTT}(\mathbf{s}_2)$, and $\hat{\mathbf{t}}_0 := \mathsf{NTT}(\mathbf{t}_0)$
    $\mathbf{y} \in S_{\gamma_1}^{\ell} := \mathsf{ExpandMask}(\rho', \kappa)$
    $\mathbf{w} := \mathbf{A}\mathbf{y}$    ▷ $\mathbf{w} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \cdot \mathsf{NTT}(\mathbf{y}))$
    $\mathbf{w}_1 := \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := \mathsf{H}(\mu \,\|\, \mathbf{w}_1)$
    $c \in B_\tau := \mathsf{SampleInBall}(\tilde{c})$    ▷ Store $c$ in NTT representation as $\hat{c} = \mathsf{NTT}(c)$
    $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$    ▷ Compute $c\mathbf{s}_1$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_1)$
    $\mathbf{r}_0 := \mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)$    ▷ Compute $c\mathbf{s}_2$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_2)$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta$ then
        $(\mathbf{z}, \mathbf{h}) := \perp$
    else
        $\mathbf{h} := \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2)$    ▷ Compute $c\mathbf{t}_0$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{t}}_0)$
        if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than $\omega$ then
            $(\mathbf{z}, \mathbf{h}) := \perp$
    $\kappa := \kappa + \ell$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$
```

Polynomials from

$$R_q = \mathbb{Z}_q[X]/(X^{256}+1)$$

where $q = 2^{23} - 2^{13} + 1$ and stored as 32-bit values.

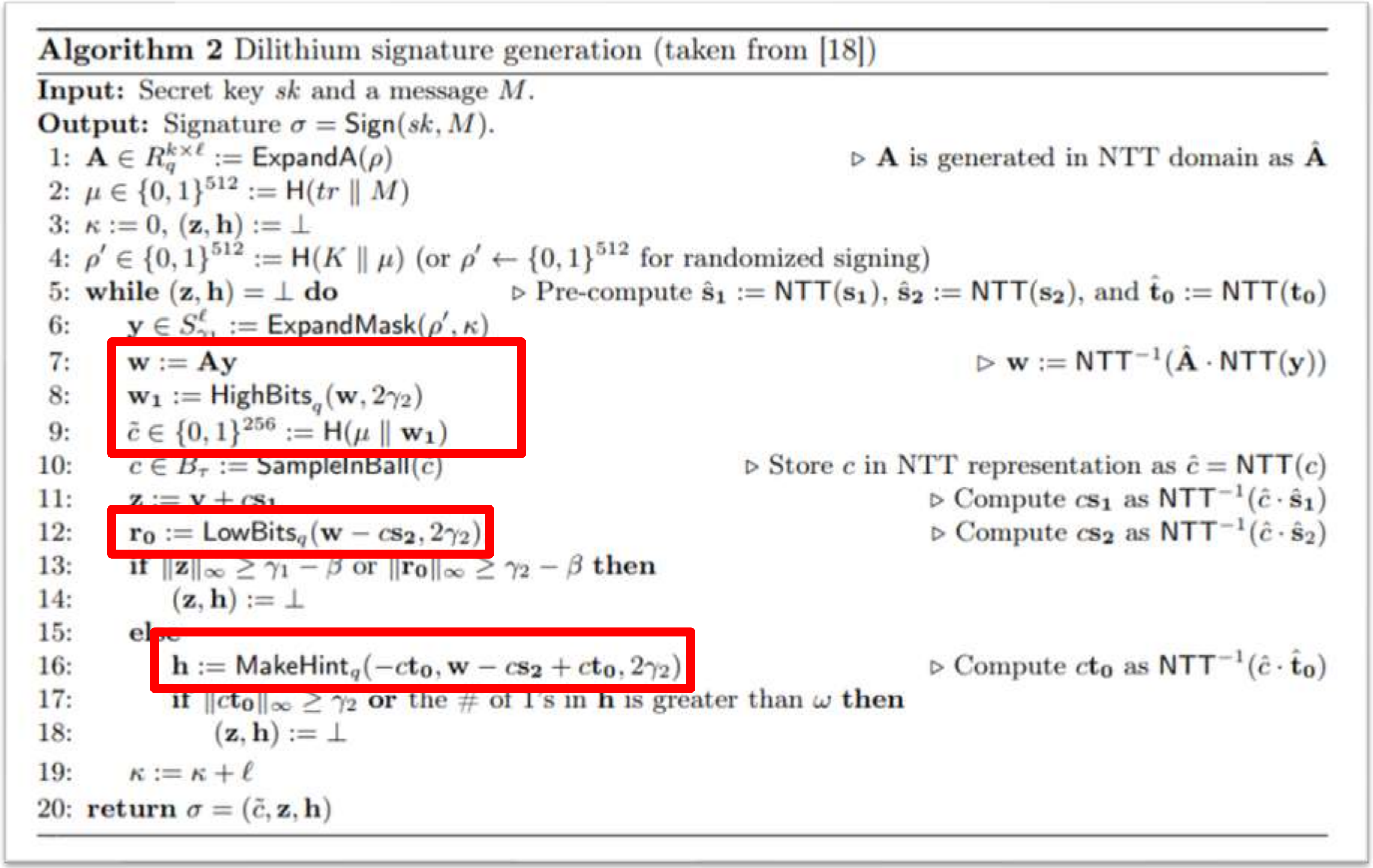
→ One R_q elements needs **1KB**

Dilithium-3: $(k, \ell) = (6{,}5)$

(Re-)generate matrix A and y on-the-fly

- Reduce by $k \cdot \ell$ KB for A
  → **30 KB**
- Reduce by ℓ KB for y
  → **5 KB**

DILITHIUM SIGNATURE GENERATION



Algorithm 2 Dilithium signature generation (taken from [18])

Input: Secret key sk and a message M.

Output: Signature $\sigma = \mathsf{Sign}(sk, M)$.

```
$\mathbf{A} \in R_q^{k \times \ell} := \mathsf{ExpandA}(\rho)$    ▷ $\mathbf{A}$ is generated in NTT domain as $\hat{\mathbf{A}}$
$\mu \in \{0,1\}^{512} := \mathsf{H}(tr \parallel M)$
$\kappa := 0$, $(\mathbf{z}, \mathbf{h}) := \perp$
$\rho' \in \{0,1\}^{512} := \mathsf{H}(K \parallel \mu)$ (or $\rho' \leftarrow \{0,1\}^{512}$ for randomized signing)
while $(\mathbf{z}, \mathbf{h}) = \perp$ do    ▷ Pre-compute $\hat{\mathbf{s}}_1 := \mathsf{NTT}(\mathbf{s}_1)$, $\hat{\mathbf{s}}_2 := \mathsf{NTT}(\mathbf{s}_2)$, and $\hat{\mathbf{t}}_0 := \mathsf{NTT}(\mathbf{t}_0)$
    $\mathbf{y} \in S_{\gamma_1}^{\ell} := \mathsf{ExpandMask}(\rho', \kappa)$
    $\mathbf{w} := \mathbf{A}\mathbf{y}$    ▷ $\mathbf{w} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \cdot \mathsf{NTT}(\mathbf{y}))$
    $\mathbf{w}_1 := \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := \mathsf{H}(\mu \parallel \mathbf{w}_1)$
    $c \in B_\tau := \mathsf{SampleInBall}(\tilde{c})$    ▷ Store $c$ in NTT representation as $\hat{c} = \mathsf{NTT}(c)$
    $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$    ▷ Compute $c\mathbf{s}_1$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_1)$
    $\mathbf{r}_0 := \mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)$    ▷ Compute $c\mathbf{s}_2$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_2)$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta$ then
        $(\mathbf{z}, \mathbf{h}) := \perp$
    else
        $\mathbf{h} := \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2)$    ▷ Compute $c\mathbf{t}_0$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{t}}_0)$
        if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than $\omega$ then
            $(\mathbf{z}, \mathbf{h}) := \perp$
    $\kappa := \kappa + \ell$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$
```

Polynomials from

$$R_q = \mathbb{Z}_q[X]/(X^{256} + 1)$$

where $q = 2^{23} - 2^{13} + 1$ and stored as 32-bit values.

→ One R_q elements needs **1KB**

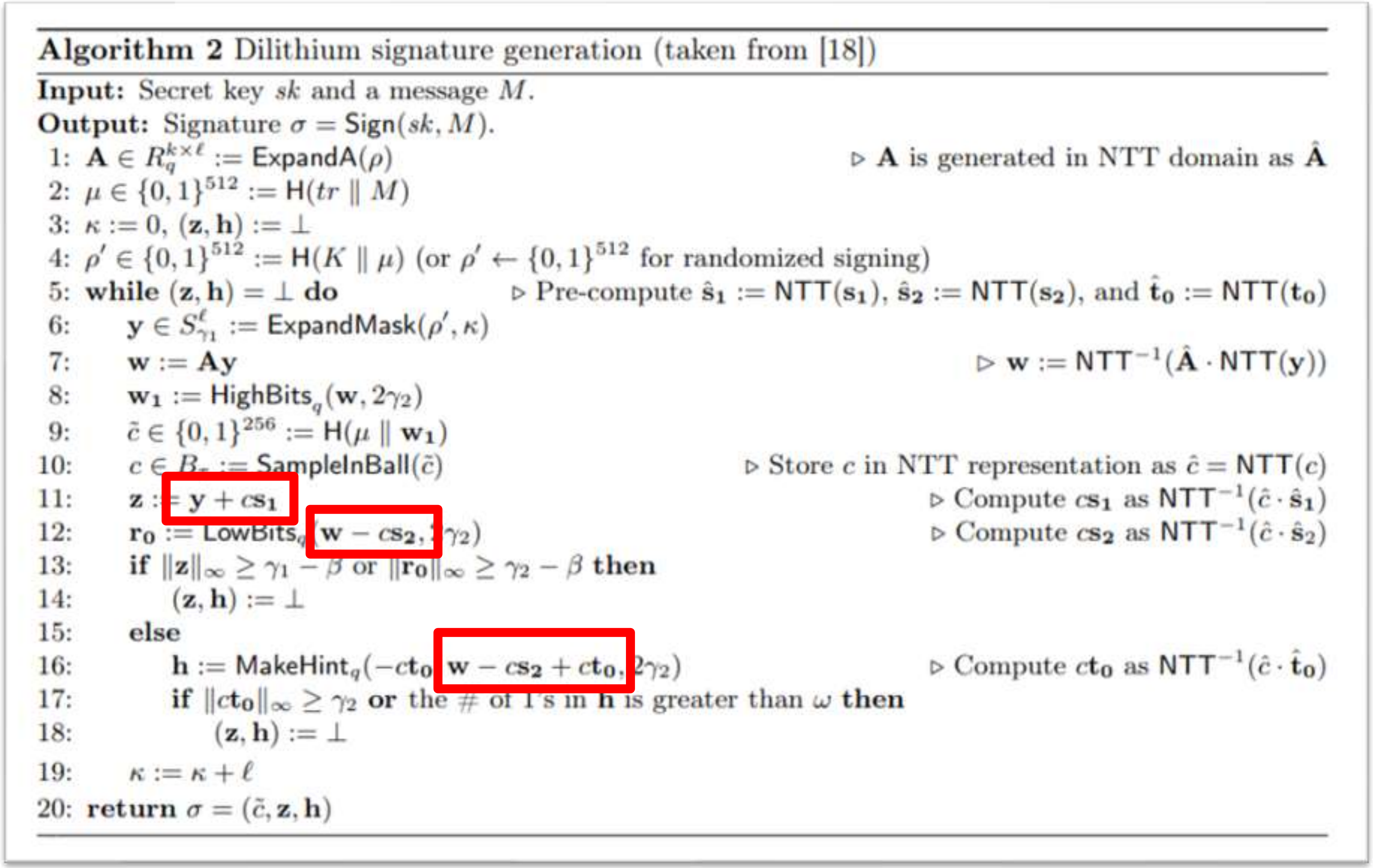
Dilithium-3: $(k, \ell) = (6,5)$

(Re-)generate matrix A and y on-the-fly: ~~80~~, 45 KB

Compress w

- Store values as 24-bit
- One R_q elements needs 768 bytes
- Packing and unpacking is simple and efficient
- Reduces memory by Reduce by $256k$ bytes → 1.5 KB

DILITHIUM SIGNATURE GENERATION



Algorithm 2 Dilithium signature generation (taken from [18])

Input: Secret key sk and a message M.

Output: Signature $\sigma = \mathsf{Sign}(sk, M)$.

```
$\mathbf{A} \in R_q^{k\times\ell} := \mathsf{ExpandA}(\rho)$    ▷ $\mathbf{A}$ is generated in NTT domain as $\hat{\mathbf{A}}$
$\mu \in \{0,1\}^{512} := \mathsf{H}(tr \,\|\, M)$
$\kappa := 0$, $(\mathbf{z}, \mathbf{h}) := \perp$
$\rho' \in \{0,1\}^{512} := \mathsf{H}(K \,\|\, \mu)$ (or $\rho' \leftarrow \{0,1\}^{512}$ for randomized signing)
while $(\mathbf{z}, \mathbf{h}) = \perp$ do    ▷ Pre-compute $\hat{\mathbf{s}}_1 := \mathsf{NTT}(\mathbf{s}_1)$, $\hat{\mathbf{s}}_2 := \mathsf{NTT}(\mathbf{s}_2)$, and $\hat{\mathbf{t}}_0 := \mathsf{NTT}(\mathbf{t}_0)$
    $\mathbf{y} \in S_{\gamma_1}^{\ell} := \mathsf{ExpandMask}(\rho', \kappa)$
    $\mathbf{w} := \mathbf{A}\mathbf{y}$    ▷ $\mathbf{w} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \cdot \mathsf{NTT}(\mathbf{y}))$
    $\mathbf{w}_1 := \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := \mathsf{H}(\mu \,\|\, \mathbf{w}_1)$
    $c \in B_\tau := \mathsf{SampleInBall}(\tilde{c})$    ▷ Store $c$ in NTT representation as $\hat{c} = \mathsf{NTT}(c)$
    $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$    ▷ Compute $c\mathbf{s}_1$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_1)$
    $\mathbf{r}_0 := \mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)$    ▷ Compute $c\mathbf{s}_2$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_2)$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta$ then
        $(\mathbf{z}, \mathbf{h}) := \perp$
    else
        $\mathbf{h} := \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2)$    ▷ Compute $c\mathbf{t}_0$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{t}}_0)$
        if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than $\omega$ then
            $(\mathbf{z}, \mathbf{h}) := \perp$
    $\kappa := \kappa + \ell$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$
```

Polynomials from

$$R_q = \mathbb{Z}_q[X]/(X^{256} + 1)$$

where $q = 2^{23} - 2^{13} + 1$ and stored as 32-bit values.

→ One R_q elements needs **1KB**

Dilithium-3: $(k, \ell) = (6,5)$

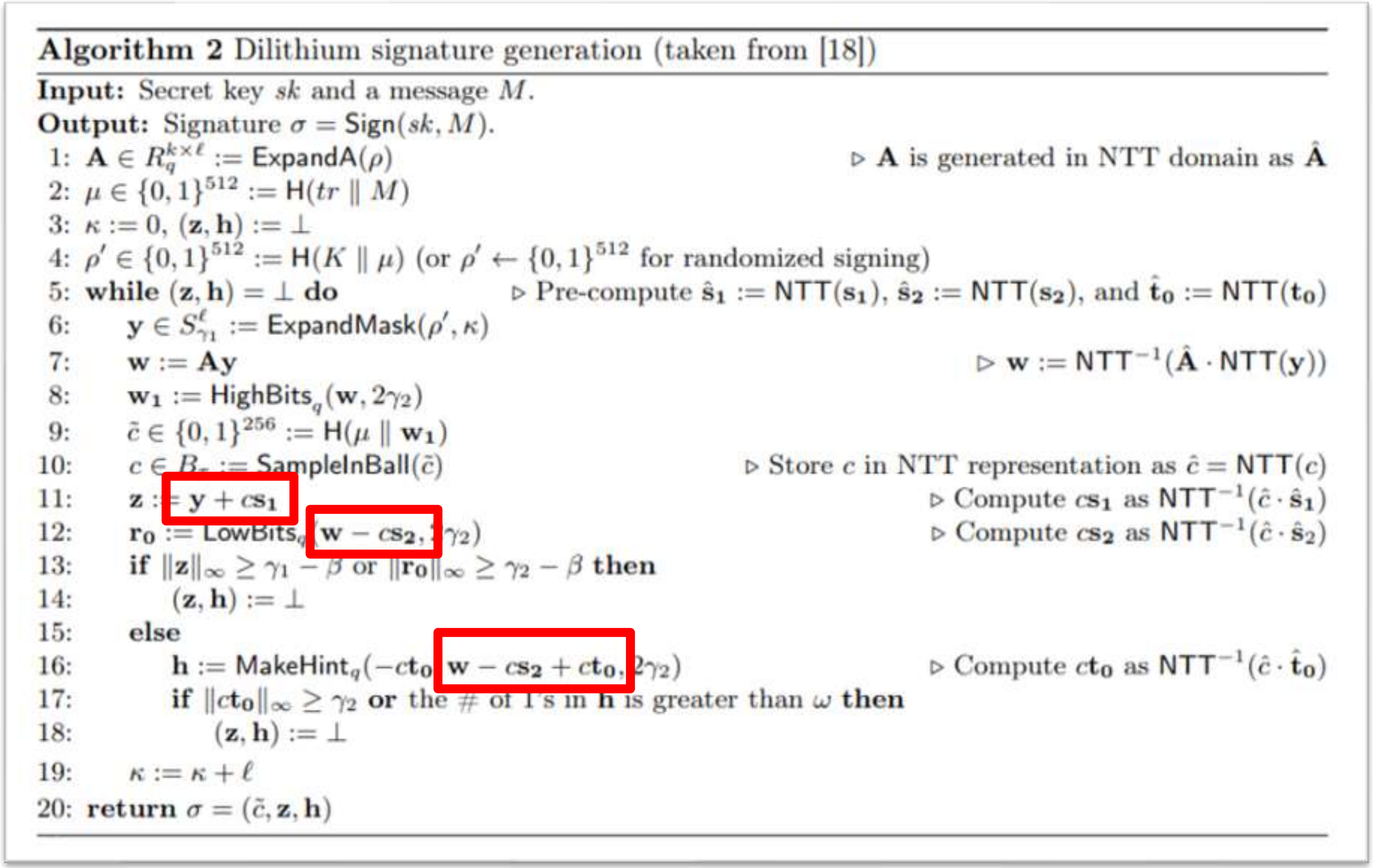
(Re-)generate matrix A and y on-the-fly: ~~80 KB~~ → 45 KB

Compress w: ~~45 KB~~ → 43.5 KB

Compressing multiplications

- NTT used for faster polynomial multiplication
- Secret key coefficient range is much smaller
- Not using NTT reduces by $2k + \ell$ KB → **17 KB**

DILITHIUM SIGNATURE GENERATION



Algorithm 2 Dilithium signature generation (taken from [18])

Input: Secret key sk and a message M.

Output: Signature $\sigma = \mathsf{Sign}(sk, M)$.

```
$\mathbf{A} \in R_q^{k \times \ell} := \mathsf{ExpandA}(\rho)$    ▷ $\mathbf{A}$ is generated in NTT domain as $\hat{\mathbf{A}}$
$\mu \in \{0,1\}^{512} := \mathsf{H}(tr \parallel M)$
$\kappa := 0$, $(\mathbf{z}, \mathbf{h}) := \perp$
$\rho' \in \{0,1\}^{512} := \mathsf{H}(K \parallel \mu)$ (or $\rho' \leftarrow \{0,1\}^{512}$ for randomized signing)
while $(\mathbf{z}, \mathbf{h}) = \perp$ do    ▷ Pre-compute $\hat{\mathbf{s}}_1 := \mathsf{NTT}(\mathbf{s}_1)$, $\hat{\mathbf{s}}_2 := \mathsf{NTT}(\mathbf{s}_2)$, and $\hat{\mathbf{t}}_0 := \mathsf{NTT}(\mathbf{t}_0)$
    $\mathbf{y} \in S_{\gamma_1}^{\ell} := \mathsf{ExpandMask}(\rho', \kappa)$
    $\mathbf{w} := \mathbf{A}\mathbf{y}$    ▷ $\mathbf{w} := \mathsf{NTT}^{-1}(\hat{\mathbf{A}} \cdot \mathsf{NTT}(\mathbf{y}))$
    $\mathbf{w}_1 := \mathsf{HighBits}_q(\mathbf{w}, 2\gamma_2)$
    $\tilde{c} \in \{0,1\}^{256} := \mathsf{H}(\mu \parallel \mathbf{w}_1)$
    $c \in B_\tau := \mathsf{SampleInBall}(\tilde{c})$    ▷ Store $c$ in NTT representation as $\hat{c} = \mathsf{NTT}(c)$
    $\mathbf{z} := \mathbf{y} + c\mathbf{s}_1$    ▷ Compute $c\mathbf{s}_1$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_1)$
    $\mathbf{r}_0 := \mathsf{LowBits}_q(\mathbf{w} - c\mathbf{s}_2, 2\gamma_2)$    ▷ Compute $c\mathbf{s}_2$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{s}}_2)$
    if $\|\mathbf{z}\|_\infty \geq \gamma_1 - \beta$ or $\|\mathbf{r}_0\|_\infty \geq \gamma_2 - \beta$ then
        $(\mathbf{z}, \mathbf{h}) := \perp$
    else
        $\mathbf{h} := \mathsf{MakeHint}_q(-c\mathbf{t}_0, \mathbf{w} - c\mathbf{s}_2 + c\mathbf{t}_0, 2\gamma_2)$    ▷ Compute $c\mathbf{t}_0$ as $\mathsf{NTT}^{-1}(\hat{c} \cdot \hat{\mathbf{t}}_0)$
        if $\|c\mathbf{t}_0\|_\infty \geq \gamma_2$ or the # of 1's in $\mathbf{h}$ is greater than $\omega$ then
            $(\mathbf{z}, \mathbf{h}) := \perp$
    $\kappa := \kappa + \ell$
return $\sigma = (\tilde{c}, \mathbf{z}, \mathbf{h})$
```

Polynomials from

$$R_q = \mathbb{Z}_q[X]/(X^{256}+1)$$

where $q = 2^{23} - 2^{13} + 1$ and stored as 32-bit values.
→ One R_q elements needs **1KB**

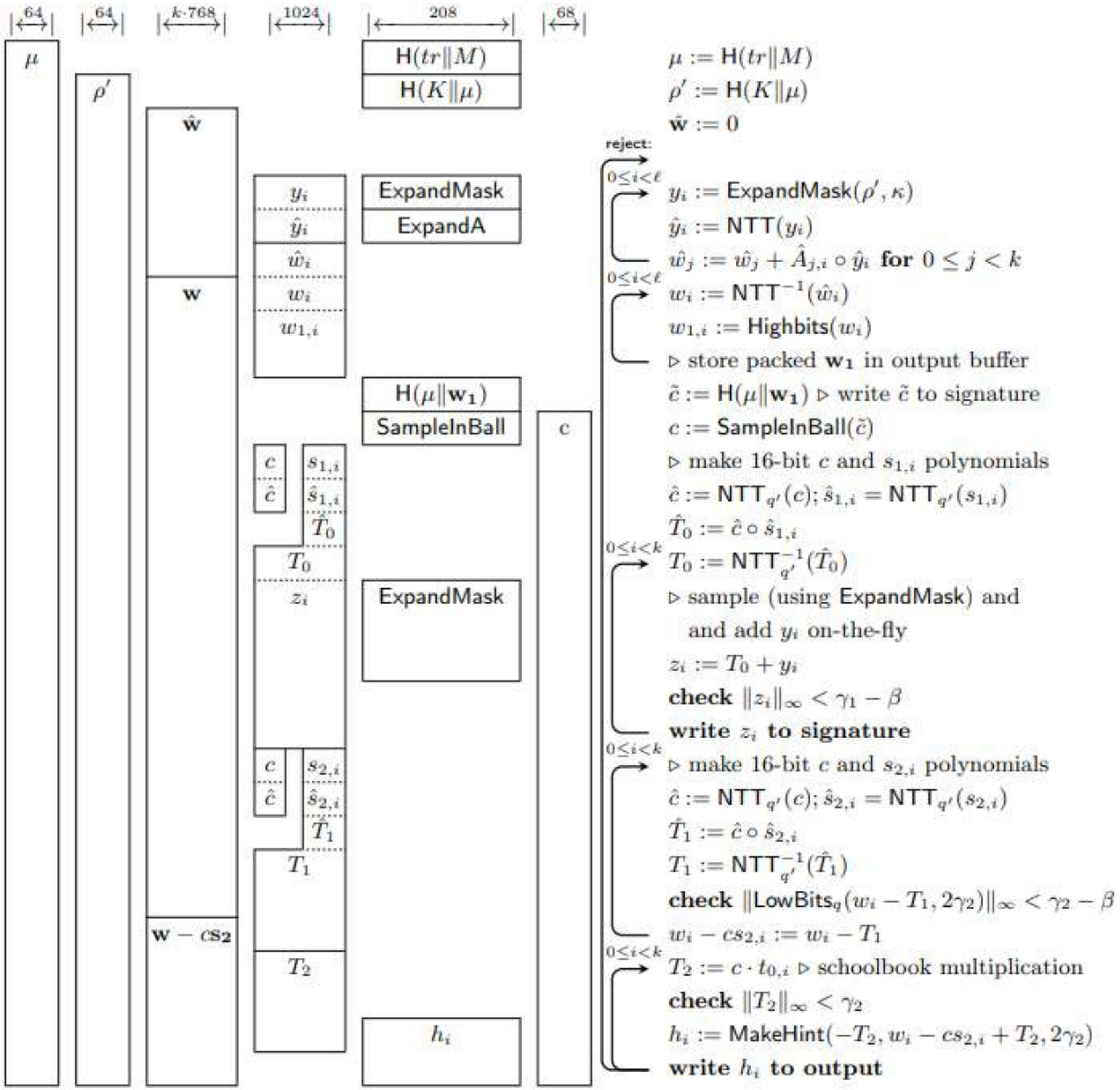
Dilithium-3: $(k, \ell) = (6,5)$

(Re-)generate matrix A and y on-the-fly: ~~80 KB~~ → 45 KB

Compress w: ~~45 KB~~ → 43.5 KB

Compressing multiplications ~~43.5 KB~~ → 26.5 KB

Variable Allocation



```
$\mu := \mathsf{H}(tr\|M)$
$\rho' := \mathsf{H}(K\|\mu)$
$\hat{\mathbf{w}} := 0$
reject:
$0 \le i < \ell$
    $y_i := \mathsf{ExpandMask}(\rho', \kappa)$
    $\hat{y}_i := \mathsf{NTT}(y_i)$
    $\hat{w}_j := \hat{w}_j + \hat{A}_{j,i} \circ \hat{y}_i$ for $0 \le j < k$
$0 \le i < \ell$
    $w_i := \mathsf{NTT}^{-1}(\hat{w}_i)$
    $w_{1,i} := \mathsf{Highbits}(w_i)$
    ▷ store packed $\mathbf{w_1}$ in output buffer
$\tilde{c} := \mathsf{H}(\mu\|\mathbf{w_1})$ ▷ write $\tilde{c}$ to signature
$c := \mathsf{SampleInBall}(\tilde{c})$
▷ make 16-bit $c$ and $s_{1,i}$ polynomials
$\hat{c} := \mathsf{NTT}_{q'}(c); \hat{s}_{1,i} = \mathsf{NTT}_{q'}(s_{1,i})$
$\hat{T}_0 := \hat{c} \circ \hat{s}_{1,i}$
$0 \le i < k$
    $T_0 := \mathsf{NTT}_{q'}^{-1}(\hat{T}_0)$
    ▷ sample (using ExpandMask) and
      and add $y_i$ on-the-fly
    $z_i := T_0 + y_i$
    check $\|z_i\|_\infty < \gamma_1 - \beta$
    write $z_i$ to signature
$0 \le i < k$
    ▷ make 16-bit $c$ and $s_{2,i}$ polynomials
    $\hat{c} := \mathsf{NTT}_{q'}(c); \hat{s}_{2,i} = \mathsf{NTT}_{q'}(s_{2,i})$
    $\hat{T}_1 := \hat{c} \circ \hat{s}_{2,i}$
    $T_1 := \mathsf{NTT}_{q'}^{-1}(\hat{T}_1)$
    check $\|\mathsf{LowBits}_q(w_i - T_1, 2\gamma_2)\|_\infty < \gamma_2 - \beta$
    $w_i - cs_{2,i} := w_i - T_1$
$0 \le i < k$
    $T_2 := c \cdot t_{0,i}$ ▷ schoolbook multiplication
    check $\|T_2\|_\infty < \gamma_2$
    $h_i := \mathsf{MakeHint}(-T_2, w_i - cs_{2,i} + T_2, 2\gamma_2)$
    write $h_i$ to output
```

(Re-)generate matrix A and y on-the-fly: ~~80 KB~~ → 45 KB

Compress w: ~~45 KB~~ → 43.5 KB

Compressing multiplications ~~43.5 KB~~ → 26.5 KB

Variable Allocation:

Total of

$64 + 64 + 768k + 1024 + 208 + 68$ bytes → 5268 bytes

In practice: 6.5 KB needed

DILITHIUM SIGNATURE GENERATION: LOW-MEMORY VERSION

Variant			Dilithium-3					
			KiB			Cc		
With asm	[7]	K	59.6			2,835		
		S	72.3			6,742		
		V	56.6			2,700		
	[1]	K	59.6			2,830		
		S	67.4			6,624		
		V	56.6			2,692		
C only	PQClean	K	59.4			3,504		
		S	77.7			12,987		
		V	56.4			3,666		
	New	K	6.4	9.3x	9.3x	5,112	1.8x	1.5x
		S	6.5	10.4x	12.0x	36,303	5.5x	2.8x
		V	2.7	21.0x	20.9x	7,249	2.7x	2.0x

Smaller

Slower

PQC Use-Cases

AUTOMOTIVE

70%

70% connected cars by 2025

INDUSTRIAL & IOT

12B

IoT Edge & end nodes from 6B units in ‘21 to 12B units in ‘25

MOBILE

60B

Tagging 60B products per year by 2025

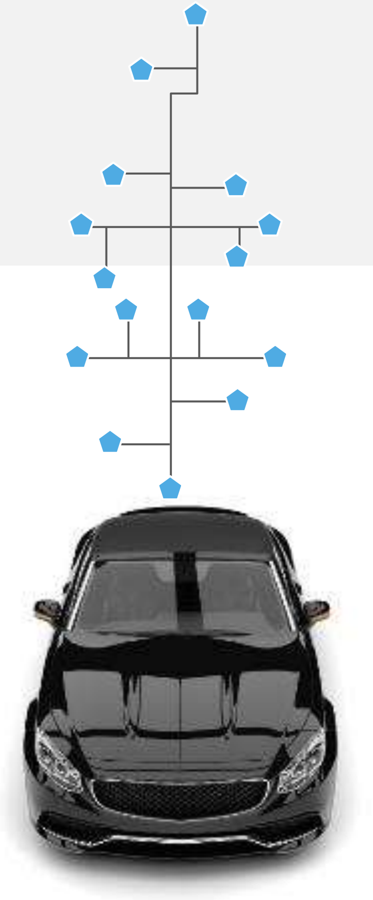
COMMUNICATION INFRASTRUCTURE

40B

Secure anchors & services for 40B processors

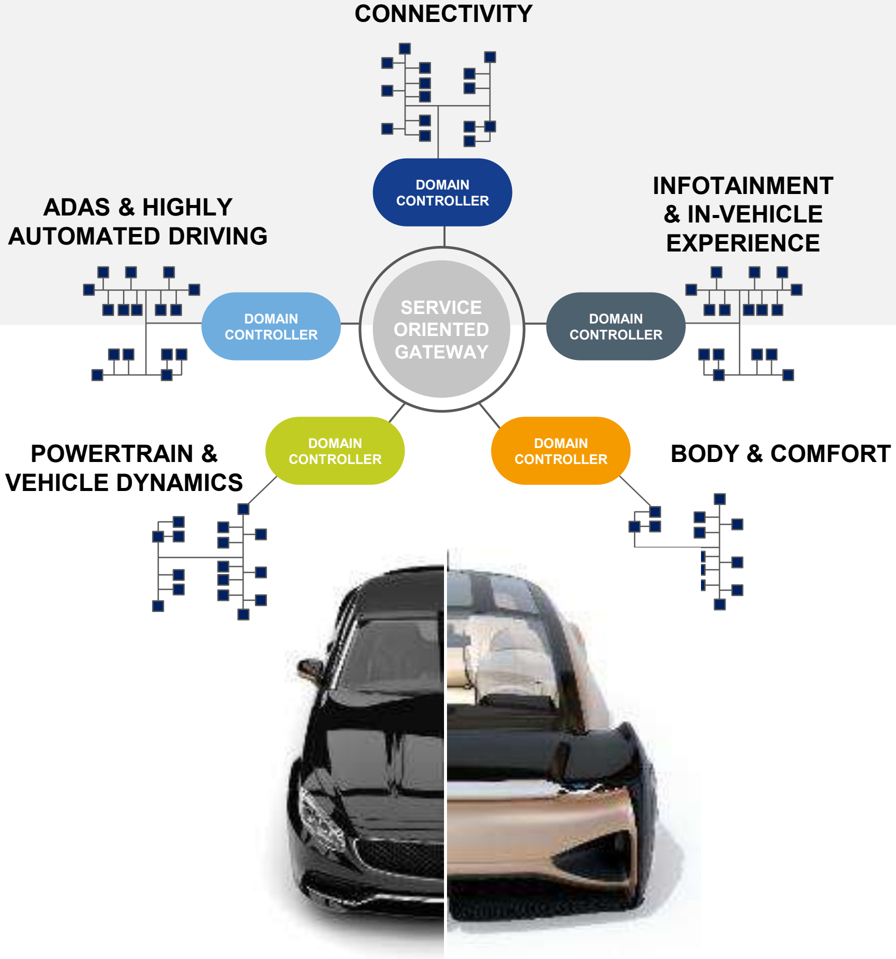
1. Source: NXP, Strategy Analytics, Evercore, Ericsson, IDTechex,

VEHICLE ARCHITECTURE TRANSFORMATION



TODAY | FLAT

UNFIT FOR FUTURE MOBILITY



CONNECTIVITY

DOMAIN CONTROLLER

ADAS & HIGHLY AUTOMATED DRIVING

DOMAIN CONTROLLER

SERVICE ORIENTED GATEWAY

INFOTAINMENT & IN-VEHICLE EXPERIENCE

DOMAIN CONTROLLER

POWERTRAIN & VEHICLE DYNAMICS

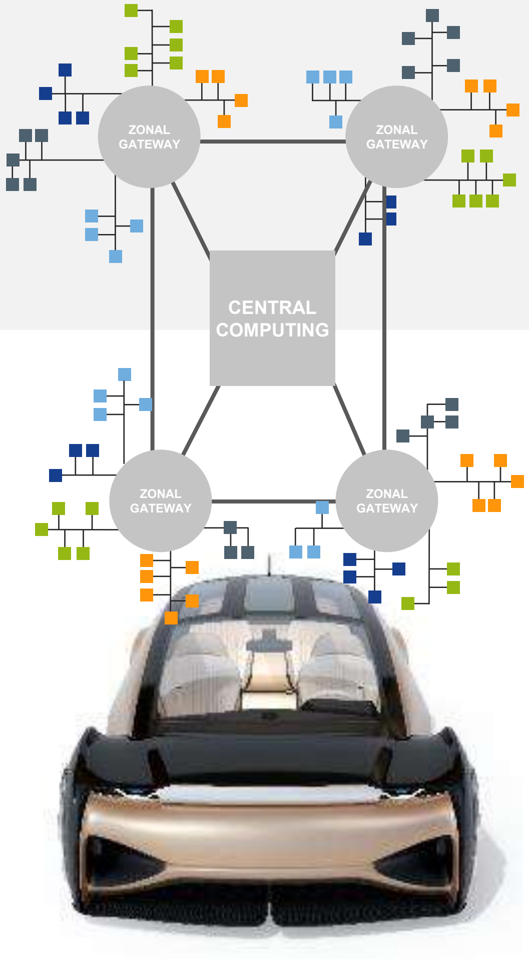
DOMAIN CONTROLLER

BODY & COMFORT

DOMAIN CONTROLLER

LOGICAL RESTRUCTURE | DOMAINS

ENABLING AUTONOMOUS CAR



ZONAL GATEWAY

ZONAL GATEWAY

CENTRAL COMPUTING

ZONAL GATEWAY

ZONAL GATEWAY

PHYSICAL RESTRUCTURE | ZONES

ENABLING USER-DEFINED CAR

NXP S32G2 VEHICLE NETWORK PROCESSOR WITH PQC INTEGRATION

OUR TARGET PLATFORM: S32G274A

3 Lockstep Arm® Cortex®-M7 Microcontrollers

4 Cluster Lockstep Cortex-A53 Microprocessors

8 MB of System RAM

Network Accelerators (LLCE/PFE)

Hardware Security Engine (HSE)

ASIL D Functional Safety Support

POST-QUANTUM CRYPTO

Integrate PQC secure signature verification

Enable PQC secure boot

Secure Over-the-Air (OTA) updates

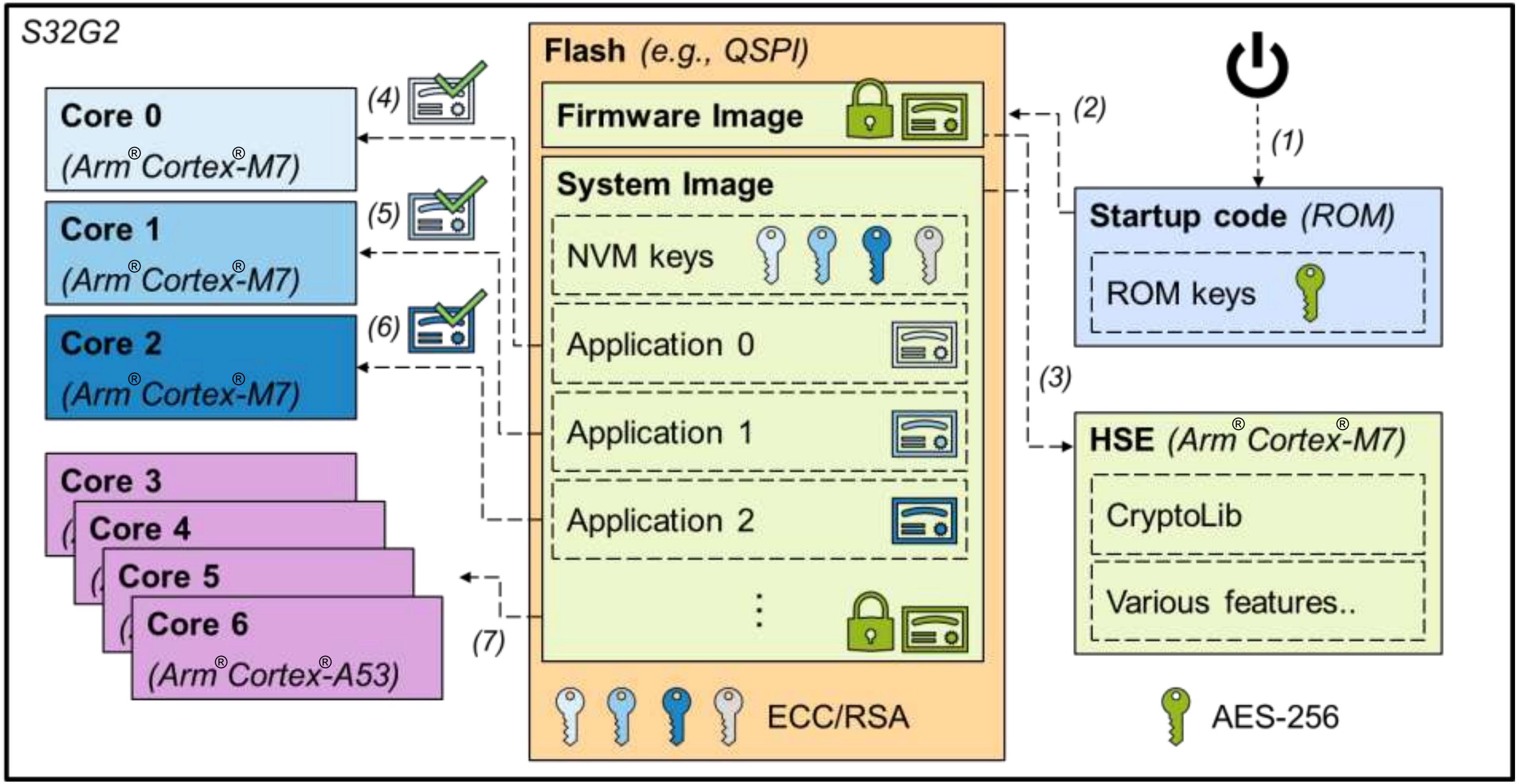
Secure vehicle and driver data

SAFE ASSURE™ by NXP

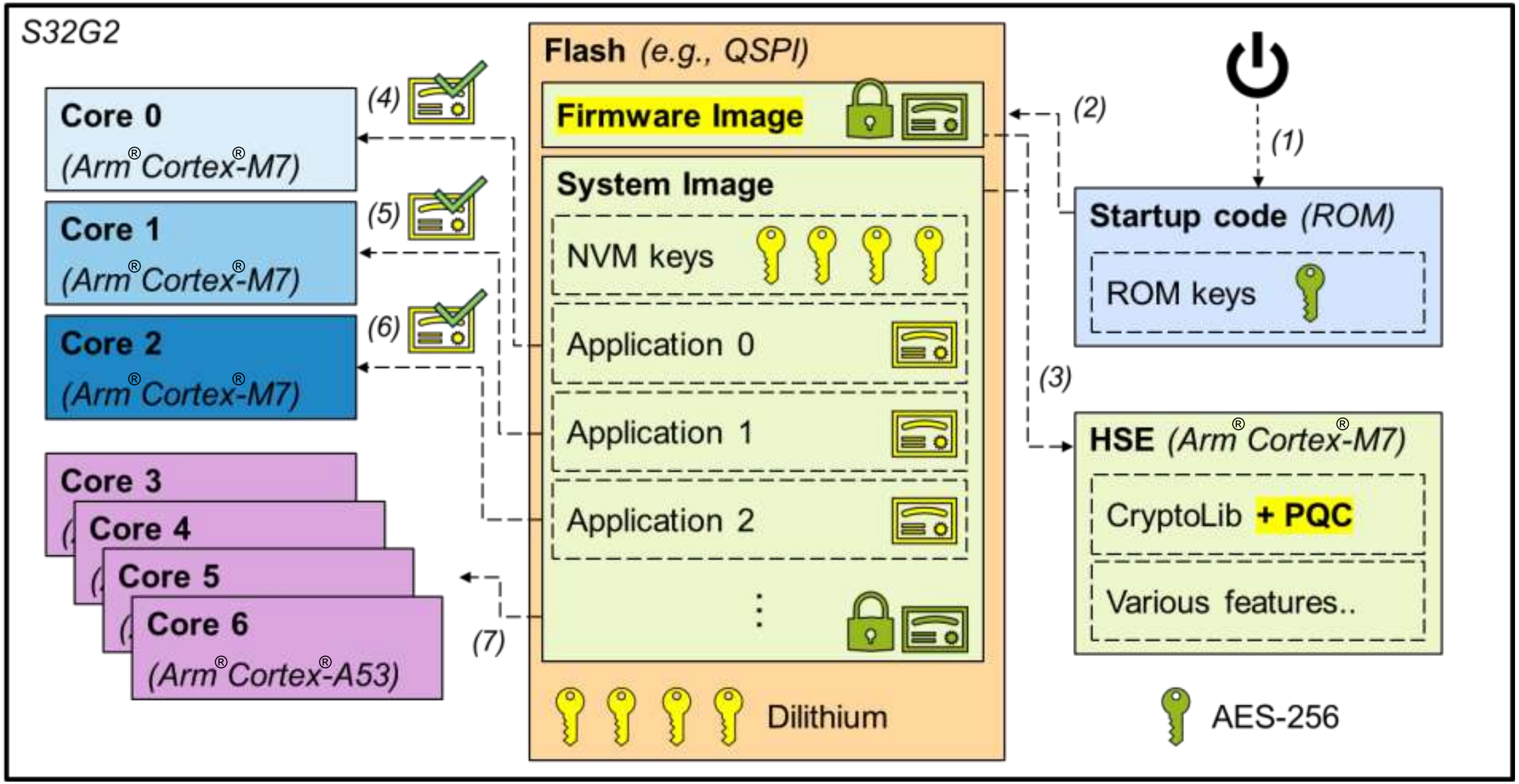
www.nxp.com/S32G2

EDGELOCK™ ASSURANCE by NXP

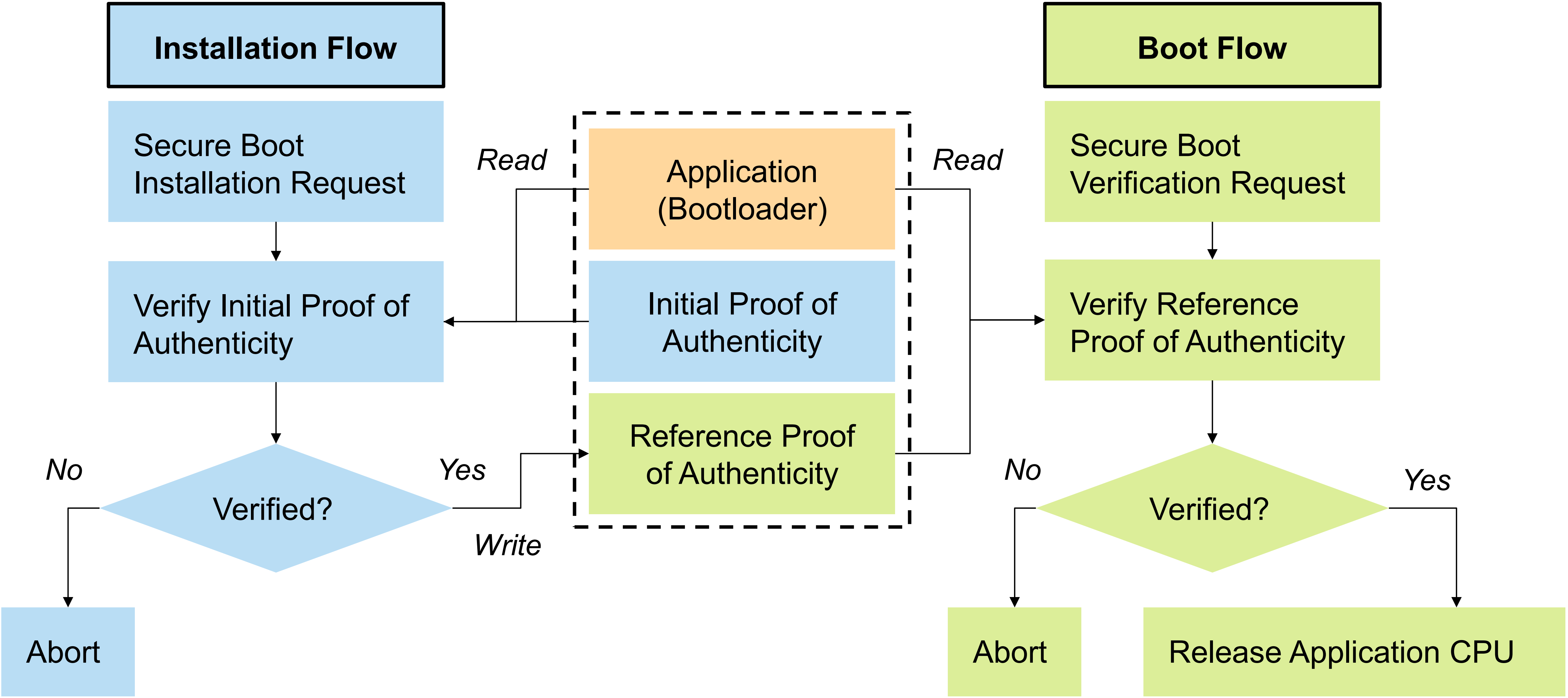
PQC DEMO: HSE SECURE BOOT OVERVIEW



PQC DEMO: HSE SECURE BOOT OVERVIEW



S32G2 INSTALL VS BOOT (CONFIGURABLE)



Installation Flow
Secure Boot Installation Request
Verify Initial Proof of Authenticity
No
Verified?
Yes
Write
Abort
Read
Application (Bootloader)
Initial Proof of Authenticity
Reference Proof of Authenticity
Read
Boot Flow
Secure Boot Verification Request
Verify Reference Proof of Authenticity
No
Verified?
Yes
Abort
Release Application CPU

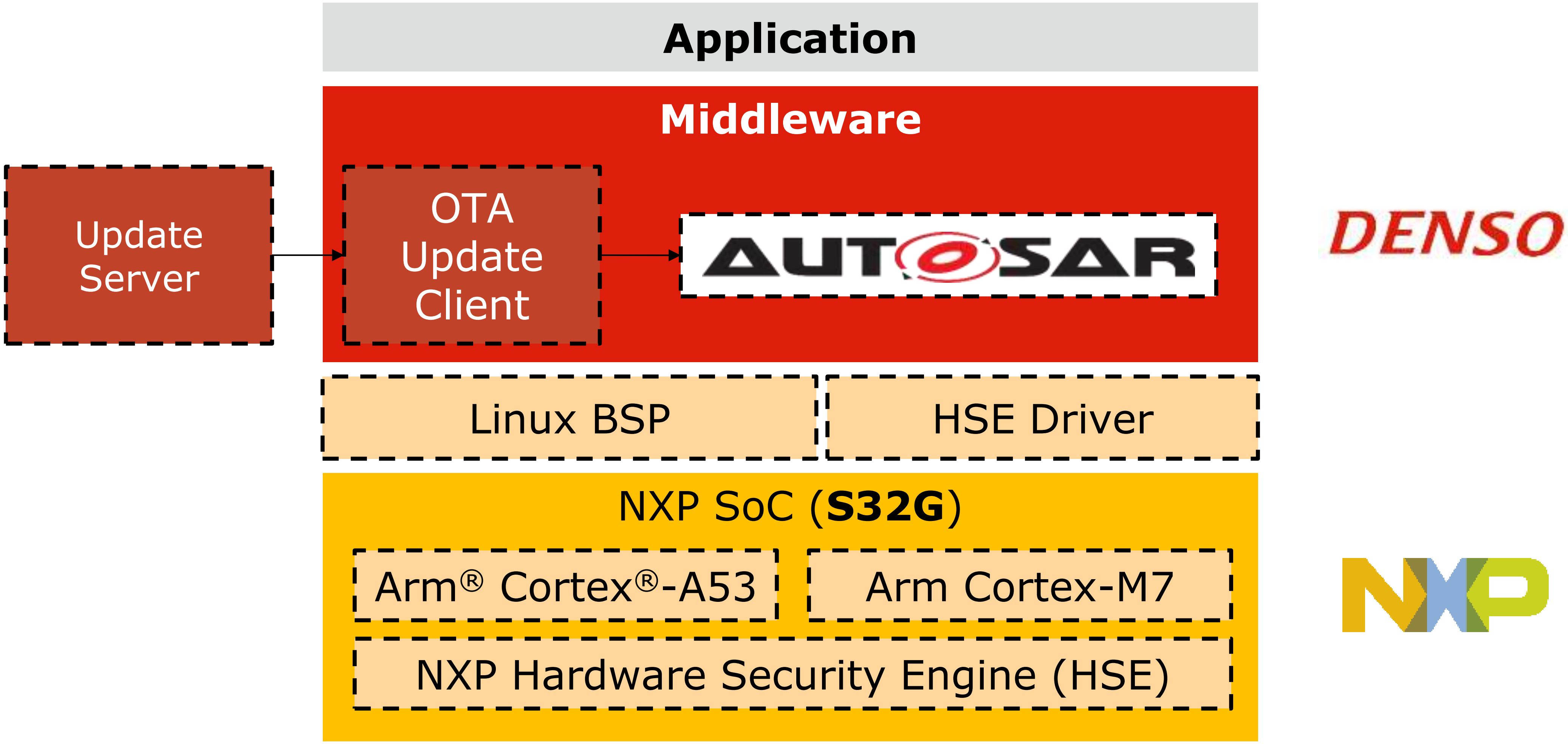
BENCHMARKS FOR AUTHENTICATION OF FW SIGNATURE ON THE S32G2

Alg.	Size		Performance (ms)			
			1 KB		128 KB	
	PK	Sig.	Inst.	Boot	Inst.	Boot
RSA 4K	512	512	2.6	0.0	2.7	0.2
ECDSA-p256	64	64	6.2	0.0	6.4	0.2
Dilithium-3	**1952**	**3293**	**16.7**	**0.0**	**16.9**	**0.2**

- Demonstrator only, further optimizations are possible (such as hardware accelerated SHA-3)
- Signature verification only required once for installation!
- During boot the signature verification can be replaced with a check of the Reference Proof of Authenticity

Bos, Carlson, Renes, Rotaru, Sprenkels, Waters: Post-Quantum Secure Boot on Vehicle Network Processors. Embedded Security in Cars. Escar 2022

NXP + DENSO : PQC SECURE OVER-THE-AIR (OTA) UPDATE



Joppe W. Bos, Alexander Dima, Alexander Kiening and Joost Renes: Post-Quantum Secure Over-the-Air Update of Automotive Systems. Cryptology ePrint Archive, Report 2023/965, IACR, 2023.

CONCLUSIONS

- Migration to PQC is a difficult & hot topic
- Many practical challenges
  - Memory
  - Available hardware (co-processors)
  - Efficient side-channel countermeasures

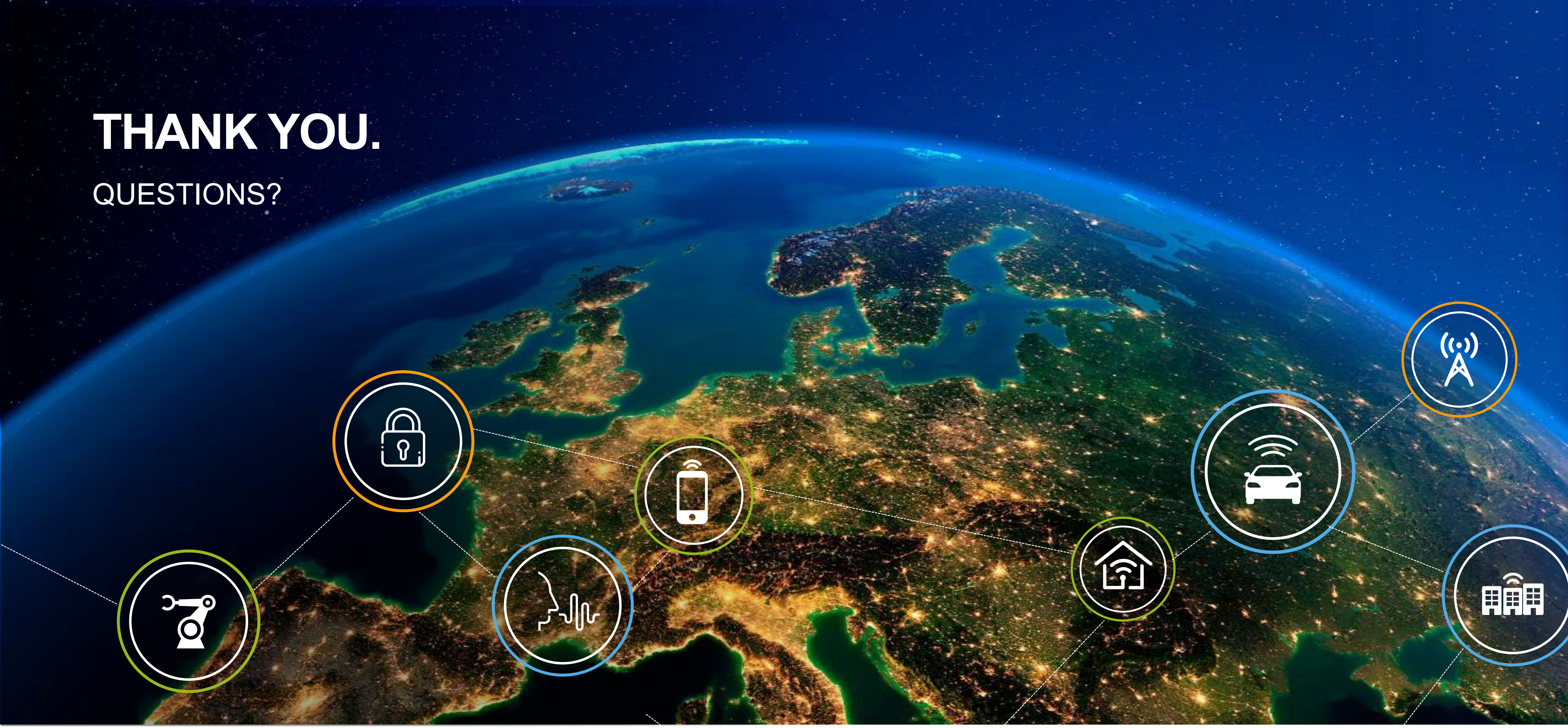
For automotive

✓ Large key sizes no issue, marginal increase in stack usage

- SHA-3 performance crucial, hardware acceleration important
- Little impact on OTA time (verification time not critical)
- Transition to PQC practical

THANK YOU.

QUESTIONS?



SECURE CONNECTIONS
FOR A SMARTER WORLD

SECURE CONNECTIONS
FOR A SMARTER WORLD