NTNU | Norwegian University of Science and Technology

SIDE-CHANNEL ATTACKS 4: POST-QUANTUM CRYPTO

TTM4205 – Lecture 10

Tjerand Silde

11.10.2024

Contents

Contents

Reminder

This is the last week of lab on Tuesdays. The remaining ones will be lectures.

Exercises sessions will continue as before on Fridays with B2 and then A176.

You should start thinking about groups and topics for the technical essay.

Contents

Black Box Crypto

We design the security of a cryptographic scheme to follow Kerckhoff's principle: if everything about the scheme, except for the key, is known, then the scheme should be secure.

We analyze the scheme mathematically as black-box algorithms that take some (public or secret) input and give some (public or secret) output, and prove it secure concerning the algorithm description and the public data.

However, security depends on your model. In practice, it matters how these algorithms are implemented and what kind of information the *physical* system leaks about the inner workings of the algorithm computing on secret data.

Leakage

- The time it takes to compute...
- The power usage while computing...
- The electromagnetic radiation...
- The temperature variation...
- The memory pattern accessed...
- The sounds your laptop makes...

Exploiting Leakage

- Timing or power traces can leak secret bits
- Fault injection might leak dummy operations
- Differential analysis allow statistical attacks
- The adversary can choose the input (adaptively)
- The secret key might be static and re-used

Attack Categories

- Remote vs physical attacks
- Software and hardware attacks
- Passive vs active attacks
- Invasive vs non-invasive attacks

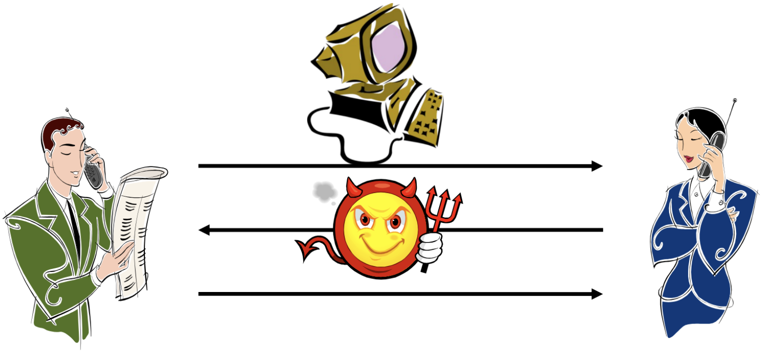
Preventing Leakage

- Constant time operations and algorithms
- The result must depend on all operations
- Randomize input and/or secrets each time
- Split secrets into random additive shares

Contents

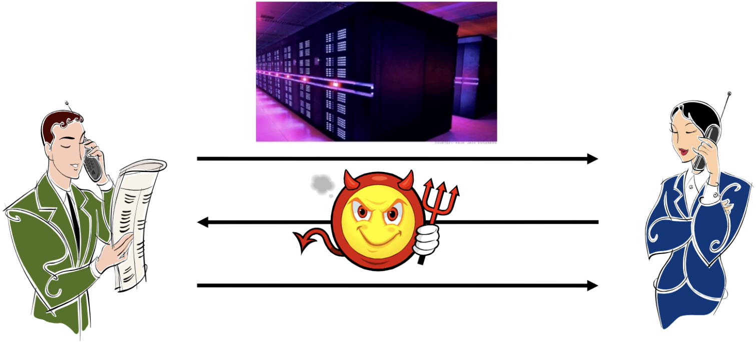
Cryptography Today

Allows for secure communication in the presence of malicious parties



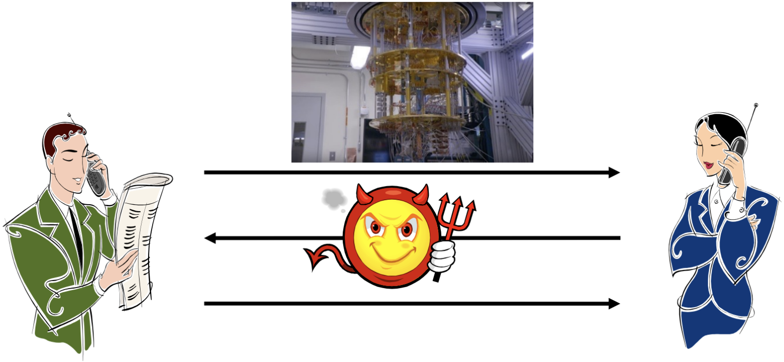
Cryptography Today

Large increase in the adversary's computing power requires only a small increase in the key size



Cryptography Tomorrow

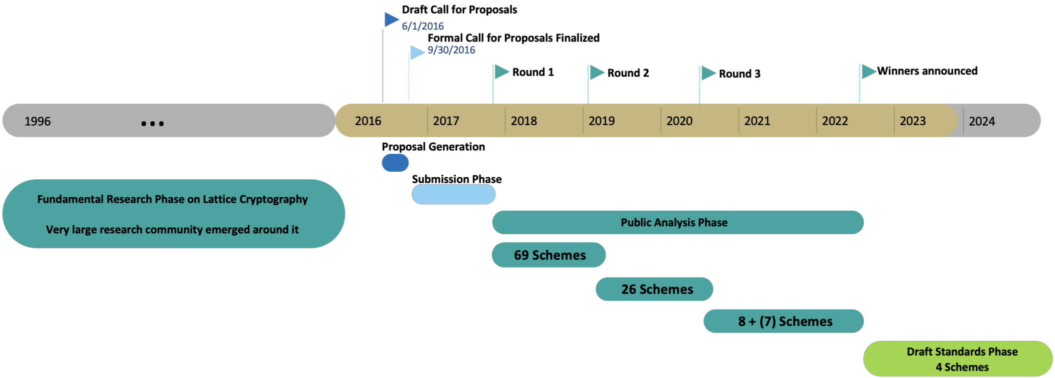
A quantum computer is outside the classical model of computation for efficiency purposes



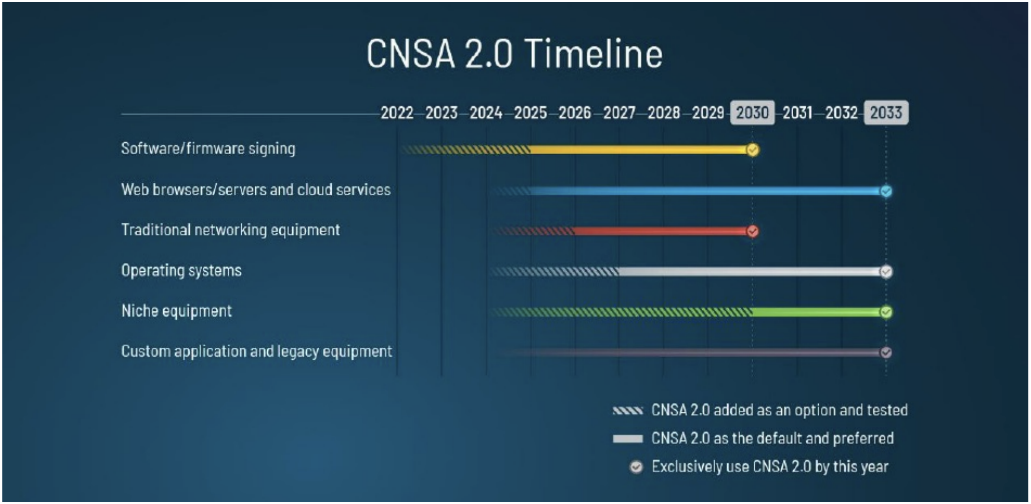
Cryptography Tomorrow

Shor's quantum algorithm can factorize integers and compute discrete logs essentially as fast as using them, given a large quantum computer. This would break the RSA, DH, DSA schemes and others built on these assumptions. To achieve future secrecy, there is an urgent need to replace those algorithms.

NIST Timeline



NSA Timeline



Crypto Categories

No Changes Necessary

Symmetric Cryptography:
- AES
- SHA-256 / SHA-3
- HMAC
- etc.

Done.

Almost Drop-in Replacements

NIST standardizations:
- Public Key Encryption
- Key Exchange
- Digital Signatures

A few other things:
- Identity-Based Encryption

Almost standards. Ready for deployment.

Serious Alterations of Protocols Required

Advanced Primitives:
- Zero-Knowledge Proofs
- Distributed Privacy
- Many blockchain privacy applications

Lots of recent progress on design. Near-optimality has just been achieved for certain primitives. Implementation starting at ZRL.

Can Only Be Done with Lattice Cryptography

- Fully-Homomorphic Encryption (FHE) - computation over encrypted data
- Some Obfuscation (still unclear if it can be efficient or have any useful applications)

Implementation / deployment of FHE at Haifa.

Contents

Learning With Errors (LWE)

Definition 1. For positive integers m, n, q, and $\beta < q$, the $\mathsf{LWE}_{n,m,q,\beta}$ problem asks to distinguish between the following two distributions:

1. $(\mathbf{A}, \mathbf{A}\mathbf{s} + \mathbf{e})$, where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow [\beta]^m, \mathbf{e} \leftarrow [\beta]^n$
2. $(\mathbf{A}, \mathbf{u})$, where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{u} \leftarrow \mathbb{Z}_q^n$.

Short Integer Solution (SIS)

Definition 4. For positive integers m, n, q, and $\beta < q$, the $\mathsf{SIS}_{n,m,q,\beta}$ problem asks to find, for a randomly-chosen matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$, vectors $\mathbf{s}_1 \in [\beta]^m$ and $\mathbf{s}_2 \in [\beta]^n$ such that $\mathbf{A}\mathbf{s}_1 + \mathbf{s}_2 = \mathbf{0} \pmod q$.

Hardness of LWE and SIS

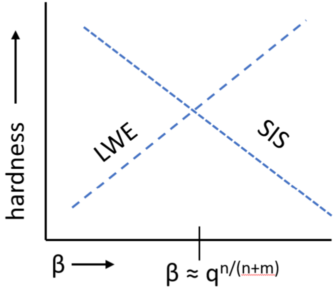


Figure 2: The hardness of $\mathsf{LWE}_{n,m,q,\beta}$ and $\mathsf{SIS}_{n,m,q,\beta}$ for fixed n, m, q, and varying β. The lines are not meant to describe the concrete hardness of these problems, but rather to illustrate the dependence of the hardness of these problems on β. The intersection point is approximately at $\beta = q^{n/(n+m)}$.

Parameters for LWE and SIS

Table 1: Approximate values of δ-hardness of the $\mathsf{LWE}_{m,q,\beta}$ problem for some parameters that resemble those used in the Kyber encryption (ML-KEM) scheme

$\mathsf{LWE}_{m,q,\beta}$ Parameters			
m	β	q	δ
512	2	2^{12}	1.0043
768	2	2^{12}	1.0029
1024	2	2^{12}	1.0022

Table 2: Approximate values of δ-hardness of the $\mathsf{LWE}_{m,q,\beta}$ and $\mathsf{SIS}_{n,q,\beta}$ problems for some parameters that resemble those used in the Dilithium (ML-DSA) signature scheme.

$\mathsf{LWE}_{m,q,\beta}$ Parameters			
m	β	q	δ
1024	2	2^{23}	1.004
1280	4	2^{23}	1.003
1792	2	2^{23}	1.0023

$\mathsf{SIS}_{n,q,\beta}$ Parameters			
n	β	q	δ
1024	2^{18}	2^{23}	1.0041
1536	2^{20}	2^{23}	1.0032
2048	2^{20}	2^{23}	1.0025

Survey

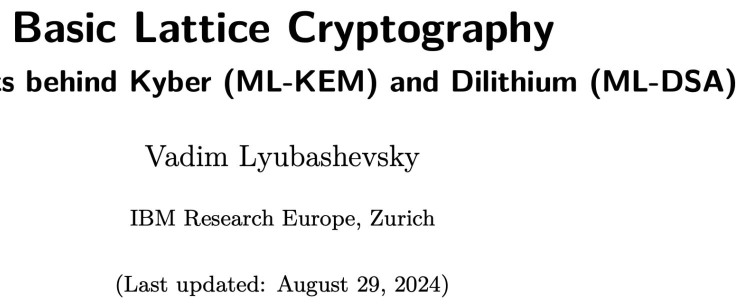


Figure: https://eprint.iacr.org/2024/1287.pdf

Contents

KGen and Enc

$$\mathsf{sk}: \mathbf{s} \leftarrow [\beta]^m, \ \mathsf{pk}: (\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times m}, \mathbf{t} = \mathbf{A}\mathbf{s} + \mathbf{e}_1), \text{ where } \mathbf{e}_1 \leftarrow [\beta]^m. \tag{6}$$

To encrypt a message $\mu \in \{0, 1\}$, the encryptor chooses $\mathbf{r}, \mathbf{e}_2 \leftarrow [\beta]^m$ and $e_3 \leftarrow [\beta]$, and outputs

$$\left(\mathbf{u}^T = \mathbf{r}^T\mathbf{A} + \mathbf{e}_2^T, v = \mathbf{r}^T\mathbf{t} + e_3 + \left\lceil \frac{q}{2} \right\rfloor \mu\right). \tag{7}$$

Figure: Q: Which operations might leak information?

Dec

To decrypt, one computes $v - \mathbf{u}^{\hat{T}}\mathbf{s}$. But rather than this cleanly giving us the message μ as in (4), we instead obtain

$$v - \mathbf{u}^T\mathbf{s} = \mathbf{r}^T(\mathbf{A}\mathbf{s} + \mathbf{e}_1) + e_3 + \frac{q}{2}\mu - \left(\mathbf{r}^T\mathbf{A} + \mathbf{e}_2^T\right)\mathbf{s} \tag{8}$$

$$= \mathbf{r}^T\mathbf{e}_1 + e_3 + \frac{q}{2}\mu - \mathbf{e}_2^T\mathbf{s} \tag{9}$$

Size

Kyber-768

Sizes (in bytes)		Haswell cycles (ref)		Haswell cycles (avx2)	
sk:	2400	gen:	199408	gen:	52732
pk:	1184	enc:	235260	enc:	67624
ct:	1088	dec:	274900	dec:	53156

Contents

Scheme

Private information: $\mathbf{s}_1 \in [\beta]^m, \mathbf{s}_2 \in [\beta]^n$
Public information: $\mathbf{A} \in \mathcal{R}_{q,f}^{n\times m}, \mathbf{t} = \mathbf{A}\mathbf{s}_1 + \mathbf{s}_2 \in \mathcal{R}_{q,f}^{n}$

Prover		Verifier
$\mathbf{y}_1 \leftarrow [\gamma + \bar{\beta}]^m$ $\mathbf{y}_2 \leftarrow [\gamma + \bar{\beta}]^n,$ $\mathbf{w} := \mathbf{A}\mathbf{y}_1 + \mathbf{y}_2$		
	$\xrightarrow{\mathbf{w}}$	$c \leftarrow \mathcal{C}$
	$\xleftarrow{c}$	
$\mathbf{z}_1 := c\mathbf{s}_1 + \mathbf{y}_1$ $\mathbf{z}_2 := c\mathbf{s}_2 + \mathbf{y}_2$ if $\mathbf{z}_1 \notin [\bar{\beta}]^m$ or $\mathbf{z}_2 \notin [\bar{\beta}]^n$ then $(\mathbf{z}_1, \mathbf{z}_2) := \perp$		
	$\xrightarrow{(\mathbf{z}_1, \mathbf{z}_2)}$	Accept iff $\mathbf{z}_1 \in [\bar{\beta}]^m$ and $\mathbf{z}_2 \in [\bar{\beta}]^n$ and $\mathbf{A}\mathbf{z}_1 + \mathbf{z}_2 - c\mathbf{t} = \mathbf{w}$

Figure: Q: Which operations might leak information?

Size

Dilithium3

Sizes (in bytes)		Skylake cycles (ref)		Skylake cycles (avx2)	
sk:		gen:	544232	gen:	256403
pk:	1952	sign:	2348703	sign:	529106
sig:	3293	verify:	522267	verify:	179424

Contents

Protection Techniques

- constant time sampling of secrets
- avoid the rejection sampling step
- masking multiplication with secrets

Trade-offs

Signature schemes strike a balance between:

- Sizes (verification key and signatures)
- Speed (signing, verification)
- Portability
- Conservative assumptions
- **Resistance against side-channel attacks**

And so on...

Criteria					
Dilithium	★★½	★★★	★★★	★★	😭
Falcon	★★★	★★★	★★	★★	😭
SPHINCS+	★½	★★	★★	★★★	😭
Raccoon	★★	★★★	★★★	★★	★★★

t-Probing Model

t-probing model

- Adversary can probe t circuit values at runtime
- Unrealistic but a good starting point

Masking

- Each sensitive value x is split in d shares:

$$[\![x]\!] = (x_0, x_1, \ldots, x_{d-1}) \tag{1}$$

such that

$$x_0 + x_1 + \cdots + x_{d-1} = x \tag{2}$$

- In t-probing model, ideally 0 leakage if $d > t$
- In "real life", security is exponential in d
- What about computations?

Difficulty of Masking

How difficult are operations to mask?

- 😀 **Addition ($[\![c]\!] = [\![a+b]\!]$)?**
  - Compute $[\![c]\!] = (a_0 + b_0, \ldots, a_{d-1} + b_{d-1})$, simple and fast: $\Theta(d)$ operations
- ☹️ **Multiplication ($[\![c]\!] = [\![a \cdot b]\!]$)?**
  - Complex and slower: $\Theta(d^2)$ operations
- 😱 **More complex operations?**
  - Use so-called *mask conversions*, very slow: $\gg \Theta(d^2)$ operations

Masking Dilithium

Dilithium follows the Fiat-Shamir **with aborts** paradigm.

Sign(sk = s, vk = (A, t), msg) → sig

1. Generate a short ephemeral secret $\mathbf{r}$ ▷ Slow
2. Compute the commitment $\mathbf{w} = \mathbf{A} \cdot \mathbf{r}$ ▷ Fast
3. Compute the challenge $c = \mathsf{H}(\mathbf{w}, \mathsf{msg}, \mathsf{vk})$ ▷ No mask
4. Compute the response $\mathbf{z} = \mathbf{s} \cdot c + \mathbf{r}$ ▷ Fast
5. Check that $\mathbf{z}$ is in a given interval. If not, restart. ▷ Slow
6. Signature is $\mathsf{sig} = (c, \mathbf{z})$

Masking bottlenecks:

- 😱 Short secret generation (1) requires B2A.
- 😱 Rejection sampling (5) requires A2B and B2A.

Total masking overhead: $\Theta(d^2 \log q)$

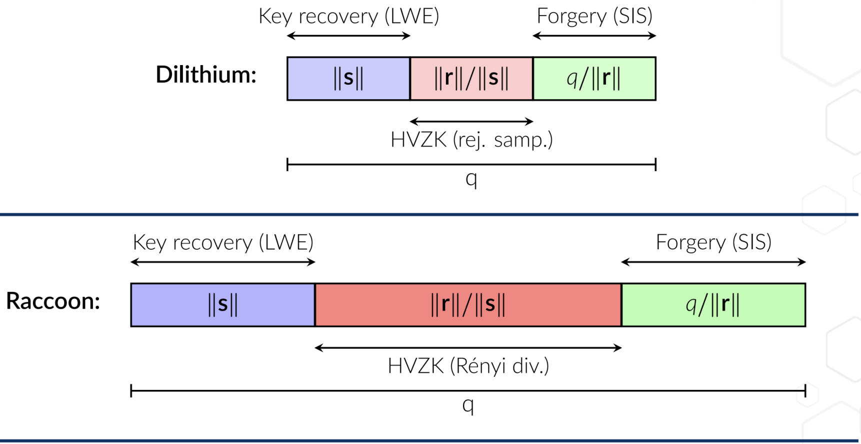
Masking Raccoon

Sign(sk = $[\![\mathbf{s}]\!]$, vk = $(\mathbf{A}, \mathbf{t})$, msg) → sig

1. Generate a masked short ephemeral secret $[\![\mathbf{r}]\!]$ using "AddRepNoise" ▷ Fast
2. Compute the commitment $[\![\mathbf{w}]\!] = \mathbf{A} \cdot [\![\mathbf{r}]\!]$ ▷ Fast
3. Unmask $[\![\mathbf{w}]\!]$ to obtain $\mathbf{w}$ ▷ Fast
4. Compute the challenge $c = \mathsf{H}(\mathbf{w}, \mathsf{msg}, \mathsf{vk})$ ▷ No mask
5. Compute the response $[\![\mathbf{z}]\!] = [\![\mathbf{s}]\!] \cdot c + [\![\mathbf{r}]\!]$ ▷ Fast
6. Unmask $[\![\mathbf{z}]\!]$ to obtain $\mathbf{z}$ ▷ Fast
7. (No more rejection sampling!)
8. Signature is $\mathsf{sig} = (c, \mathbf{z})$

Total masking overhead: $O(d \log d)$

Impact on Modulus



1. Removing rejection sampling increases $\|\mathbf{r}\|/\|\mathbf{s}\|$ from $\Theta(\dim \mathbf{s})$ to $\Theta\left(\|c\|\sqrt{\text{Queries}}\right)$
2. The increased q in turn requires increasing $\|\mathbf{s}\|$, $q/\|\mathbf{r}\|$ and/or the dimensions.

Comparison

Raccoon is a specific-purpose scheme aimed at high side-channel resistance:

- 🙂 Same assumptions as Dilithium
- 🙂 Simpler
- 🙂 Verification key size is similar
- ☹️ Signature is 4x larger
- 🤩 **When masked, orders of magnitude faster than other schemes are**

Comparison

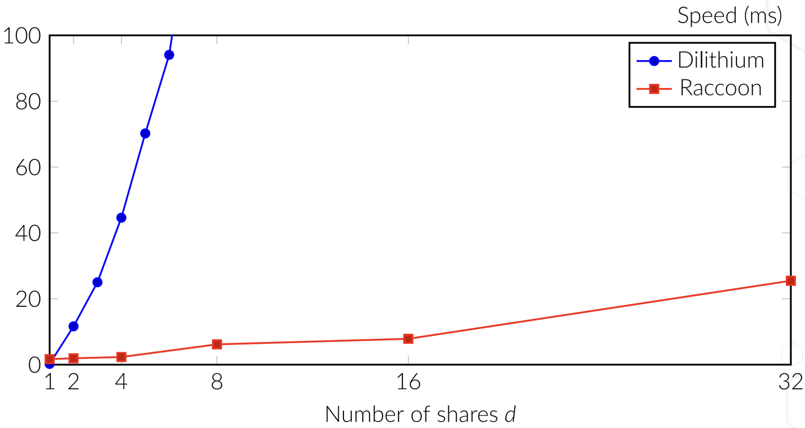


Figure: `https://raccoonfamily.org`

Questions?