PQC Transition in Hardware: **Processors, SoCs, IoT, Secure Elements**

Markku-Juhani O. Saarinen
<markku-juhani.saarinen@tuni.fi>

18 November 2024
AusQRC, Melbourne

Tampereen yliopisto
Tampere University

RISC-V

Hello! I'm Markku-Juhani O. Saarinen 👋

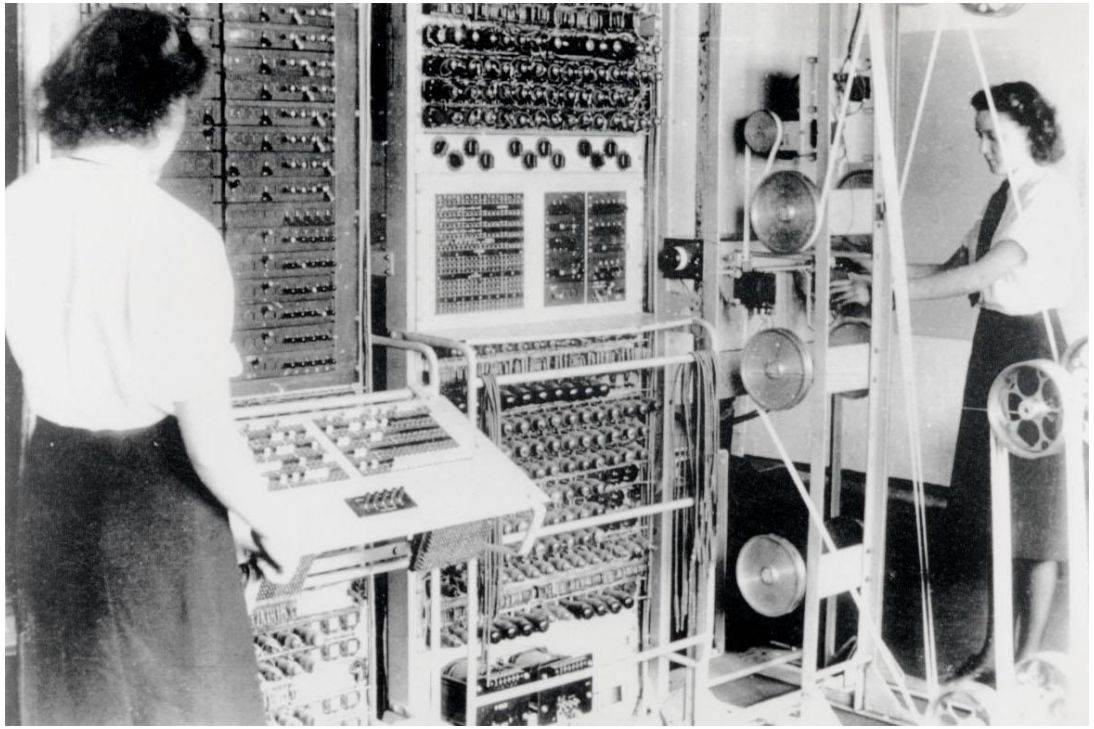
Some random biographical things:

- Started as a cryptographer in 1997 at SSH Communications Security. Helped create SSH2.
- Moved to technical consulting (mainly in the Middle East) + Pentest gigs, PCI DSS audits.
- Got bored & went back to school. PhD Royal Holloway 2009 (hash function cryptanalysis.)
- Post-doc periods and more industry gigs followed. Post-Quantum stuff since ~2015.
- First employee at PQShield Ltd, Oxford UK in 2018. Architected, tinkered, prototyped, patented, and helped license hardware PQC modules to semiconductor companies.
- RISC-V Stuff since 2019. I designed some of the now-standard crypto instructions.
- Chair, **RISC-V International** Post-Quantum Cryptography Task Group (**RVI PQC TG**).
- Drifted back to **Finland** in 2023-24, now a **Professor of Practice** at **Tampere University**.
- Program Co-Chair, **PQCrypto 2025** (Taipei, Taiwan April 8-10, 2025): **See you there!**

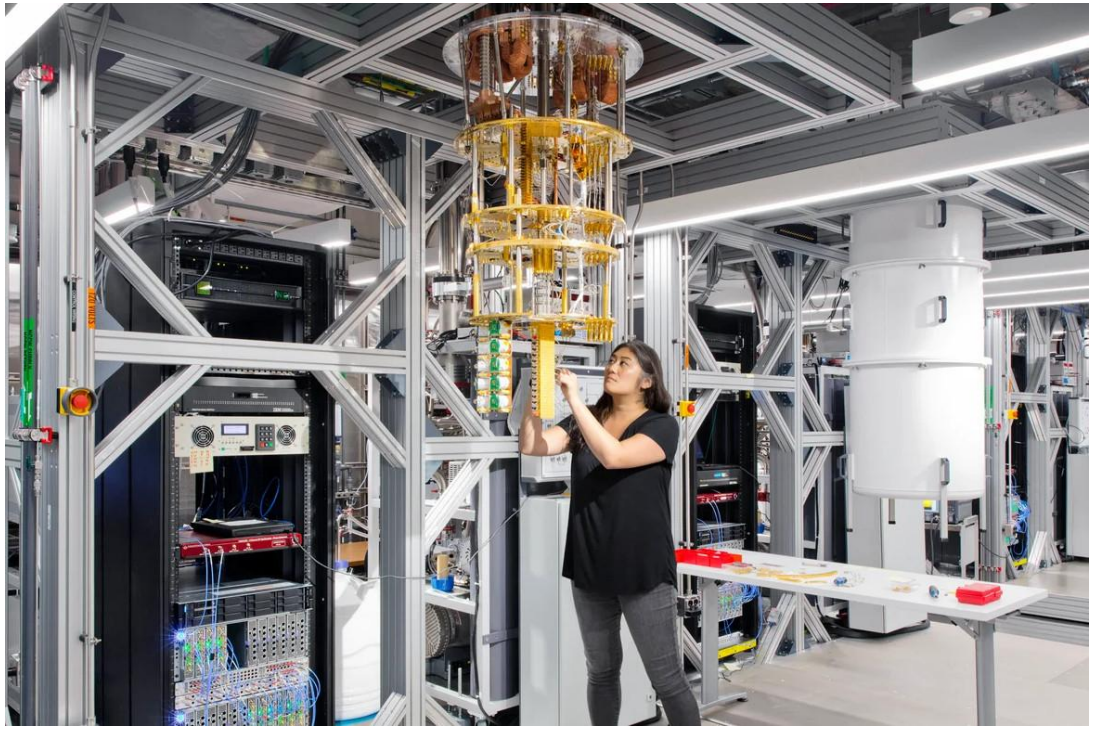
Rough Outline

1. **Intro**: Why Quantum Resilient / Post-Quantum Cryptography?
2. **International standards**: Main algorithms, protocols, and transition.
3. **Processors:** It’s not just the software stack – Impact on chips, computers.
4. **Secure Elements:** About testing of high-assurance crypto modules.

Cryptanalysis and Computing



Secret development of the Colossus digital computer during WW2 allowed the British to break the Lorentz cipher and read high-level German army messages.



Sufficiently powerful quantum computers can break RSA and Elliptic Curve cryptography, the foundation of the security of the public Internet and e-commerce.

PQC Timeline (part I)

1970s-: Public key cryptography is invented, allowing private communication in public networks without the need for pre-established secret keys ("shared secrets") between parties.

.. while elsewhere ..

1980s-: First suggestions (Benioff, Feynman) to build "non-digital" quantum computers that use quantum states and other phenomena directly. Initially proposed just for quantum simulation.

1994: Peter Shor shows that factoring (RSA) and Discrete Logarithm / Elliptic Curve problems can be solved if a large quantum computer is built (in polynomial time - basically regardless of key size.)

.. quantum computing research continues, while at the same time ..

1990s-2000s: Public internet and mobile communication revolution; digital technologies become embedded in society, commerce, government. Digital identity (e.g., web site certificates) and online privacy / confidentiality (e.g., TLS) are entirely dependent on the security of public key cryptography.

Limitations of Quantum Computing

- Shor's 1994 (factoring and discrete logarithm) algorithm was one of the earliest found – but still one of only a handful of truly effective quantum algorithms.

- Simpler annealing-based or "variational" quantum computers (e.g., d-Wave) cannot be used to implement Shor's algorithm – no real threat to cryptography.

- General (secret key) search can be sped up using **Grover's algorithm**, but it has **$O(2^{n/2})$** exponential complexity and very large overheads.

- Attacks on AES-128 or SHA-256 do not seem presently feasible with quantum computers; current standard symmetric cryptography is considered safe.

GCHQ's 2024 estimates: https://csrc.nist.gov/Presentations/2024/practical-cost-of-grover-for-aes-key-recovery

Some concrete numbers – Steady progress suffices

[Gidney & Ekerå 2019] design for Shor's on **RSA-n** with superconducting QC:

n (3 + 0.002 lg n)	*Logical / abstract qubits (**2n** is also possible)*
logical qubits × $2(d + 1)^2$	*Physical qubits; d = code dist. = 27 for n=2048*
$\mathbf{n^2}$ (500 + lg n)	*Toffoli gates ("arithmetic ops")*
$\mathbf{n^3}$ (0.3 + 0.0005 lg n)	*Measurement depth ("time")*

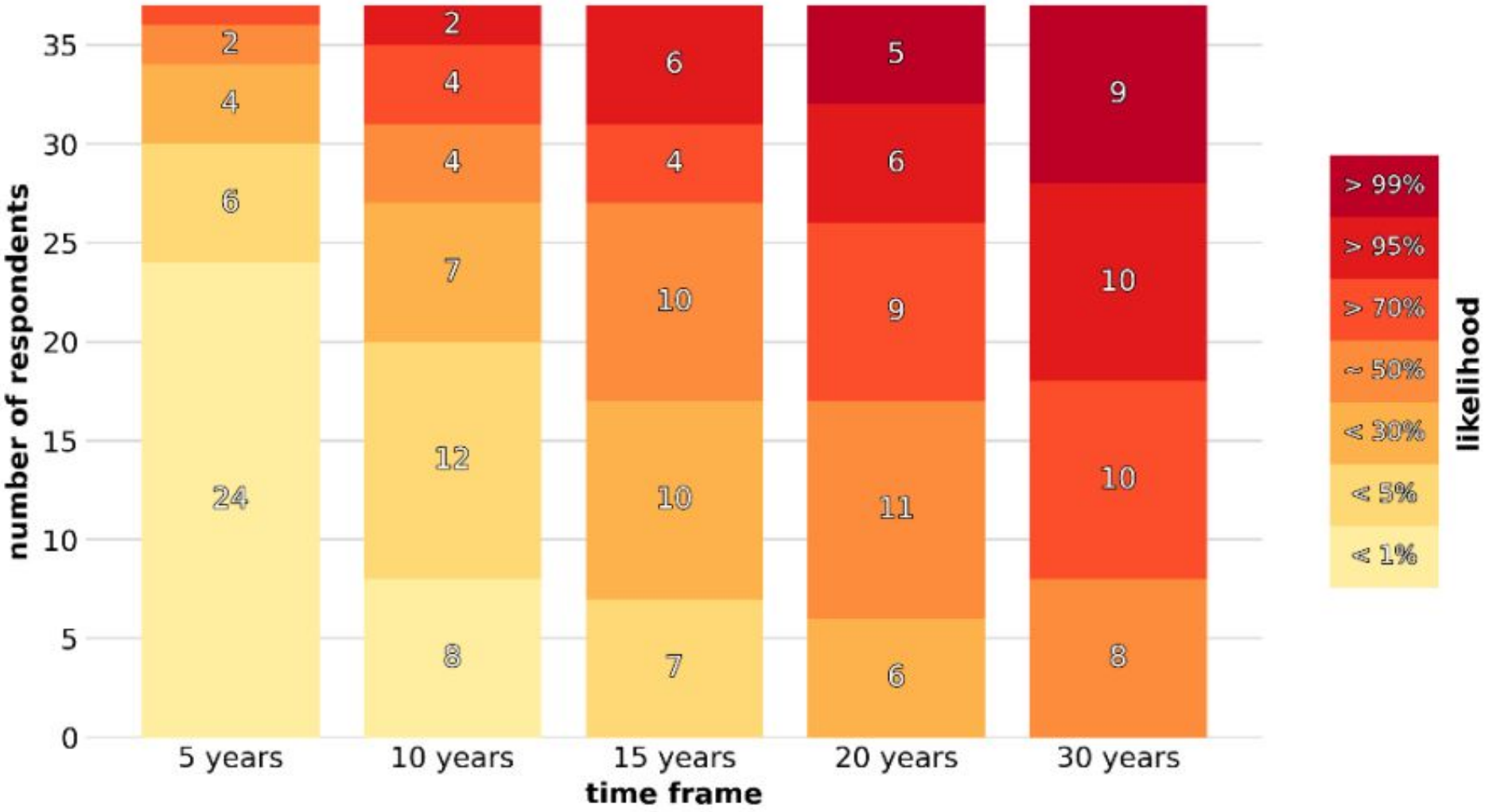
[Häner et al., 2020] estimate **8n** + 10.2 lg n logical qubits for an n-bit elliptic curve. Breaking Elliptic Curves appears easier at similar classical security level.

For quantum threat to materialize, exponential technology improvement ("Moore's Law" for traditional semiconductors – since 1965) **is not required**.

[Gidney, Ekerå, 2019] https://arxiv.org/abs/1905.09749 [Häner et al., 2020] https://arxiv.org/abs/2001.09580

2023 EXPERTS' ESTIMATES OF LIKELIHOOD OF A QUANTUM COMPUTER ABLE TO BREAK RSA-2048 IN 24 HOURS

Number of experts who indicated a certain likelihood in each indicated timeframe



https://globalriskinstitute.org/publication/2023-quantum-threat-timeline-report/

PQC Timeline (part II)

Late 2000s: ``Post-Quantum Cryptograpy'' is born: hash-based, code-based, lattice-based, and multivariate families had been identified as potential replacements to RSA and ECC by 2009.

2015: U.S. National Security Agency (NSA) CNSS Advisory Memorandum 02-15. Indicates a long-term requirement for quantum-resistant cryptography standards. Standardization is initiated.

2016: National Institute for Standards and Technology (NIST) starts an open, international standardization and evaluation process for PQC algorithms (digital signature and key establishment.)

82 submissions by 30 Nov 2017 deadline. First selections after three evaluation rounds in July 2022.

2024: Ratified specs (FIPS 203, FIPS 204, FIPS 205) in August 2024. Standards immediately in effect.

https://csrc.nist.gov/news/2024/postquantum-cryptography-fips-approved

“It’s the law” (well, in United States it is)

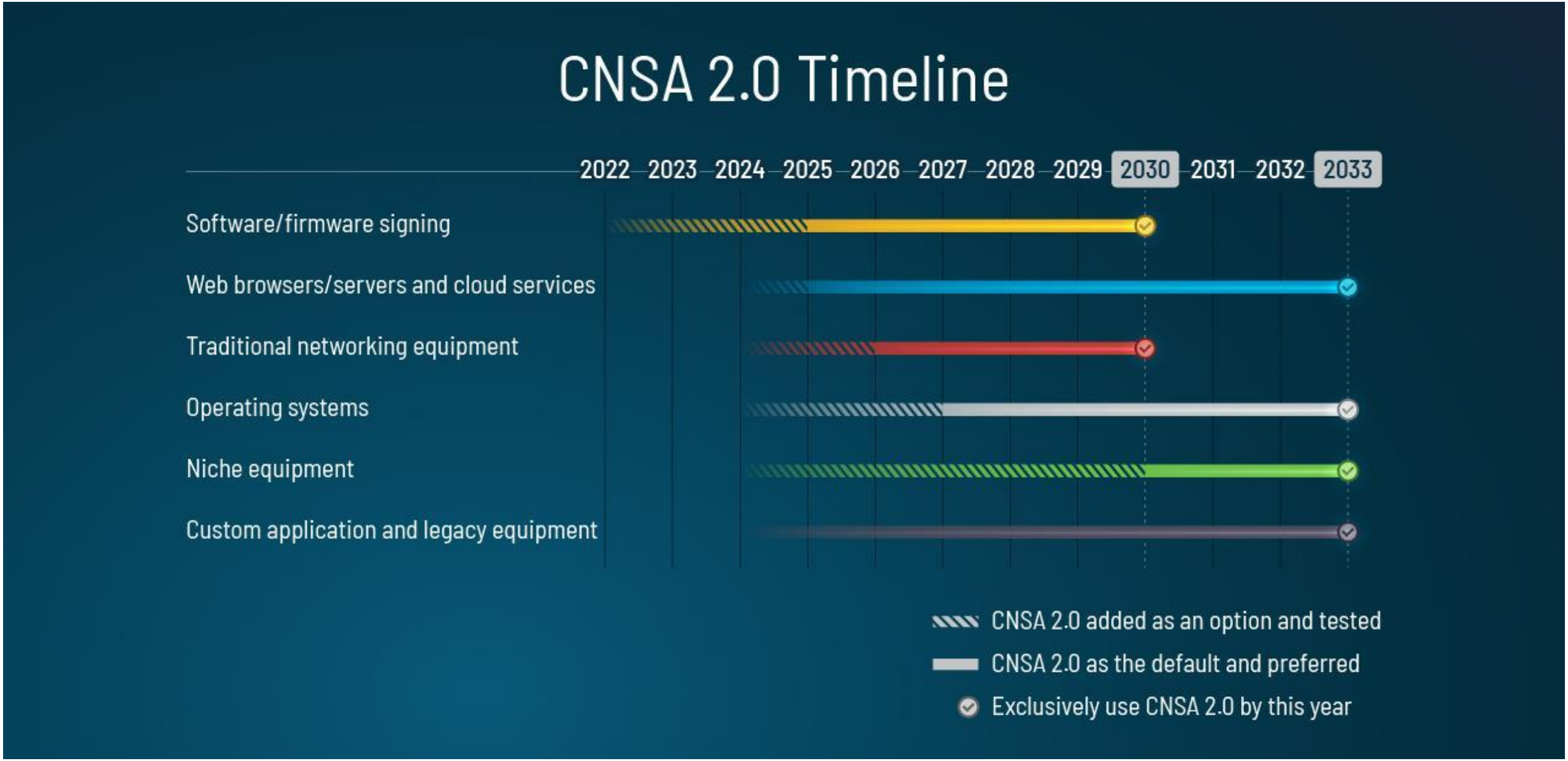
NSM-8 (Jan 2022): *“On Improving the Cybersecurity of National Security, Department of Defense, and Intelligence Community Systems”*

NSM-10 (May 2022): *“On Promoting United States Leadership in Quantum Computing While Mitigating Risks to Vulnerable Cryptographic Systems”*

HR 7535 (Dec 2022): *“Quantum Computing Cybersecurity Preparedness Act”*

<u>These PQC-related National Security Memorandums and the Public Law:</u>

- Mandates transition to Post-Quantum Cryptography in government IT.
- Assigns inventory, reporting responsibilities, sets timelines, etc.
- Outside Government’s own IT systems and some critical sectors, the use post-quantum cryptography (like most information security) is of course not enforced, apart from self-regulation and “business best practices” in many industries.



https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF

NIST IR 8547 (Draft, November 2024)

Table 2: Quantum-vulnerable digital signature algorithms

Digital Signature Algorithm Family	Parameters	Transition
ECDSA [FIPS186]	112 bits of security strength	***Deprecated*** after 2030 ***Disallowed*** after 2035
	≥ 128 bits of security strength	***Disallowed*** after 2035
EdDSA [FIPS186]	≥ 128 bits of security strength	***Disallowed*** after 2035
RSA [FIPS186]	112 bits of security strength	***Deprecated*** after 2030 ***Disallowed*** after 2035
	≥ 128 bits of security strength	***Disallowed*** after 2035

https://doi.org/10.6028/NIST.IR.8547.ipd

Main Replacements: NIST PQC Standards

KEY ESTABLISHMENT

DIGITAL SIGNATURES

Kyber: FIPS 203 ML-KEM (2024)

Primary PQC **key establishment** algorithm to replace Diffie-Hellman (ECDH) key exchange and RSA public-key encryption. Lattice-based.

Some of **{ HQC, BIKE**, **Classic McEliece }** (2025?)

Being evaluated in "Round 4." Code-based key establishment algorithms. Longer public keys.

Hybrid schemes: One still needs to support traditional Elliptic Curve and RSA methods.

Dilithium: FIPS 204 ML-DSA (2024)

Primary "general-purpose" PQC **signature algorithm** to replace ECDSA, RSA signatures. Lattice-based.

XMSS and **LMS**: NIST SP 800-208 (2020)

SPHINCS+: FIPS 205 SLH-DSA (2024)

Hash-based signatures; Firmware signing.

Falcon: FIPS 206 FN-DSA (2025). Lattice-based.

Signature "On-Ramp" Algorithms (2026?)

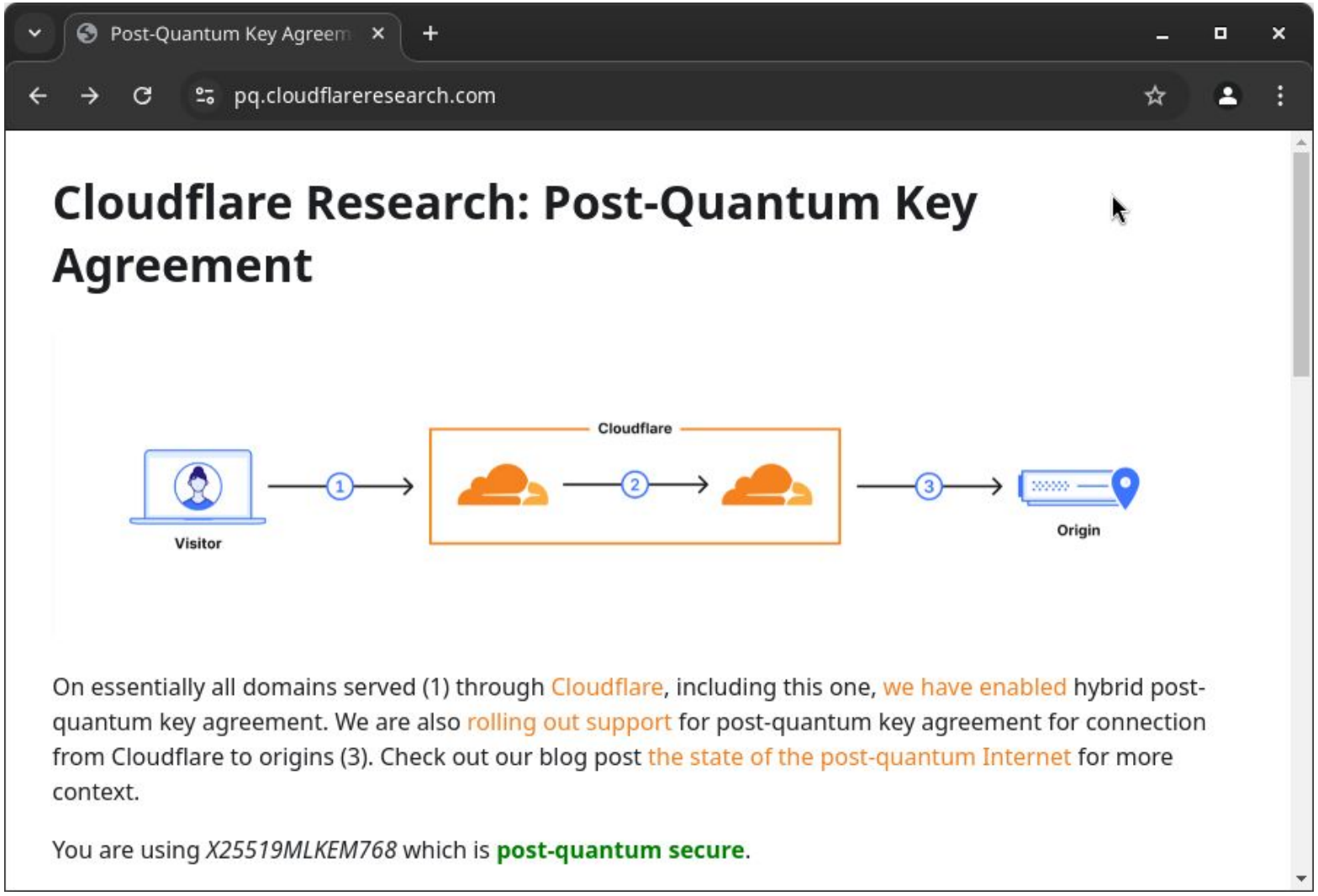
Kyber & Dilithium: Basic Characteristics

- **Performance**: Roughly as fast, sometimes even faster than RSA and Elliptic Curve crypto when implemented in software on a typical PC / mobile device.

- **Longer public keys, ciphertexts, and signatures:**
  - Elliptic Curves: 32..132 bytes, RSA: 256..1024 bytes
  - Kyber and Dilithium: 800..4896 bytes, depending on security level.

- **PQC Integration in Internet standards (TLS and PKI) is fairly advanced.**
  - Mostly Traditional + PQC “hybrids” in near future.
  - Preliminary versions already in production use: https://datatracker.ietf.org/doc/draft-kwiatkowski-tls-ecdhe-mlkem/

Try it out: TLS in Chrome 131+, Firefox 132+



https://pq.cloudflareresearch.com/

Note on **QKD** – "Opposite of mandatory"

Quantum Key Distribution (QKD) is a method of transmitting secrets that is not based on cryptography, but on physical/quantum protection of transmission "wires." QKD may be suitable for niche use cases but can not support mobile communications or long-distance end-to-end security.

Quantum Random Number Generators (QRNG) use direct measurements of quantum phenomena to create random bits. This is not required for security against quantum computers.

Some QRNGs *could* be hardened and certified as True Random Number Generators (TRNGs / RBGs), and hence allowed for use in PQC and communications security. QRNG vendors very rarely do this.

Western governments do not support the use of QKD for secure communications at this time.

US: https://www.nsa.gov/Cybersecurity/Quantum-Key-Distribution-QKD-and-Quantum-Cryptography-QC

UK: https://www.ncsc.gov.uk/whitepaper/quantum-security-technologies

FR, DE, NL, SE: https://cyber.gouv.fr/sites/default/files/document/Quantum_Key_Distribution_Position_Paper.pdf

AU: https://www.cyber.gov.au/resources-business-and-government/governance-and-user-education/governance/planning-post-quantum-cryptography

Summary of Introduction

- **Post-Quantum Cryptography standards went into effect in August 2024.**

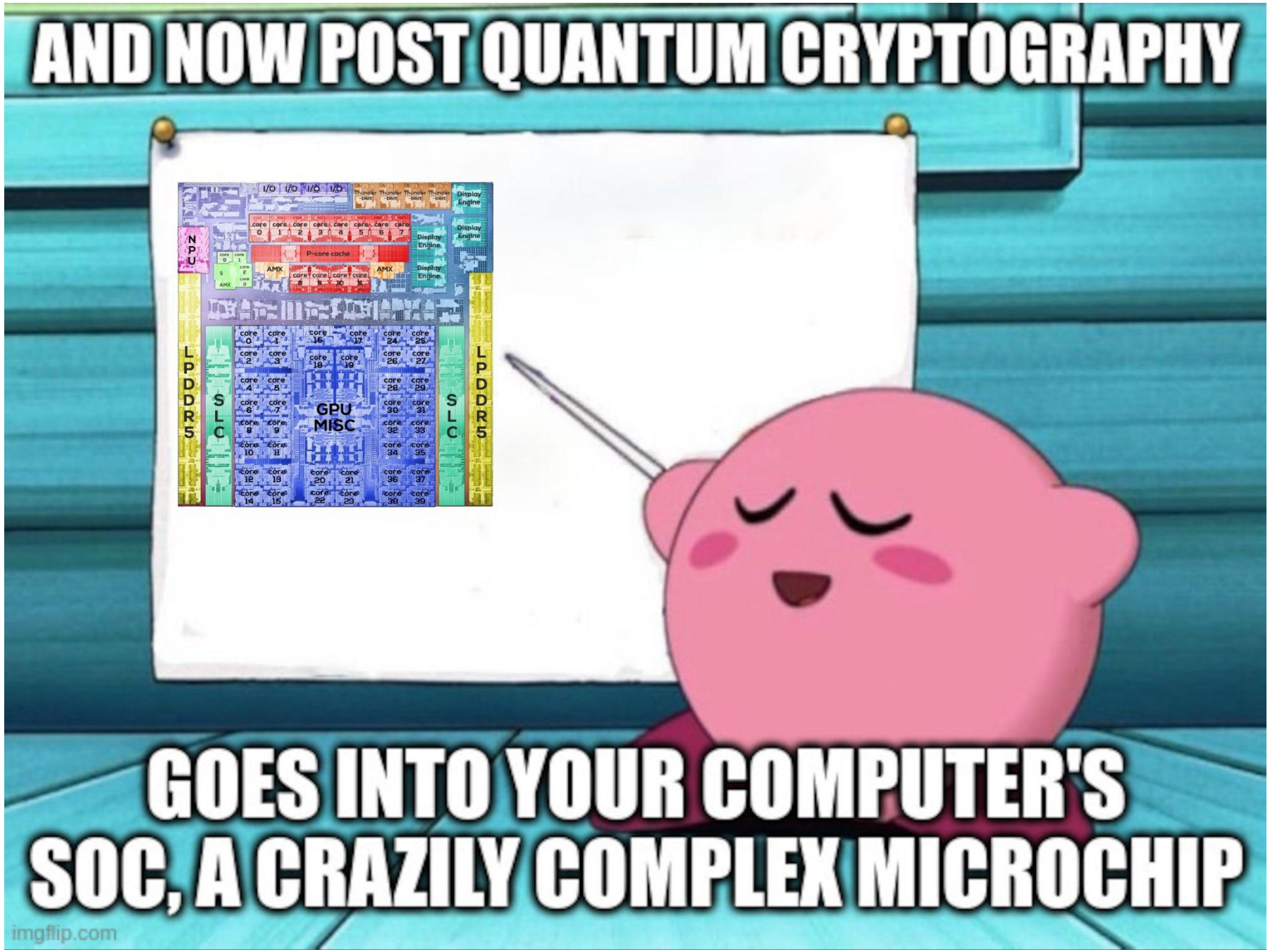
  Transition to quantum-secure cryptography is ongoing, and is becoming a compliance requirement (*old crypto depreciated perhaps by ~2035.*)

- **Hybrids**: { RSA, ECDH, ECDSA } + { ML-KEM “Kyber”, ML-DSA “Dilithium” }.

  Also: Hash-Based signature standards (SLH-DSA, LMS, XMSS) in some cases.

  Coming up: FN-DSA “Falcon”, Round 4 KEMs, Signature On-Ramp.

- PQC changes characteristics, but change is mostly hidden from end users.


AND NOW POST QUANTUM CRYPTOGRAPHY
I/O
Thunderbolt
Display Engine
NPU
P-core cache
AMX
LPDDR5
SLC
GPU MISC
GOES INTO YOUR COMPUTER'S SOC, A CRAZILY COMPLEX MICROCHIP
imgflip.com

Hardware View: Crypto functions on a SoC

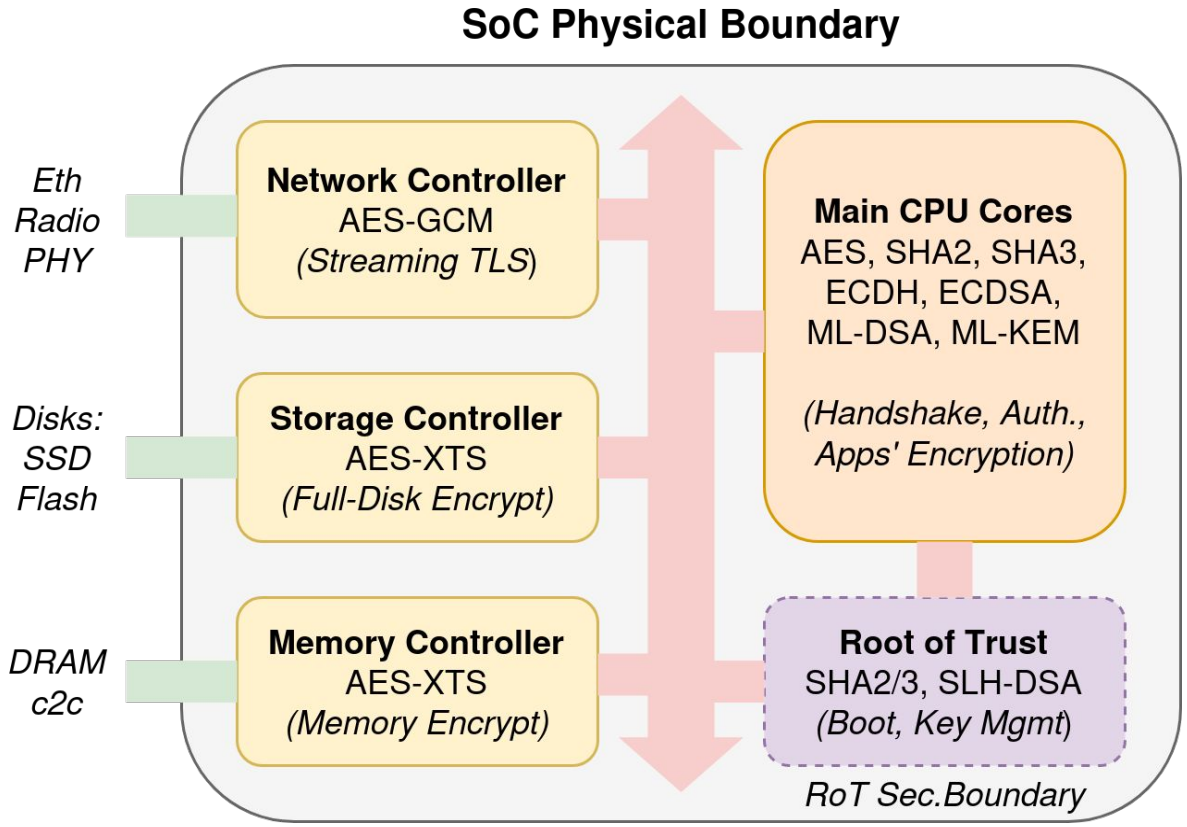
We often want to ***not use*** the main CPU for cryptography tasks..

Main CPU complex: TLS handshakes; asymmetric ops, X.509, crypto in apps. Can have ISA extensions up in the CPU.

Root of Trust (RoT): SoC-wide Platform Security. Isolated MCU + accelerators.

Disk or storage controller: E.g. AES-XTS.

Network Controller: Bulk symmetric encryption (e.g. TLS AES-GCM Frames).



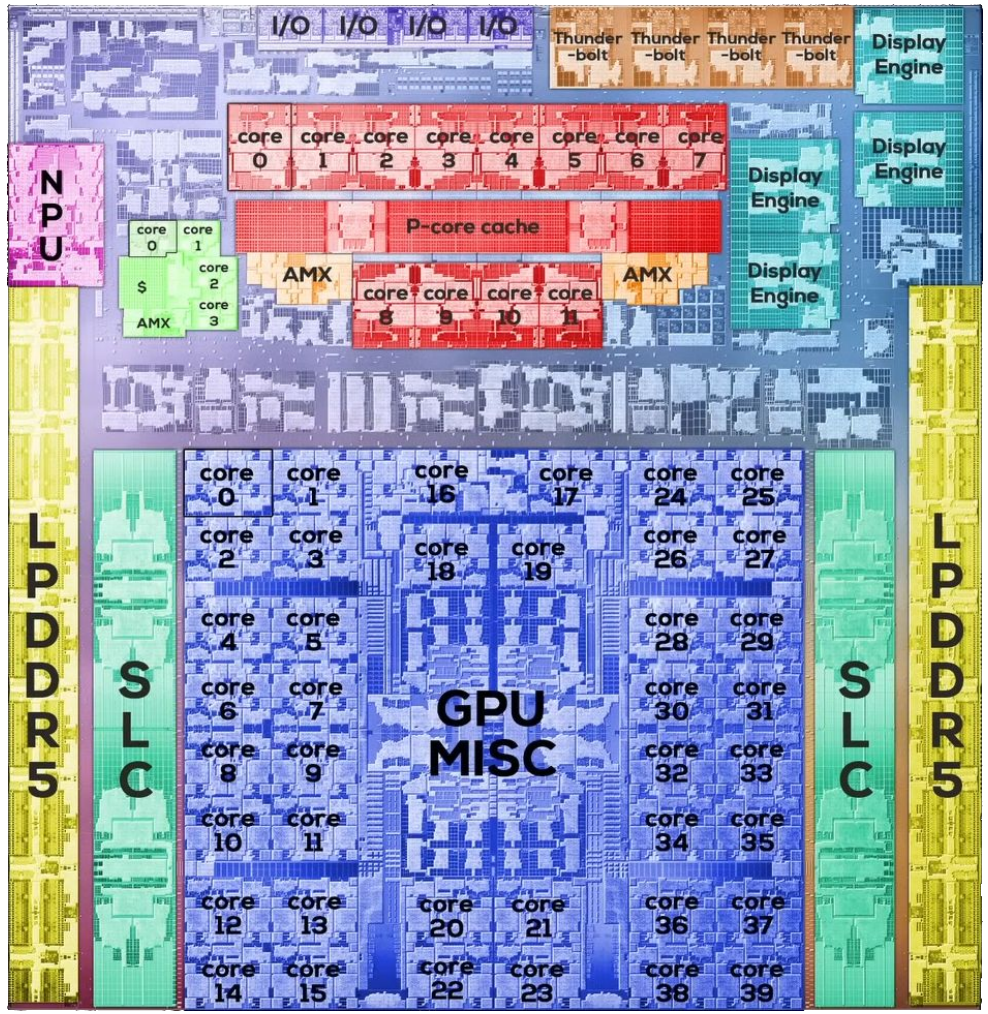
Application Processor PQC Support

These cores run the “visible” OS:

- In Kernel: IPSec, WireShark
- Sometimes also storage encryption
- Standard apps and libraries: TLS, QUIC
- OS tools, services: SSH, GnuPG
- User applications: Signal, WhatsApp, ..

Crypto acceleration is mostly done with instruction set features (for scalability).

Typically timing attack protection **only**.



How bad an extra hash can be?

By Sasha Frolov and Rafael Misoczki

- Key exchange is a (very) commonly performed operation at Meta
  - **Currently, ~0.05% of CPU cycles in Meta's data centers are spent doing X25519 key exchange**
  - We hope this data point is useful for making cost estimates while defining PQC standards specs
- This means
  - Deploying post-quantum key exchange has a non-negligible capacity cost
  - Apparently innocuous steps can cost hundreds of thousands or even millions of dollars a year
    - e.g. extra hashing steps, like hashing randomness or hashing parts of the transcript, which are being discussed as part of finalizing Kyber specification
    - Even if an extra step does not affect latency, the extra power usage/consumption of shared resources on highly parallel servers still has costs

Feedback? Write to sashafrolov@meta.com or rafam@meta.com.

Standard RISC-V Cryptography Extensions (“K”)

Done: **Scalar Crypto** (Ratified 2021)**:** AES, SHA2, SM3, SM4, CMUL (GCM) with 32- and 64-bit **scalar registers**. + "Constant time" & Entropy Source.

Done: **Vector Crypto** (Ratified 2023)**:** AES, SHA2, SM3, SM4, GCM with **vector registers**: Make bulk crypto even faster with *parallel* AES-GCM etc.

-> many of these now In Linux Kernel, OpenSSL, going into Android Platform

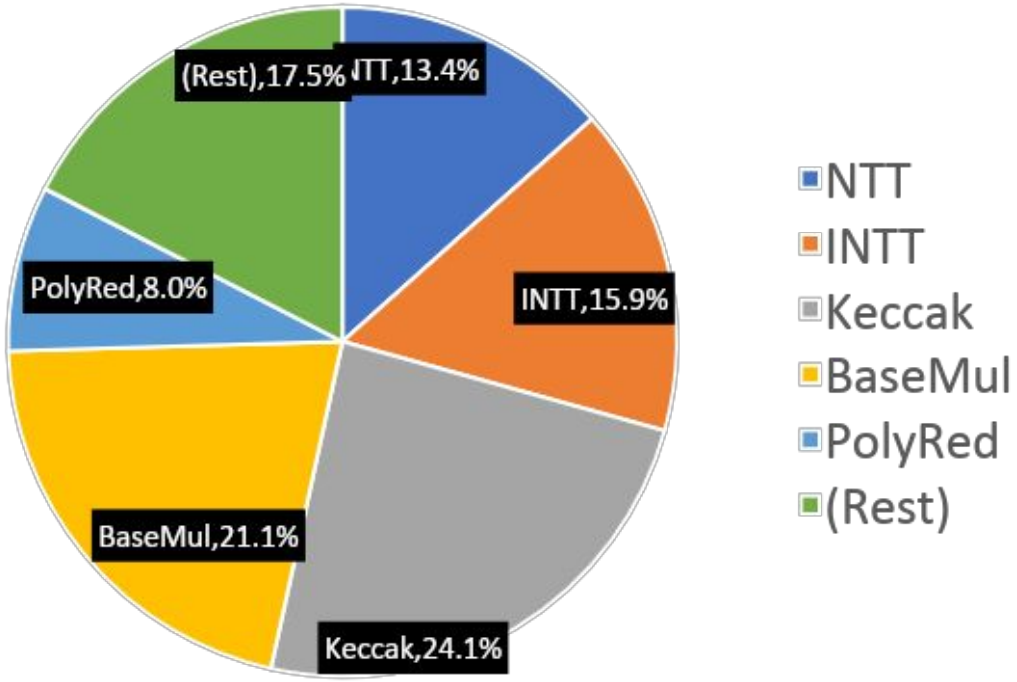
Being worked on:

High Assurance Crypto TG (From late 2023)**:** "Full-rounds" AES allowing emission/power side-channel security. Key management features.

Post-Quantum Crypto TG (From late 2023)**:** What can we do to assist standard PQC algs (notably FIPS 203,204,205 - Kyber, Dilithium, SPHINCS+) ?

Kyber Compute: Vectors (mod 3329) + Keccak

Reference Kyber-768: 2.26M Insn
KG 600k + Enc 734k + Dec 921k



Compute in NIST Lattice Crypto:

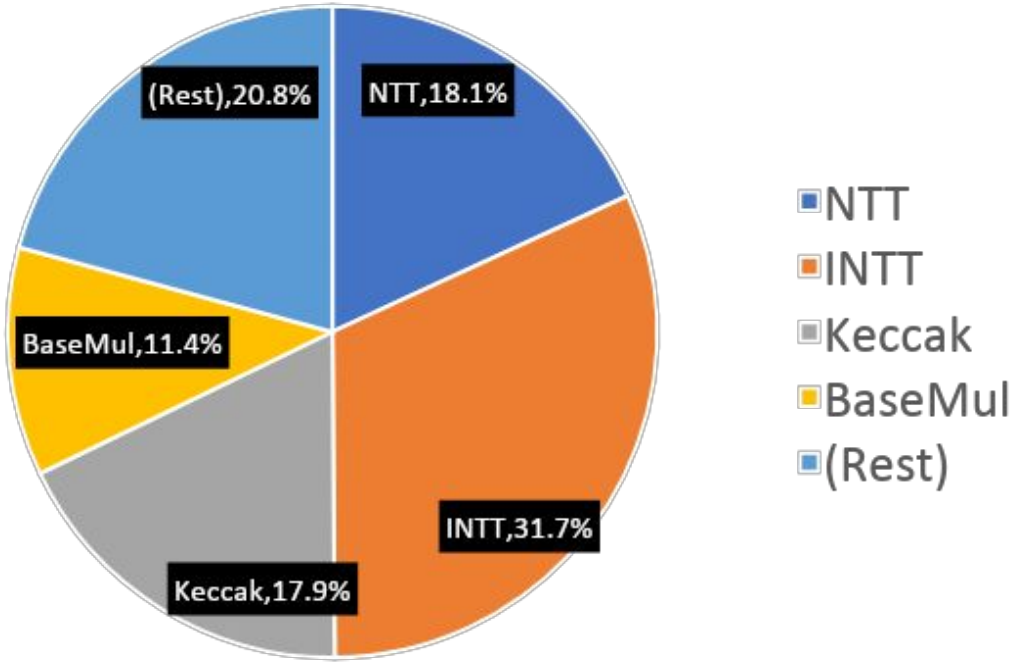
- **Keccak** i.e. SHA3/SHAKE operations. Typically well >50% of overall cycles.
- **Number Theoretic Transforms.** Vectorizable functions (256 x 16/32.)
- **Other polynomial arithmetic.** Mostly integer vectors; shifts, adds, sub.
- **Samplers** (rejection and CBD), rounding, "packing" (serialize).

Instret (with vlen:128,elen:64) - LLVM 18 snapshot, Oct 2023. `-Ofast -march=rv64gcv_zbb (zvk)`

Dilithium: Vectors (mod 8380417) + Keccak

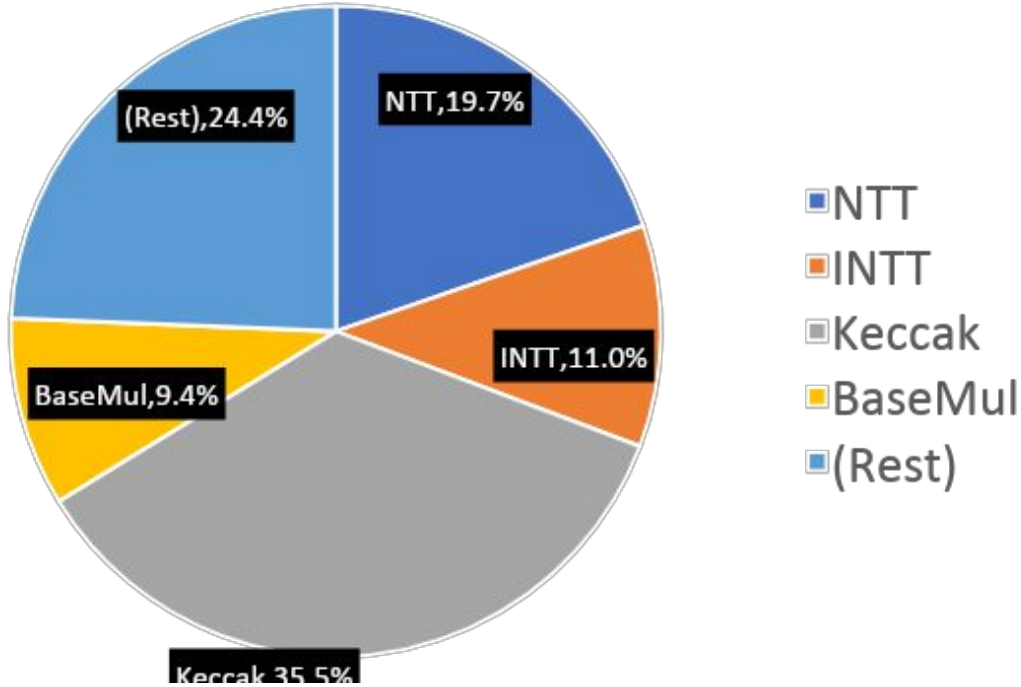
ML-DSA-44 Sign: Avg 4.60M Insn

ML-DSA-87 Sign: Avg 8.37M Insn



ML-DSA-44 Verify: 1.16M Insn

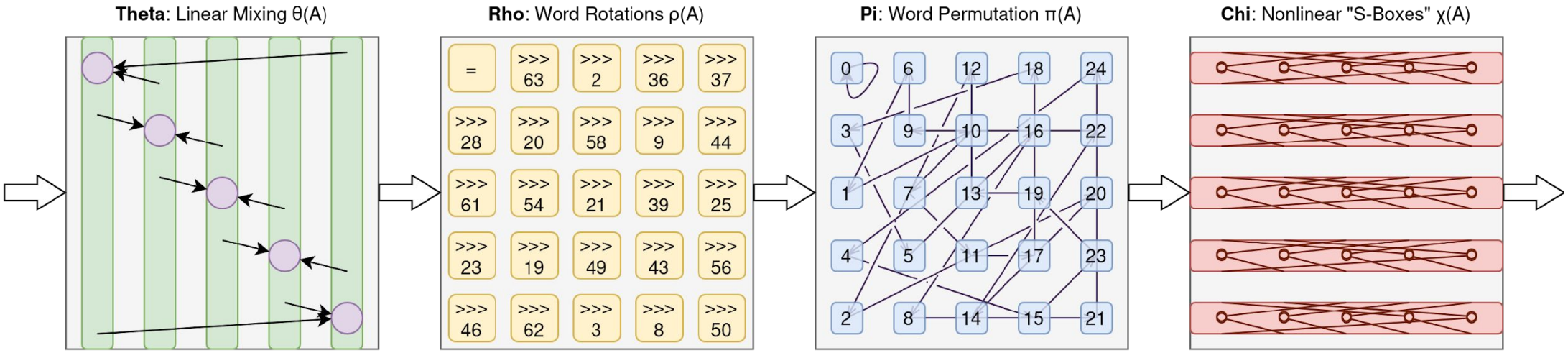
ML-DSA-87 Verify: 3.09M Insn



Instret (with vlen:128,elen:64) - LLVM 18 snapshot, Oct 2023. `-Ofast -march=rv64gcv_zbb (zvk)`

Keccak f1600: The core of SHA3 & SHAKE

- **SHA3** and **SHAKE** (FIPS 202) are built on the 25×64=1600-bit Keccak permutation. >50% of ML-KEM, ML-DSA Cycles, >90% SLH-DSA here.
- **24 Rounds, 1600-bit state**. Relatively slow in software but very fast in hardware (but rather large area.) A lot of XORs and NOT-AND gates.



Main PQC TG Proposal: **A Keccak Instruction**

Keccak state is little awkward to fit into vector architecture:

- Seemingly VLEN ≥ 256 is required (the max LMUL value is 8.)
- Element EEW = 64. Element group EGS = 32, LMUL = 2048 / VLEN:
  - VLEN = 256: LMUL = 8: A group of 8 vector registers of 256 bits.
  - VLEN = 512: LMUL = 4: A group of 4 vector registers of 512 bits.

Multi-round instruction (due to complexity of accessing 25 words):

```
vkeccak.vi vd, vs2, imm    # imm = 5-bit num rounds
```

Computes 24 rounds of Keccak-p[1600,24] permutation with imm=24.

(Ed. note: Sorry about jargon & acronyms, this slide was made for RISC-V Summit!!)

Optimizing Kyber & Dilithium with Vector/SIMD

Application-class RISC-V processors have **vector instructions** available, similarly to AVX SIMD on Intel and NEON, SVA on ARM architectures.

We optimized Kyber and Dilithium with **RISC-V Vector Intrinsics** / CLANG 20.

Benchmarked with SpacemiT X60 (VLEN=256) & C908 (VLEN=128) silicon:

- Vector really helps (~5x speedup) with arithmetic parts (NTT) and somewhat with the bit packing and sampling too.
- Vector does **not** help SHA3/SHAKE much -- that becomes a bottleneck.

Impact: Kyber-768 (ML-KEM) Key Exchange

	KeyGen()	Encaps()	Decaps()	TOTAL	Speedup
RV64GC	663,067	815,357	1,006,469	2,484,893	**1.00**
RV64GCV+ZBB	546,631	685,201	858,508	2,090,340	**1.19**
w. Intrinsics	223,400	239,714	262,241	725,355	**3.43**
w. Keccak Insn	49,363	61,632	84,120	195,115	**12.74**

Clang 20.0.0git -O3 with –march=rv64gc / rv64gcv_zbb_zvl256b

Lines 1-3 uses C language reference Keccak f1600; 4,038 instructions.

Line 4 uses SPIKE 1 cycle Keccak. In real-life in hardware ~100 cycles.

Keccak Instruction: Microarchitecture Notes

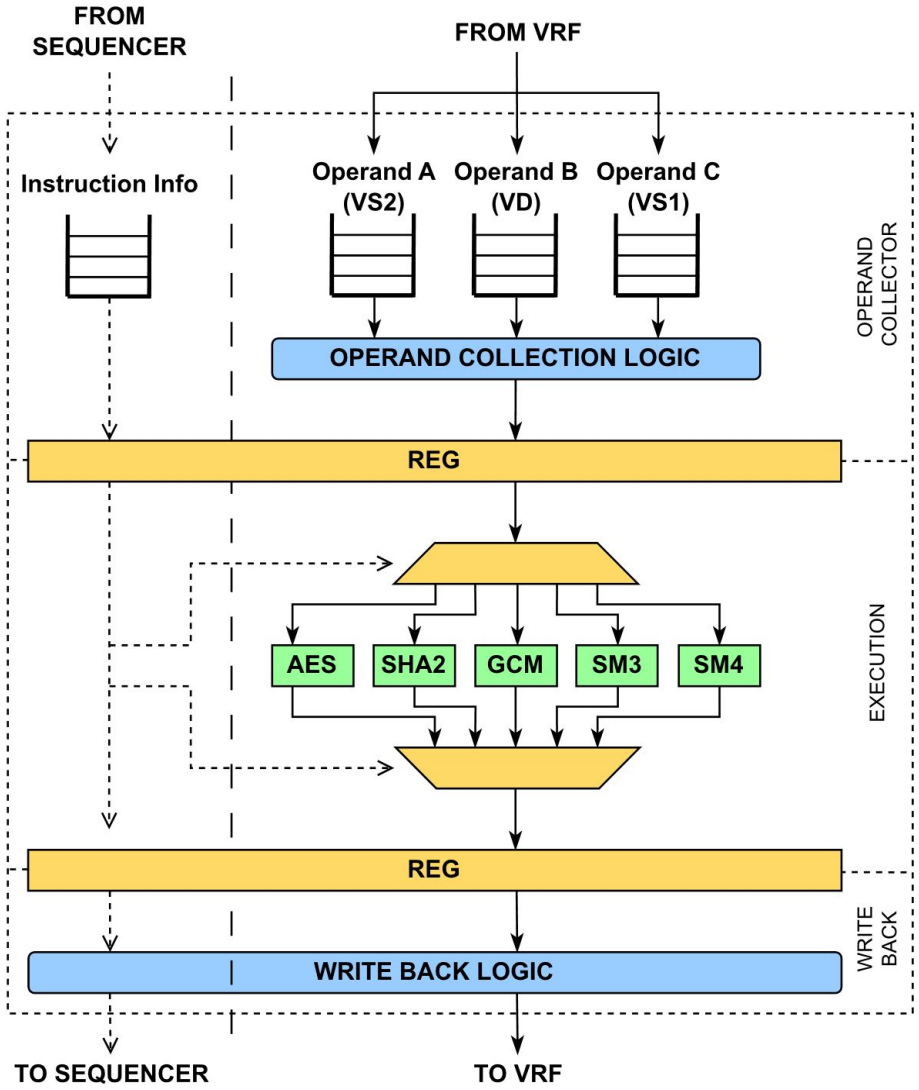
In **"Marian"** we extended the PULP Ara2 vector unit with Zvk crypto instructions.

We used a 256-bit "operand collector" because of the VRF structure. VRF is split into lanes. Each lane only has 64-bit segments of each reg; need to combine.

Keccak would need a 1600-bit collector.

But would probably still be worthwhile!

https://github.com/soc-hub-fi/Marian



PQC Support: RISC-V PQC TG Recommendation

Keccak instruction seems like a winner, giving significant speedups for all PQC algorithms. NTT can also be considered, but RVV already helps a lot with that.

This instruction is replacing thousands of instructions. Core f1600 permutation is 24 cycles. Together with operand collection + writeback can still be under 100.

Hardware note: Permutation alone is about 40kGE + "operand collector" logic.

PQC speedup of Keccak Insn. on RVV ~2-4x. Quite easy to integrate into software.

Microarchitecturally awkward but saves device battery / $$$ in data centre.

The dark side: The “Invisible” Chip Cryptography

- Vendors (Intel, AMD, Apple, Google, NVIDIA, Qualcomm, Google, … but also RISC-V system makers) need to be **able to update their system chips**.

- This creates ***incredible supply chain risks*** – think of a rootkit or malware in a microcode update to an Intel CPU. This is completely invisible to OS & users.

- Ecosystems (Android, Windows, Apple OS) and device vendors want to protect their devices against “jailbreaking” and unauthorized modification.

- **Consequence**: Much more serious measures are taken to protect the chips and devices *themselves* than any user application running on them. 🤦

Evaluating Physical Attacks against **PQC Modules**

FIPS 140-3 for PQC

- FIPS 140-3 is required by U.S. Federal government and many industrial standards.
- Currently focuses only on functional (test vector) and “checklist compliance” testing.
- Random numbers: SP 800-90 still good for PQC.
- Slowly coming: “non-invasive” (ISO 17825) leakage assessment for FIPS 140-3 level 3+.

Common Criteria and AVA_VAN

- High assurance level (EUCC: AVA_VAN.3+) is required for **Root of Trust IP**, **Smart Cards**, **Secure elements**, many types of **IoT** (SESIP).

CC AVA_VAN and “Attack Potential”

- AVA_VAN checks more of real-life security via a “penetration test.” Can be very demanding.
- AVA_VAN security level is determined by "attack potential": A score-based system that measures cost of attack.
- Specialized 3rd party testing laboratories.
- *“Evaluators must have knowledge and experience of [..] side channel attacks (SCA) such as Timing Analysis, Machine Learning based SCA, Simple Power Analysis (SPA), Differential Power Analysis (DPA), Differential EM radiation Analysis (DEMA), Template Attacks (TA); fault injection attacks such as DFA [..]” **-- EUCC documents***

Protection Profiles for Chip Security

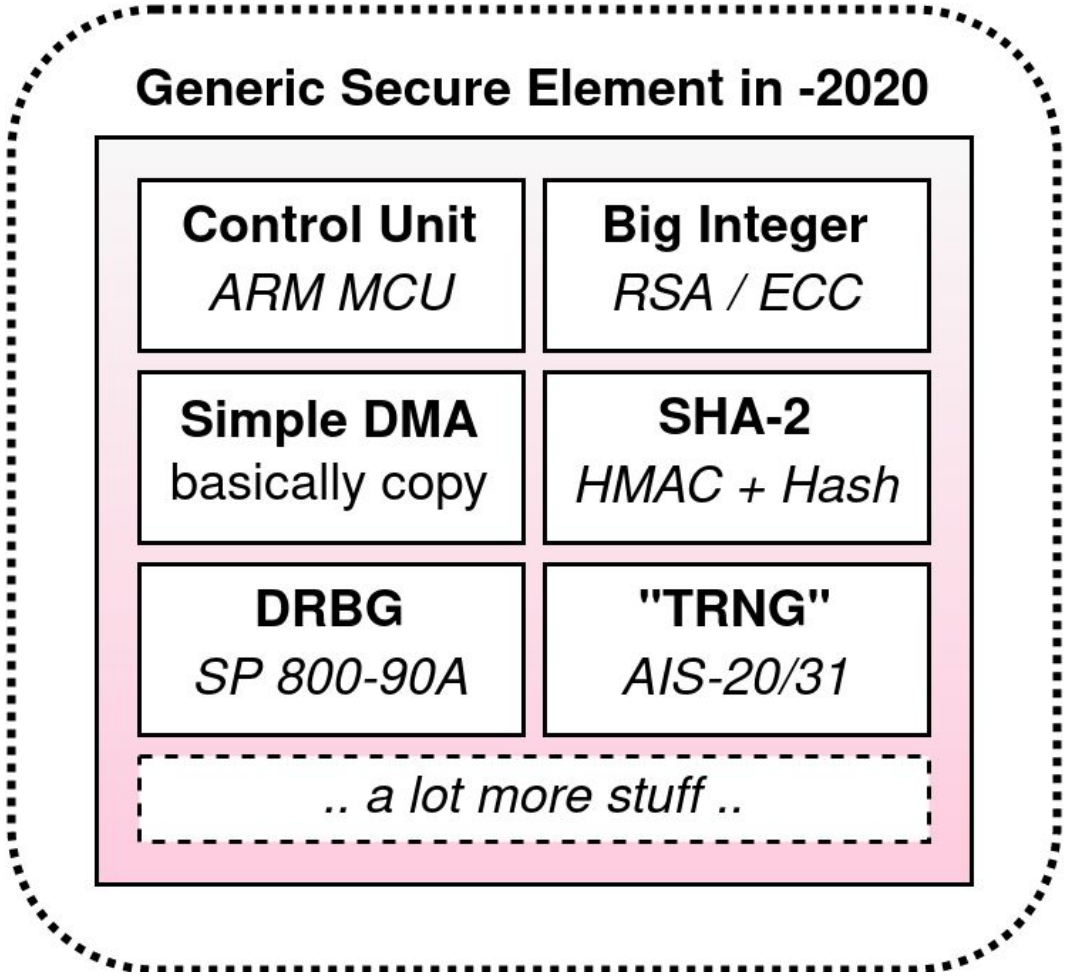
AVA_VAN.3+ is a common requirement for Root of Trust and Security IC products.
We assume that this will not change (much) with Post-Quantum Cryptography.

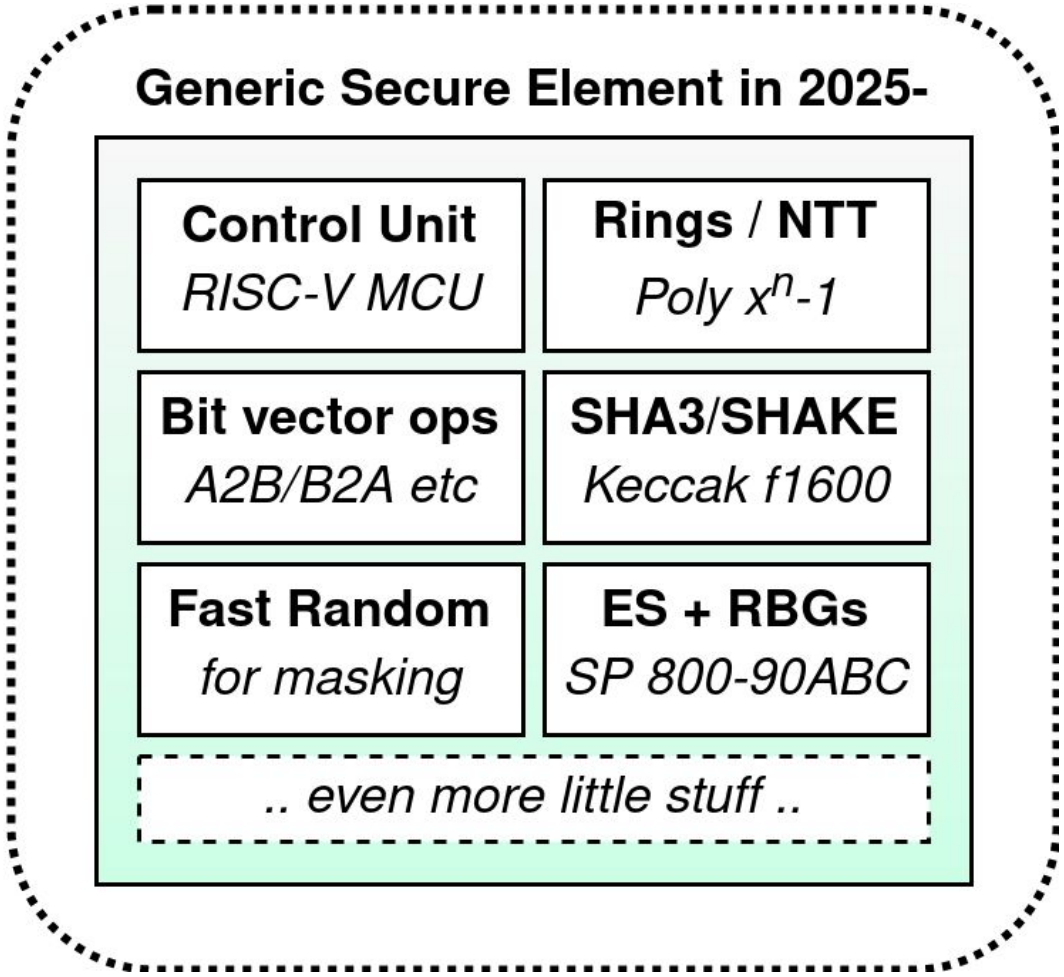
[JSADEN011] **“SESIP Profile for PSA Certified™ Level 3”**
Root of Trust (PSA-RoT): 35 person-days of AVA_VAN.3 activities.

[PP-0084] **“Security IC Platform Protection Profile”**
EAL 4 augmented by AVA_VAN.5 and ALC_DVS.2

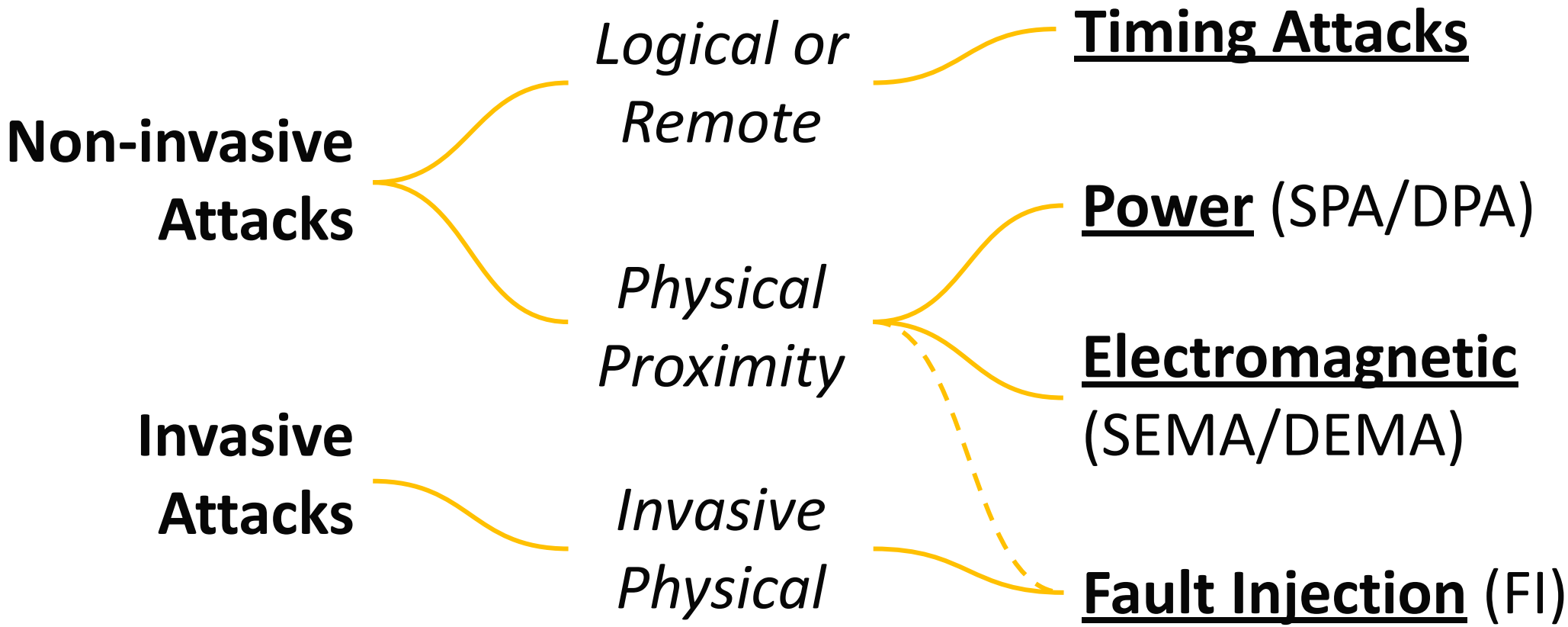
[PP-0117] **“Secure Sub-System in System-on-Chip (3S in SoC)”**
EAL 4 augmented by ATE_DPT.2, AVA_VAN.5, ALC_DVS.2 and ALC_FLR.2

On Changes in RoT Hardware Architecture





Side-Channel and Fault Attacks

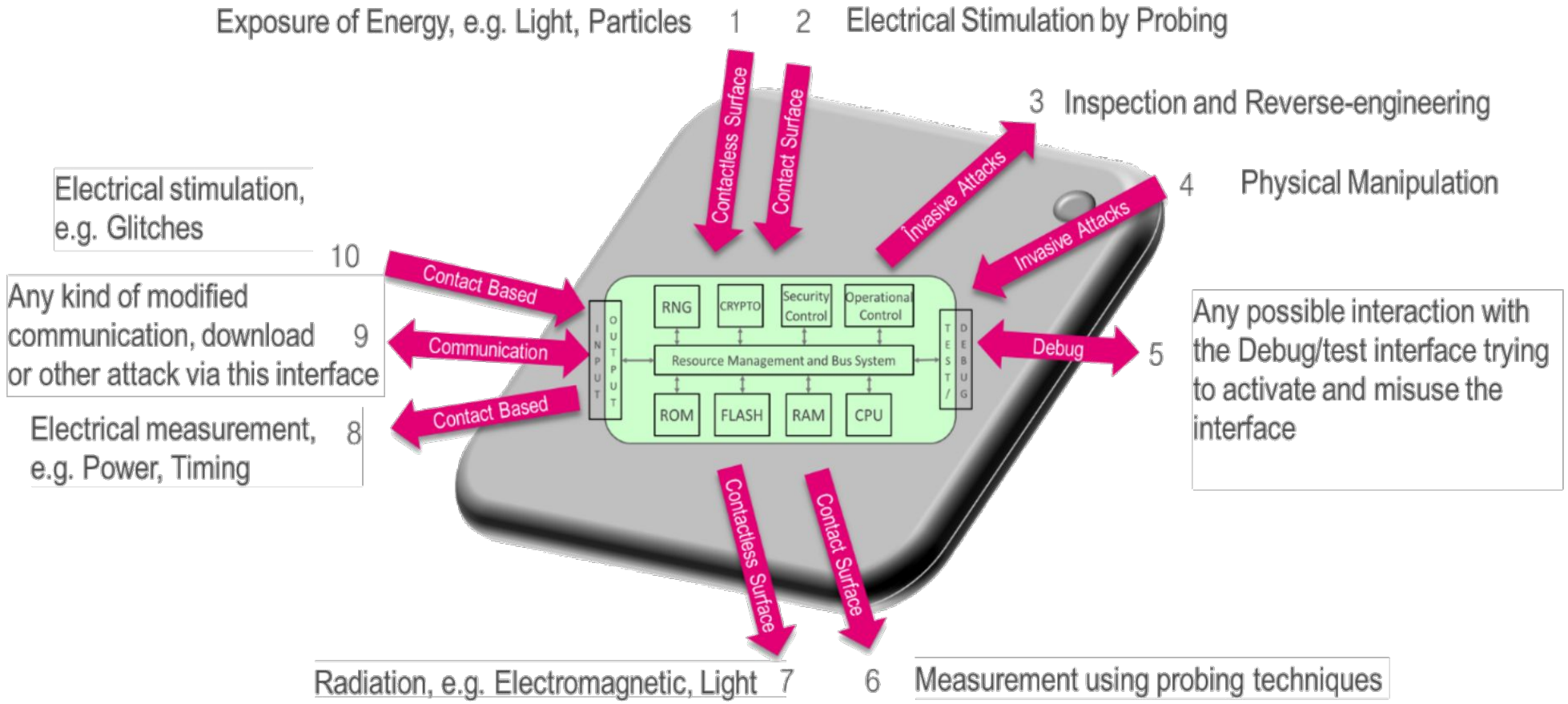


(From PP-0117)

3.2 Threats

The threats described in this section are applicable to the base Protection Profile. For threats related to functional extensions see Chapter 7.

The following figure describes the attacks that are applicable to the TOE. The interactions related to the attacks are marked with red arrows.



https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Reporte/ReportePP/pp0117V2b_pdf.pdf

Some Classical countermeasures – RSA and ECC

These were extremely simple algorithms + had algebraic structure

RSA: Blinding and masking (D. Chaum 1982, P. Kocher 1996)

- Message blinding: Pick random **r**, compute blinded $\mathbf{c'=cr^e}$ (mod n), decrypt/sign **c'** instead of **c**: $\mathbf{m'=c'^d}$ (mod n), normalize by $\mathbf{m = m'r^{-1}}$.
- Exponent masking: use **d'** = **(p-1)(q-1)r + d** to randomize exponentiation.

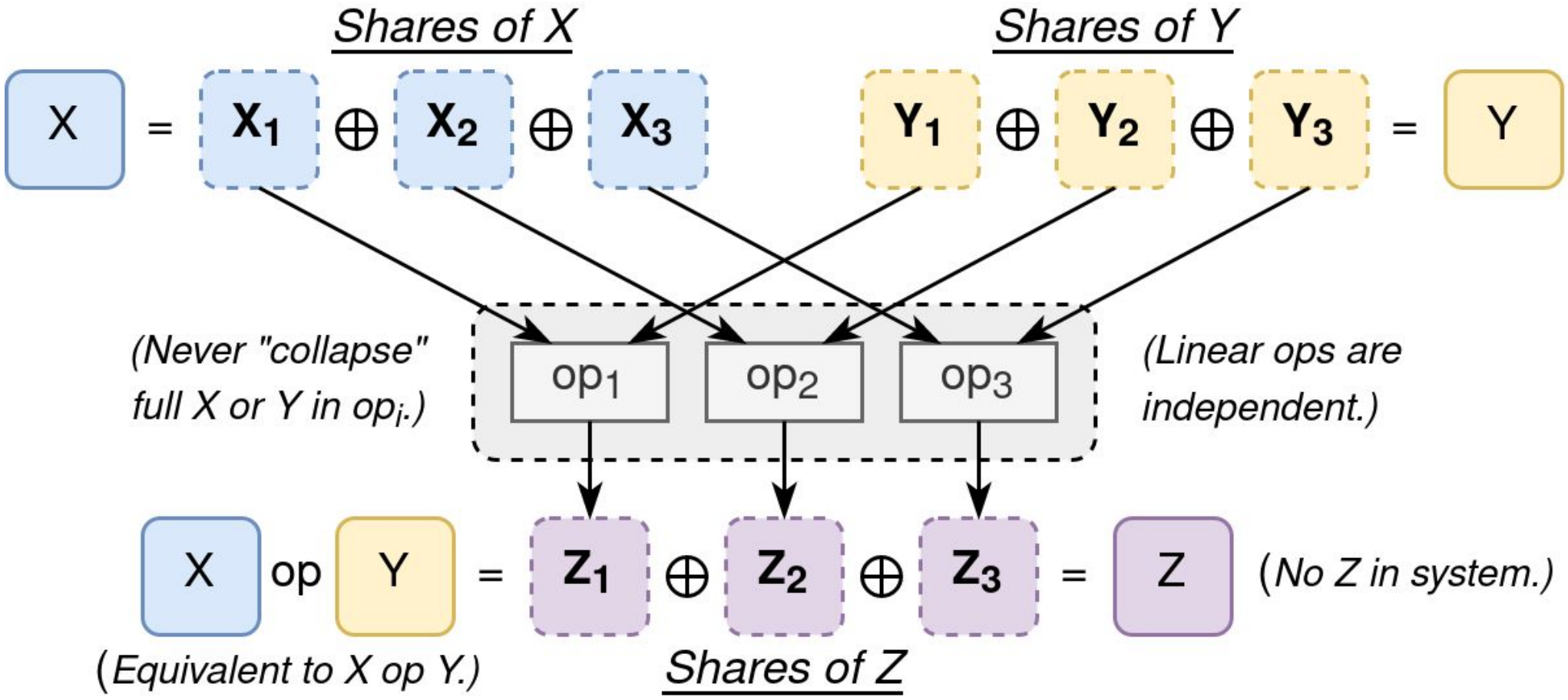
ECC: J.-S. Coron, *"Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems."* Proc. CHES'99, pp. 292–302, 1999

- For 20+ years: Randomization of the Private Exponent [Scalar], Blinding the [Base] Point P, Randomized Projective Coordinates, + misc.

Basically 1 step – ModExp (RSA) or Scalar Mult (ECC) – to protect

Masking and Threshold Implementations

Limit leakage by breaking computation into randomized shares



Masking Countermeasures

PQC needs masking (+ blinding, shuffling, random delays ..)

- Masking splits secrets into “shares.” Successful measurement of an individual share does not leak secret info. Design “Masking Gadgets” to perform arithmetic steps.

Type	Relationship	Algebraic Object
Algebraic / Prime Field	$X = X0 + X1 \pmod{q}$	Mod 3329 (Kyber) or 8380417 (Dilithium)
Algebraic / Power-of-2	$X = X0 + X1 \pmod{2^n}$	Some Lattice Crypto, SHA2, etc
Boolean / Binary Field	$X = X0 \oplus X1$	Nonlinear Functions, shifts, symmetric Crypto

- **Most cryptographers agree**: Masking and other attack mitigation techniques for PQC algorithms are much more complex than countermeasures for older cryptography.

- **Why?** The algorithms are not homogenous like RSA or ECC but contain a number of dissimilar steps. One may have to design dozen different gadgets for one algorithm.

AVA_VAN: Common Criteria Vulnerability Analysis

Attack Potential is evaluated with a score-based system that roughly maps to the "**cost of attack**." (think $ or €)

Considers attack **Identification** + **exploitation**, with many factors:

- Elapsed time (hours–months)
- Attacker Expertise (multiple)
- Knowledge (how restricted)
- Access to the TOE (samples)
- Equipment (common/bespoke)

("Application of Attack Potential" docs.)

AVA_VAN.1 Vulnerability Survey
- TOE resistance against BASIC Attack Potential (0-15)

AVA_VAN.2 (Unstructured) Vuln. Analysis
- TOE resistance against BASIC Attack Potential (16-20)

AVA_VAN.3 Focused (Unstructured) Vuln. Analysis
- TOE resistance against ENHANCED-BASIC AP (21-24)

AVA_VAN.4 Methodical Vuln. Analysis
- TOE resistance against MODERATE AP (25-30)

AVA_VAN.5 Advanced Methodical Vuln. Analysis
- TOE resistance against HIGH Attack Potential (31-)

Attack Potential: Example Calculation

AP Component	Identification	Exploitation
Elapsed time	2 (< one week)	6 (< one month)
Expertise	5 (expert)	4 (expert)
Knowledge of the TOE	4 (sensitive)	0 (public)
Access to the TOE	0 (< 10 samples)	0 (< 10 samples)
Equipment	3 (specialized)	4 (specialized)
Open Samples	0 (public)	0 (public)
Total	28 (AVA_VAN.4 / moderate AP range)	

SOG-IS: “Application of Attack Potential to Smartcards and Similar Devices”

https://www.sogis.eu/documents/cc/domains/sc/JIL-Application-of-Attack-Potential-to-Smartcards-v3.2.1.pdf

Summary on PQC Hardware Impact

- **One can already get FIPS 140-3 certificates for PQC modules (HSMs.)**

- **Current ISAs do a reasonably good job:** PQC algorithms are quite fast.

  RISC-V PQC TG is currently evaluating a vector instruction that implements “all-rounds” Keccak (SHA-3). Secondary consideration: NTT arithmetic.

- **Side-channel and fault attack countermeasures are harder: Delays?**

  PQC Secure elements and Root-of-Trust units may “inherit” the AVA_VAN.5 requirements in relevant CC Protection Profiles. These are difficult to satisfy.