Examples of PQC Schemes: Falcon Signature Scheme

Pierre-Alain Fouque
Centre Inria de l’Université de Rennes

Some of these slides were made by Alice Pellet—Mary or Thomas Prest

Signature scheme

Algorithms:

- KeyGen (n) ⇒ (pk, sk)
- Sign(m,sk) ⇒ s, Verify(m,s,pk) ⇒yes/no

Alice		Bob
KeyGen (n) ⇒ (pk, sk)	pk →	Store pk
Message m; Sign(m,sk) ⇒ s	m,s →	Verify(m,s,pk) ⇒yes/no

- Correctness: Verify(m,s,pk) ⇒yes (if Sign(m,sk) ⇒ s and KeyGen(n) ⇒(pk,sk))
- Security: an attacker not knowing sk cannot forge valid pairs (m,s)

Falcon signature scheme (FNDSA)

- Falcon signature scheme:
  - Hash-and-sign signature
  - hard lattice from NTRU assumption with polynomials of degree d=512 / 1024

- Falcon is one of the three post-quantum signature schemes selected to be standardized by NIST in 2022

- Advantage: Very compact |pk|+|sig| the shorter among lattice schemes
- Drawback: Signature process requires floating-point

[Falcon] Fouque, Hoffstein, Kirchner, Lyubashevsky, Pornin, Prest, Ricosset, Seiler, Whyte, Zhang. NIST

Falcon's performances

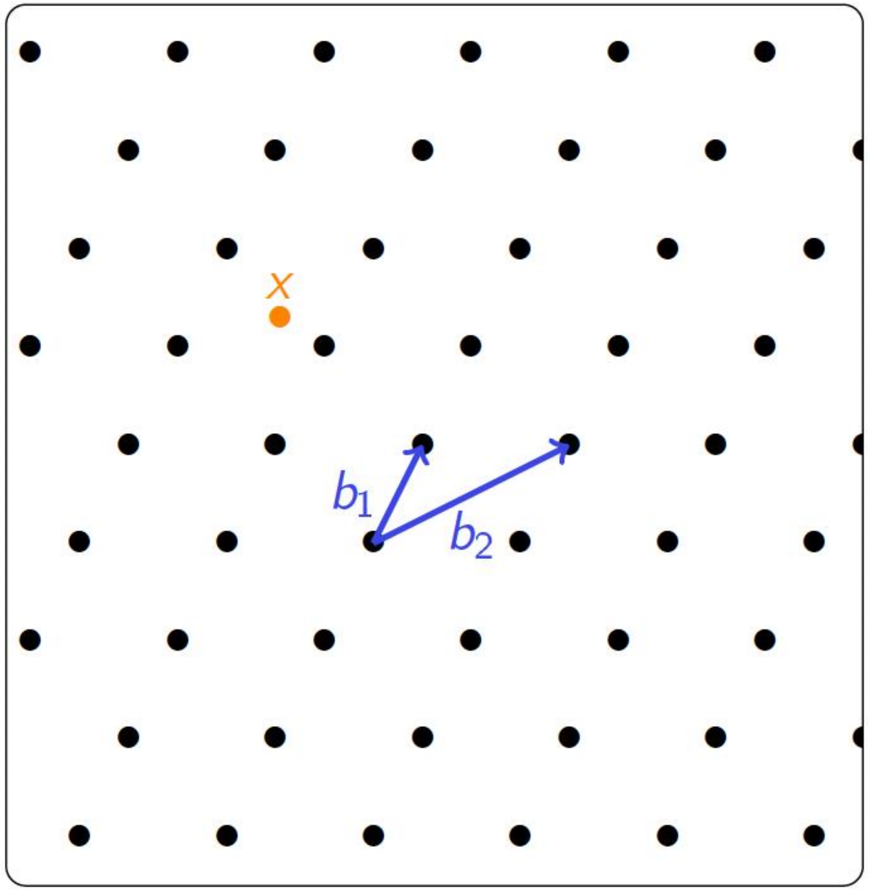
	RSA	EdDSA	Post-quantum standards (NIST 2022) Falcon	Dilithium	SPHINCS+
Public key size (bytes)	256	32	**897**	1312	32
Signature size (bytes)	256	64	**666**	2420	17088
Signature time (μs)	665	51	**241**	208	35584
Verification time (μs)	19	142	**52**	74	2091

Main drawback of post-quantum crypto is the size of the signature and public-key (1 KB)

Lattice-based Hash-and-Sign signature

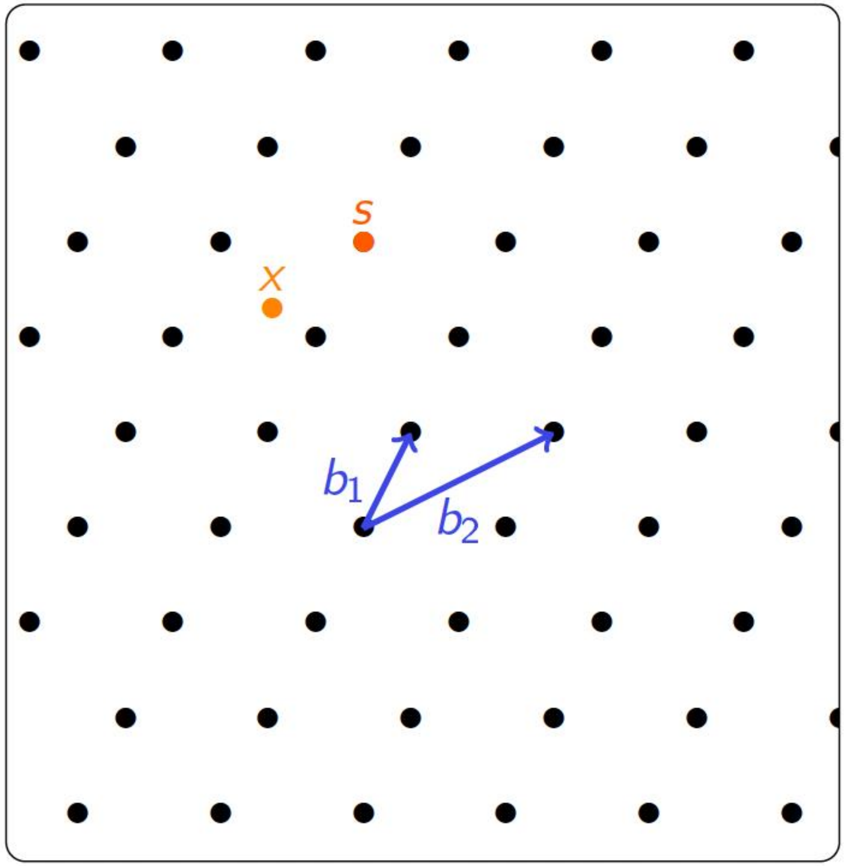
From GGH to GPV

Finding a close vector using a short basis



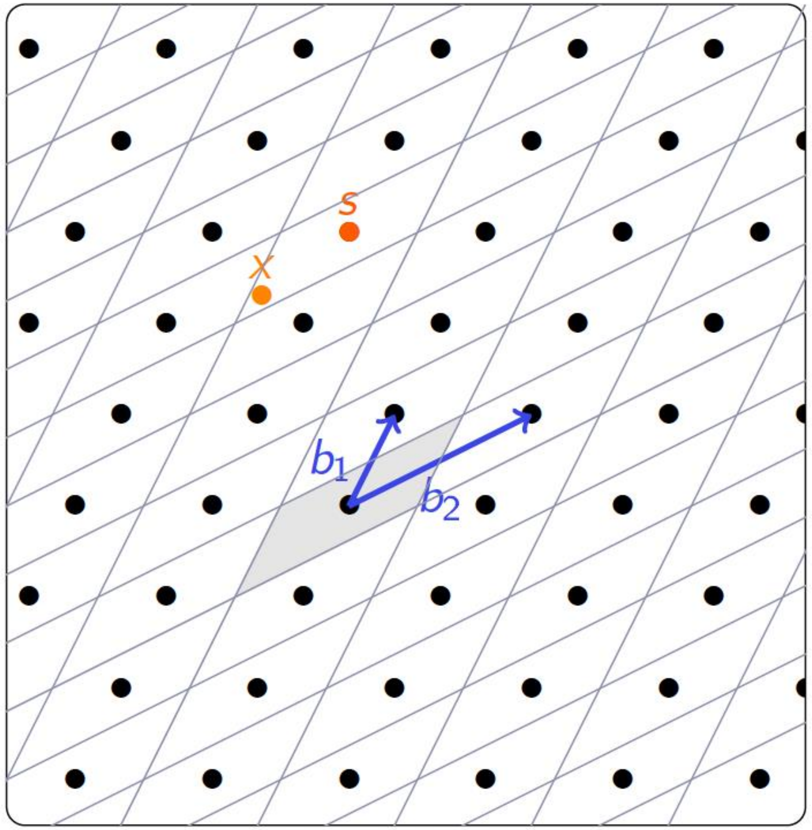
- Input: $x = 3.7b_1 - 1.4b_2$
- Goal: find $s \in L$ close to x
- Algo: round each coordinate

Finding a close vector using a short basis



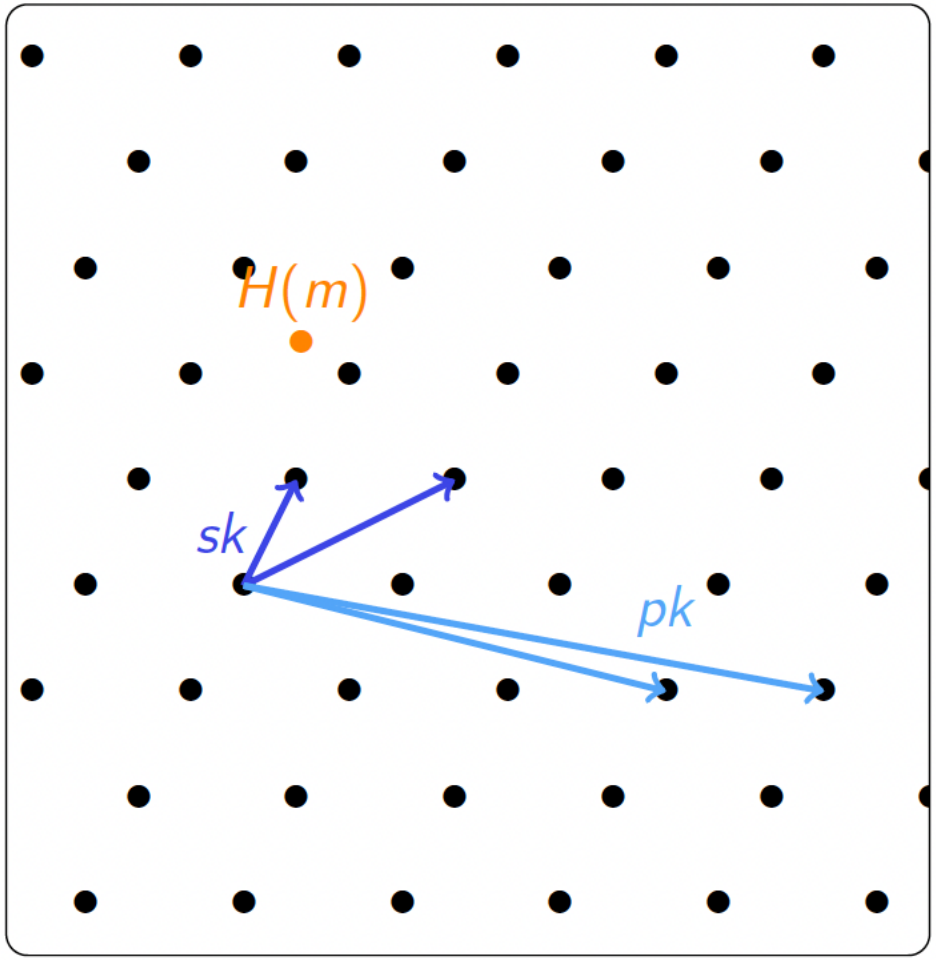
- Input: $x=3.7b_1-1.4b_2$
- Goal: find s∈L close to x
- Algo: round each coordinate
- Output: $s = 4b_1-1b_2$
- The smaller the basis, the closer the solution
- Babai's round-off algorithm

Finding a close vector using a short basis



- Input: $x = 3.7b_1 - 1.4b_2$
- Goal: find $s \in L$ close to x
- Algo: round each coordinate
- Output: $s = 4b_1 - 1b_2$
- The smaller the basis, the closer the solution
- Babai's round-off algorithm
- Area=$\{x_1b_1+x_2b_2 : |x_i| \leq 1/2\}$

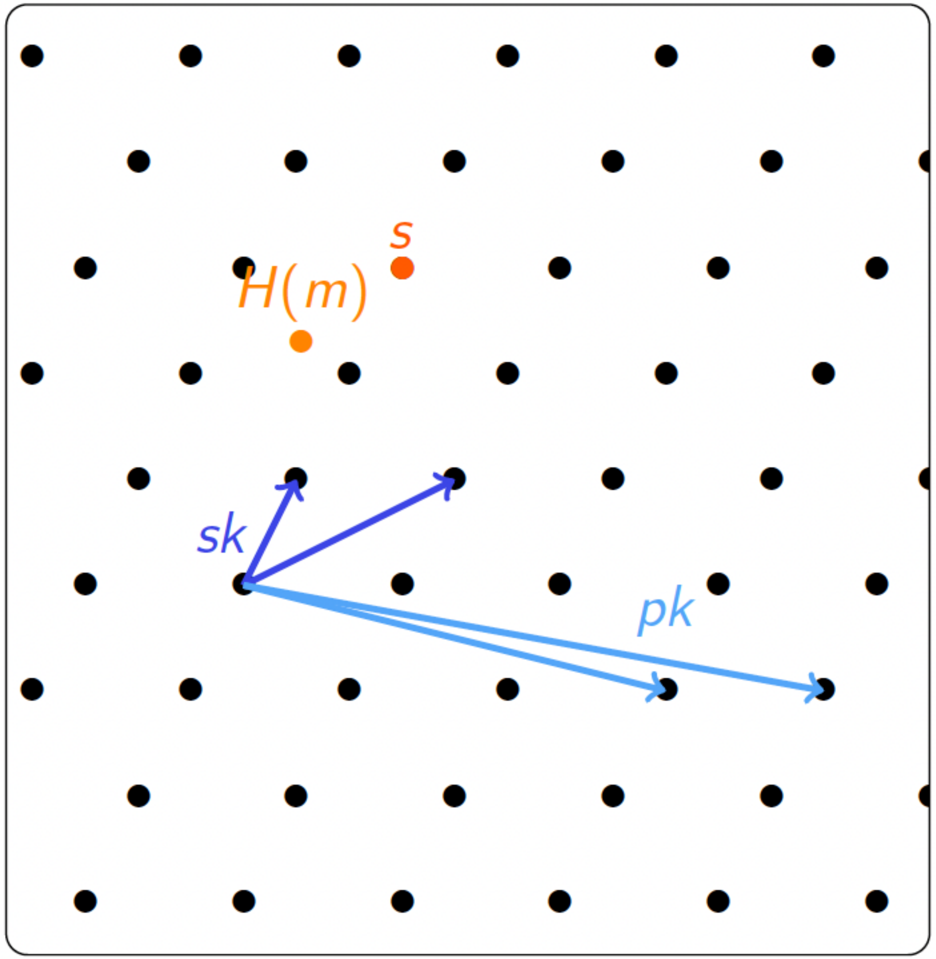
Hash-and-sign: first idea [GGH97]



- KeyGen:
  - pk = bad basis of L (HNF basis)
  - sk = short basis of L
- Sign(m,sk):
  - x = H(m) hash the message
  - output s ∈ L close to H(m)

[GGH97] Goldreich, Goldwasser, Halevi: Public-key cryptosystems from lattice reduction problems. CRYPTO

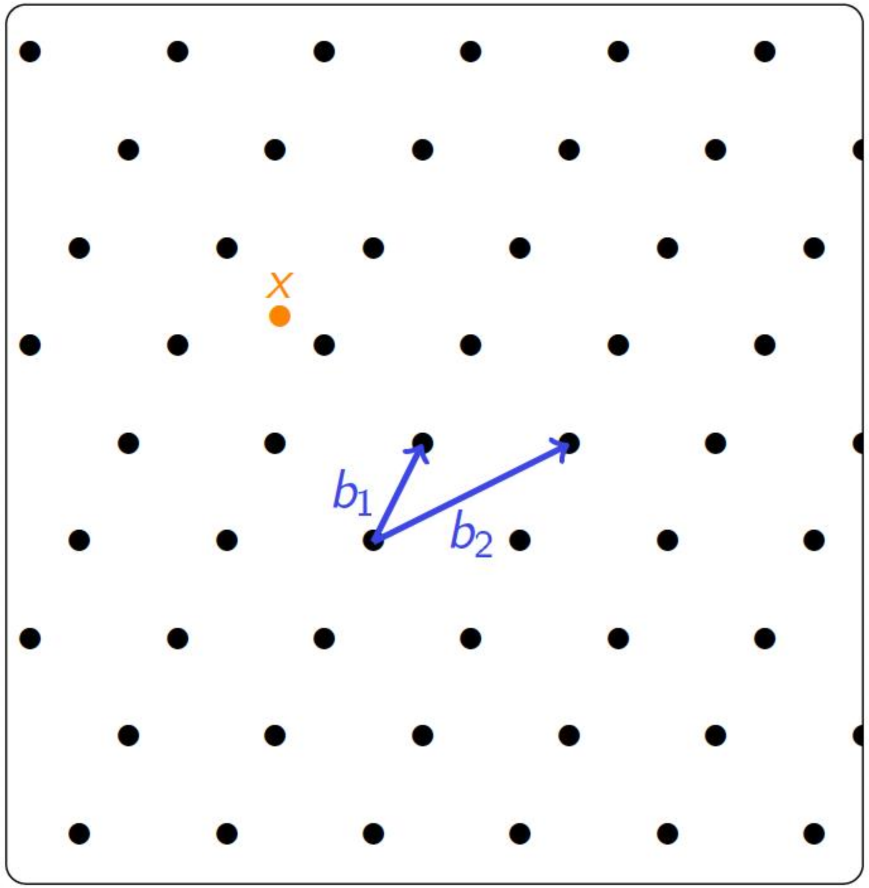
Hash-and-sign: first idea [GGH97]



- KeyGen:
  - pk = bad basis of L (HNF basis)
  - sk = short basis of L
- Sign(m,sk):
  - x = H(m) hash the message
  - output s ∈ L close to H(m)
- Verify(m,s,pk):
  - Check that s ∈ L
  - Check that H(m)-s is small

[GGH97] Goldreich, Goldwasser, Halevi: Public-key cryptosystems from lattice reduction problems. CRYPTO

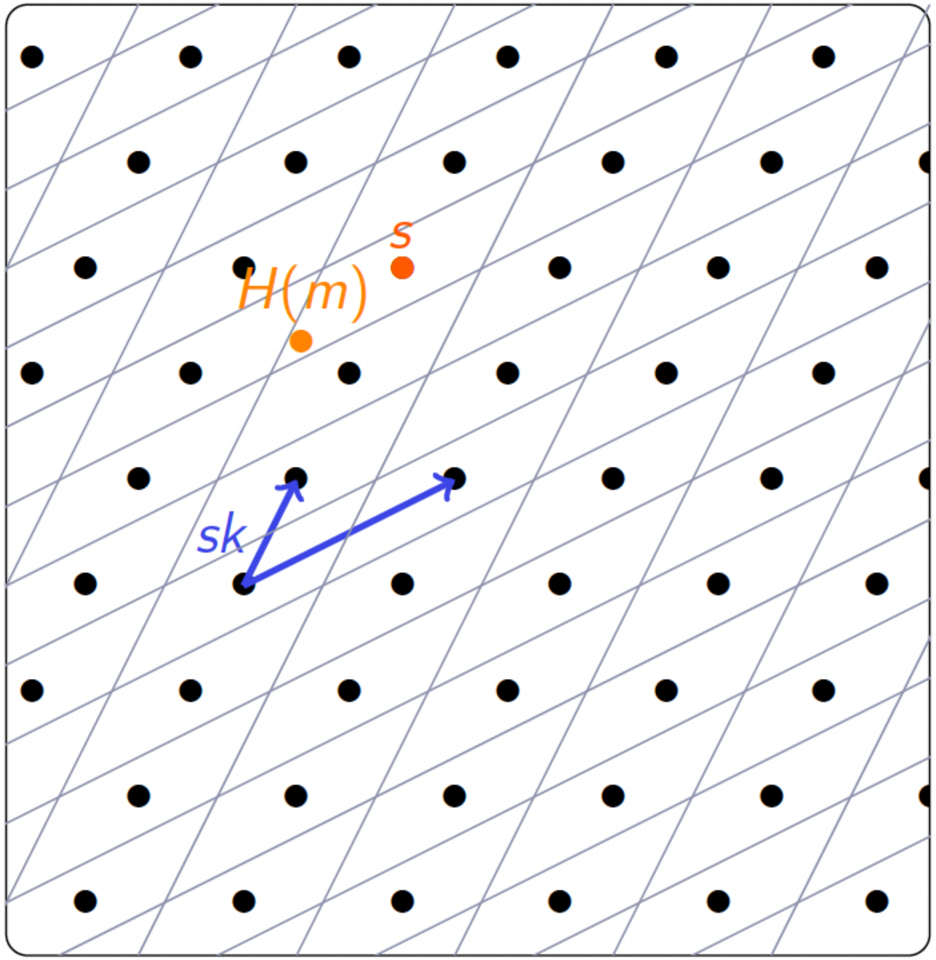
Parallelepided attack [NR06]



- Parallelepided Attack:
  - Ask for a signature s on m

[NR06] Nguyen and Regev. Learning a parallelepided: Cryptanalysis of GGH and NTRU signatures. J. of Cryptology

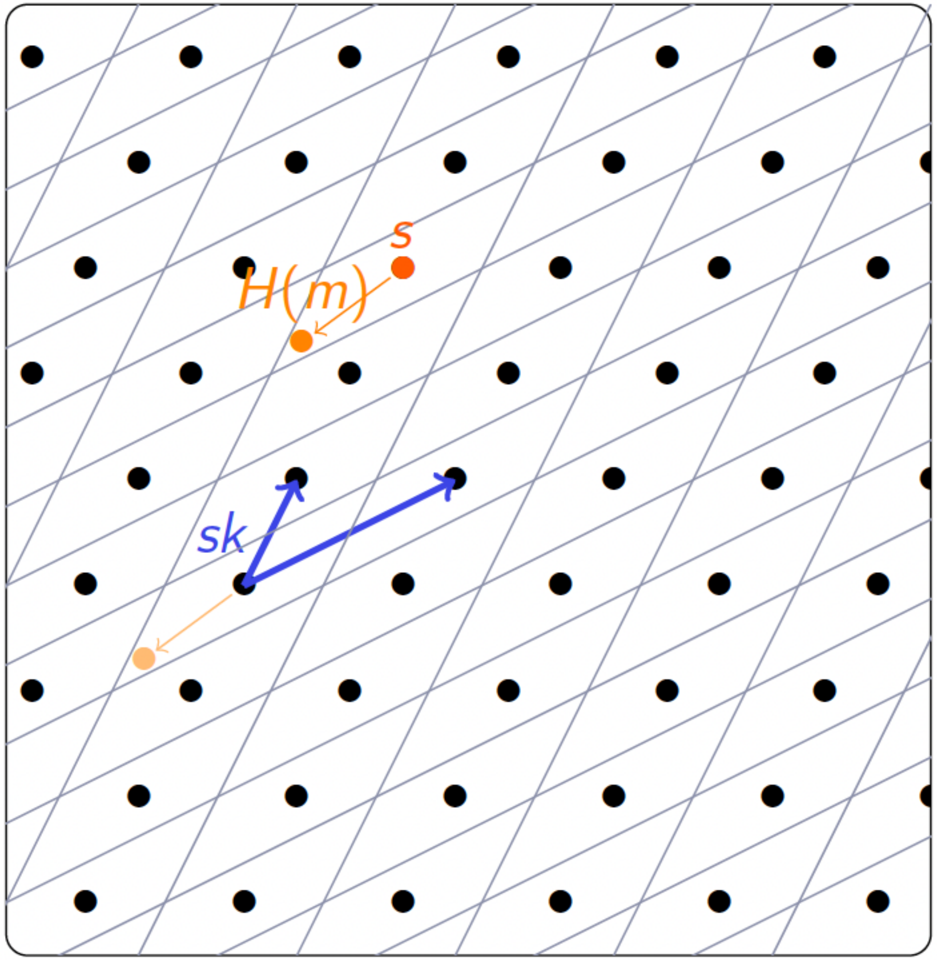
Parallelepided attack [NR06]



- Parallelepided Attack:
  - Ask for a signature s on m

[NR06] Nguyen and Regev. Learning a parallelepided: Cryptanalysis of GGH and NTRU signatures. J. of Cryptology

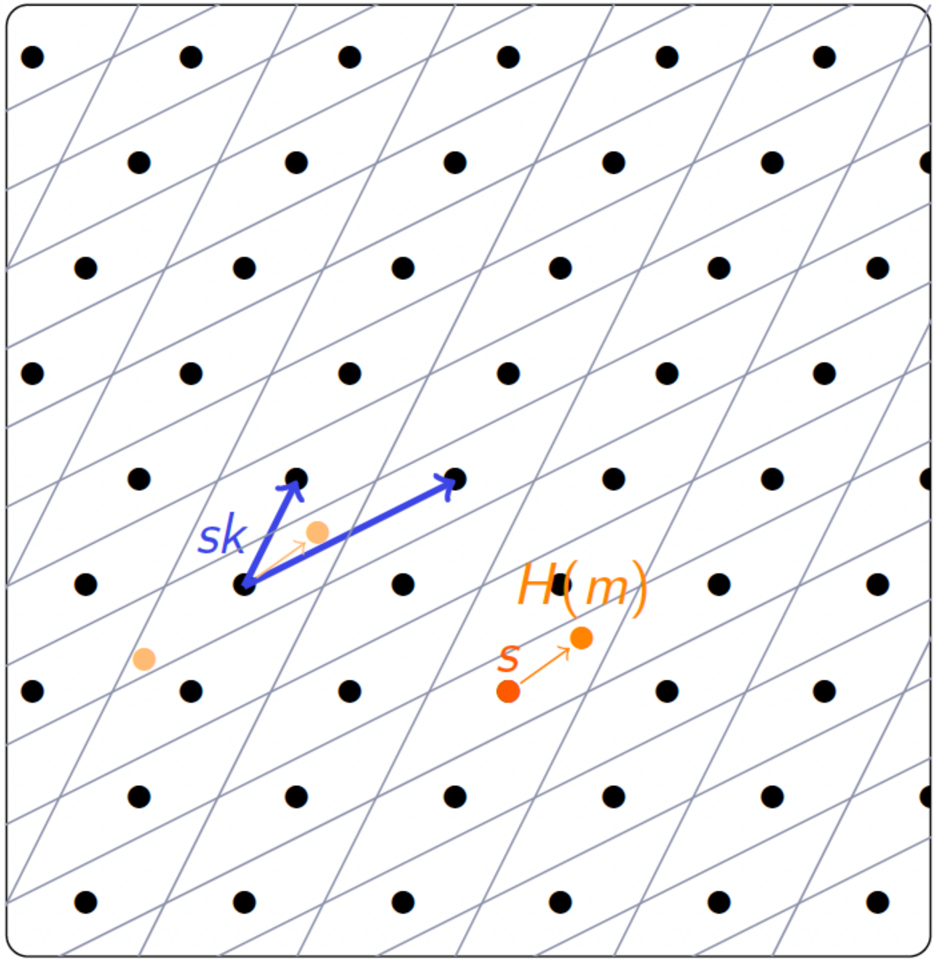
Parallelepiped attack [NR06]



- Parallelepiped Attack:
  - Ask for a signature s on m
  - Plot H(m)-s

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. of Cryptology

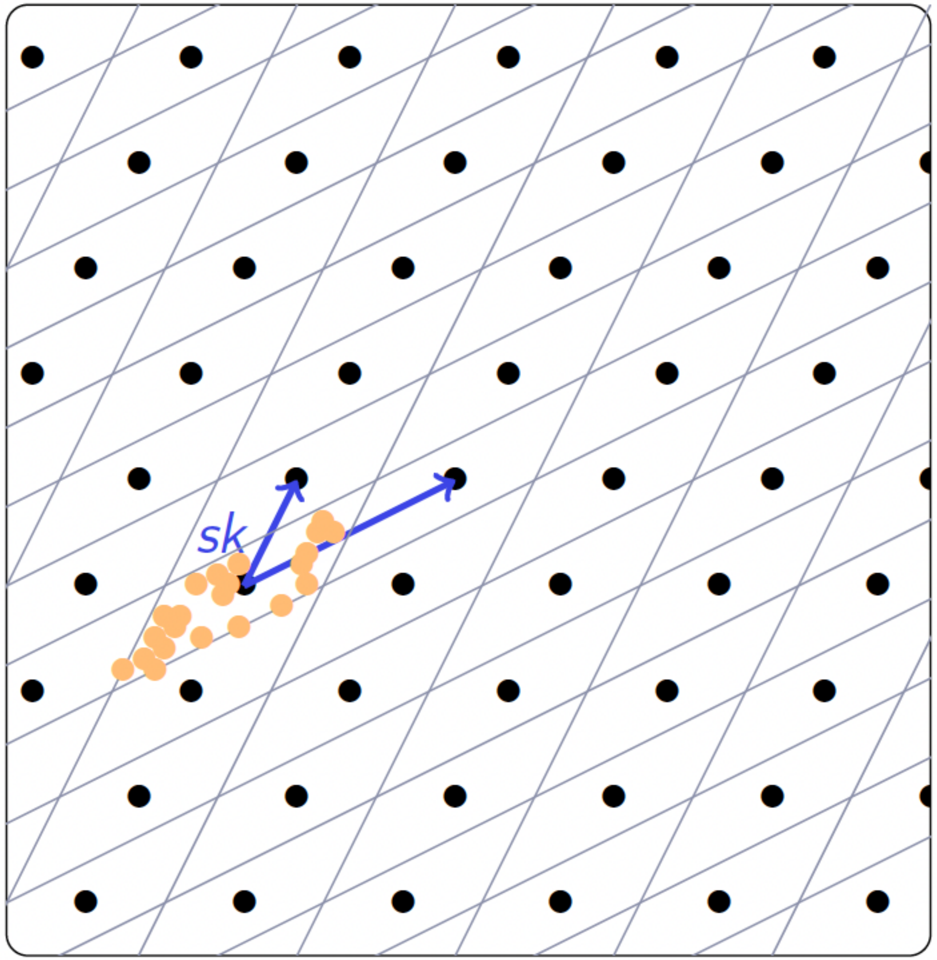
Parallelepided attack [NR06]



- Parallelepided Attack:
  - Ask for a signature s on m
  - Plot H(m)-s
  - Repeat

[NR06] Nguyen and Regev. Learning a parallelepided: Cryptanalysis of GGH and NTRU signatures. J. of Cryptology

Parallelepided attack [NR06]

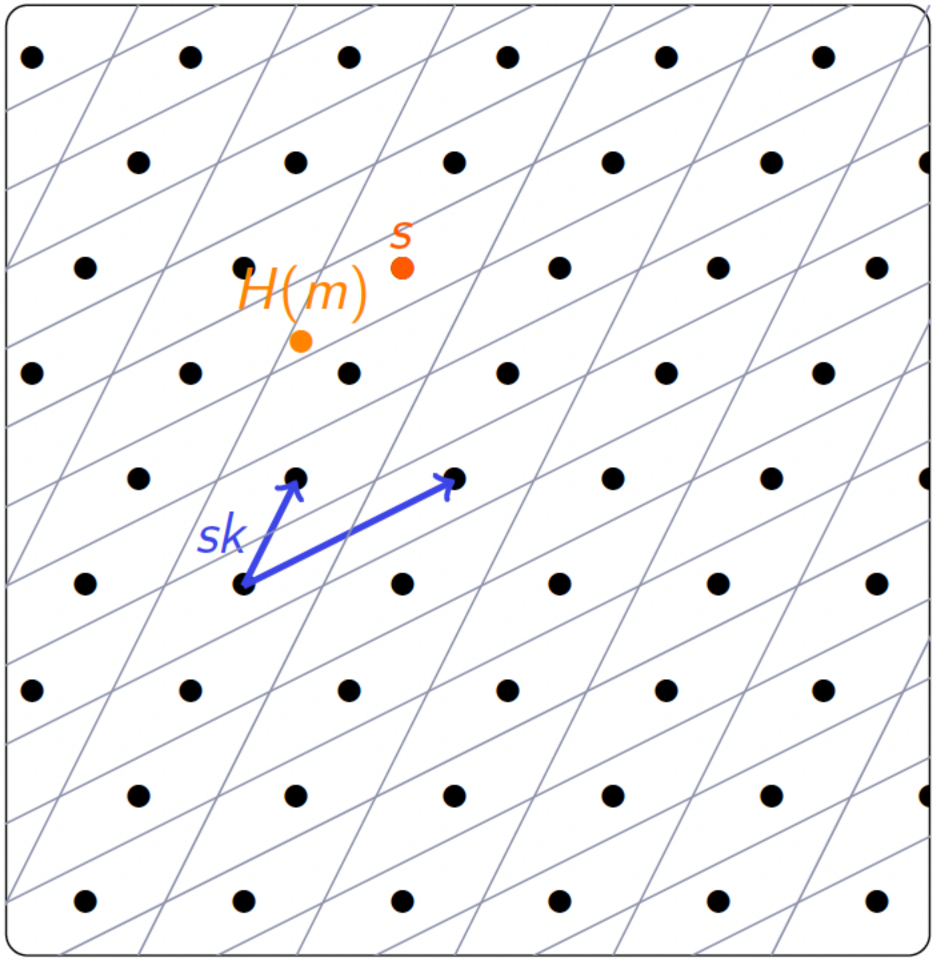


- Parallelepided Attack:
  - Ask for a signature s on m
  - Plot H(m)-s
  - Repeat

- From the shape of the parallelepided, one can recover the short basis

[NR06] Nguyen and Regev. Learning a parallelepided: Cryptanalysis of GGH and NTRU signatures. J. of Cryptology

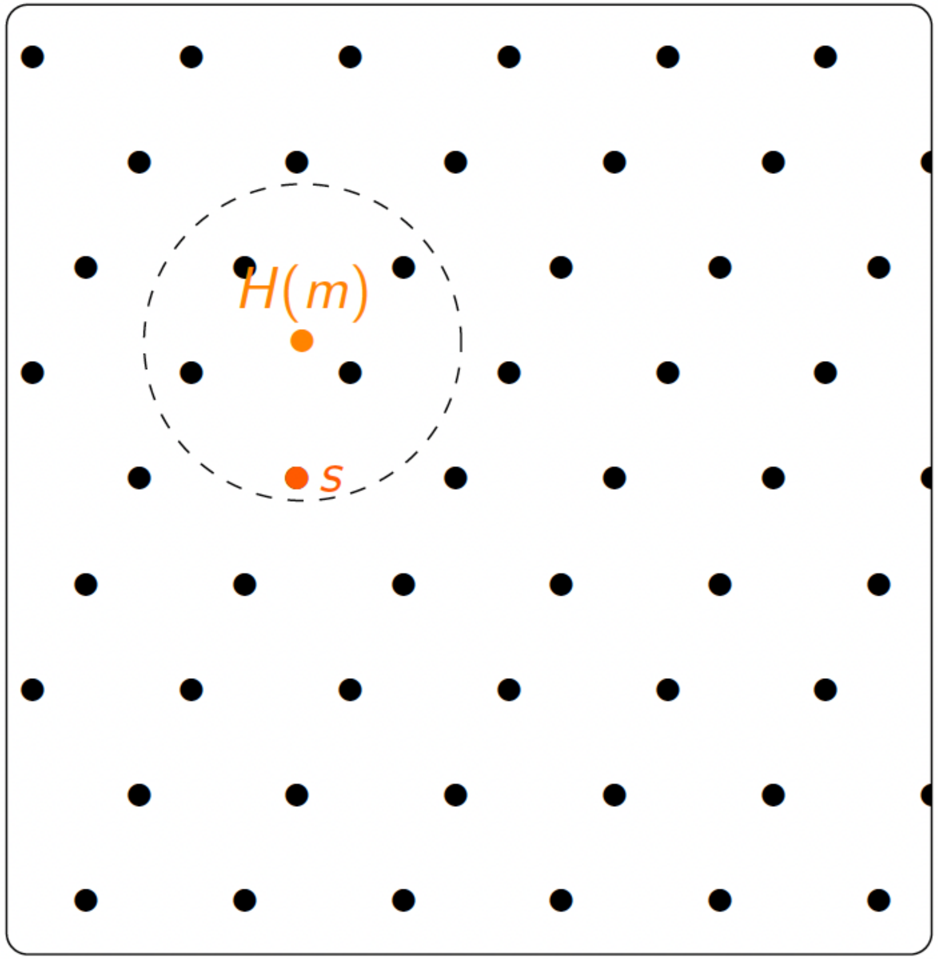
Preventing the attack [GPV08]



- Idea: do not decode deterministically but randomly
- Sign(m,sk):
  - x = H(m) hash the message

[GPV08] Gentry, Peikert,Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC

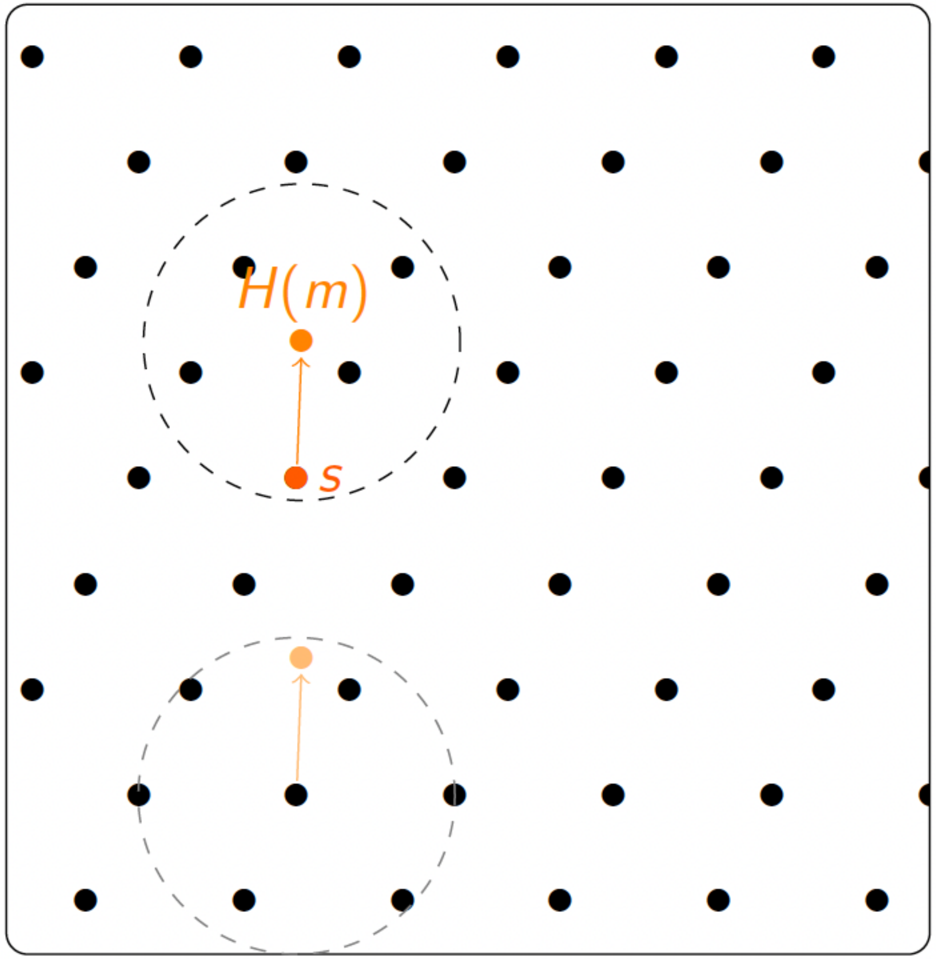
Preventing the attack [GPV08]



- Idea: do not decode deterministically but randomly
- Sign(m,sk):
  - x = H(m) hash the message
  - Sample s ∈ L⋂B_r(x) (small radius r)

[GPV08] Gentry, Peikert,Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC

Preventing the attack [GPV08]

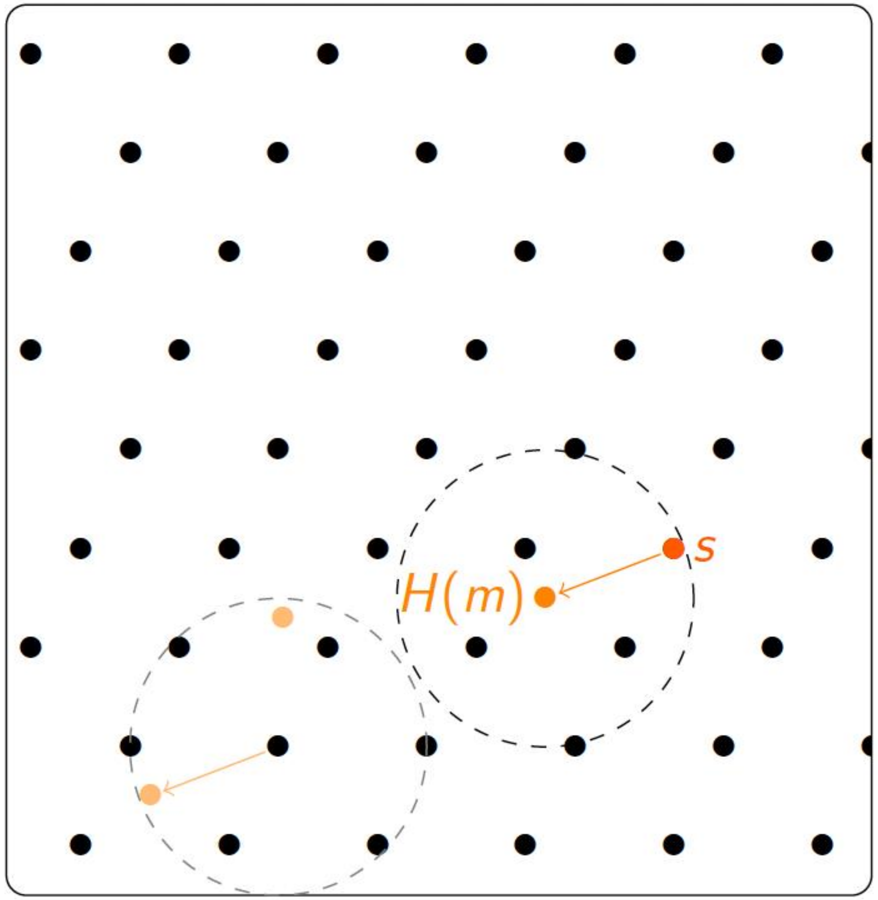


- Idea: do not decode deterministically but randomly

- Sign(m,sk):
  - x = H(m) hash the message
  - Sample s ∈ $L \cap B_r(x)$ (small radius r)

[GPV08] Gentry, Peikert,Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC

Preventing the attack [GPV08]

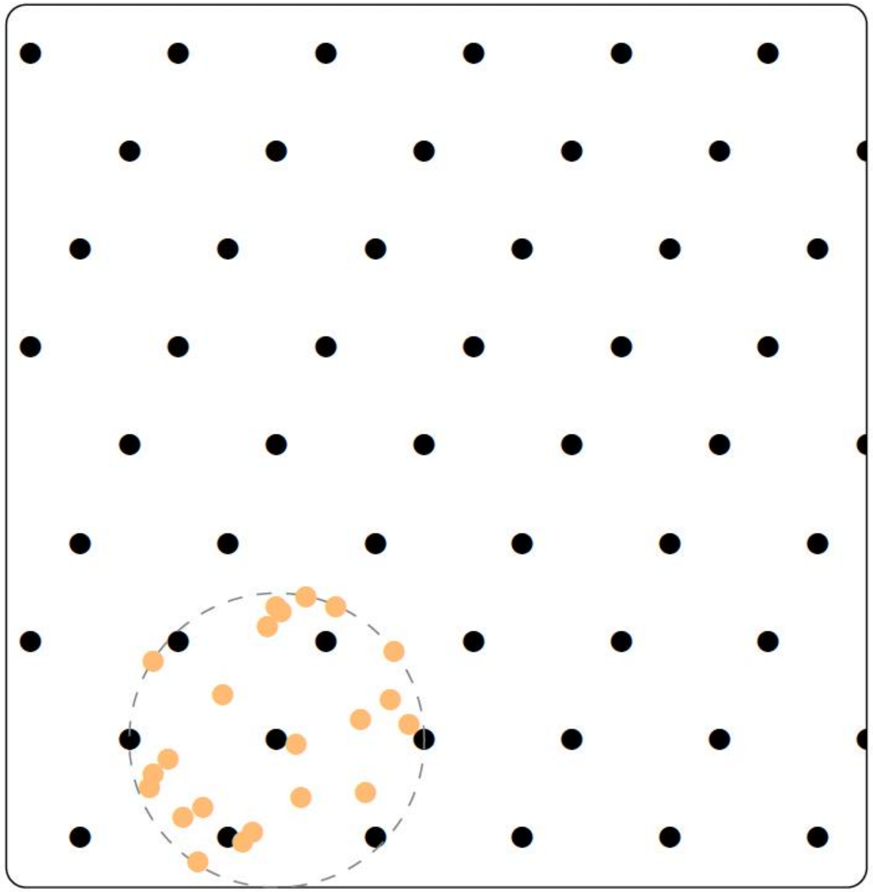


- Idea: do not decode deterministically but randomly

- Sign(m,sk):
  - x = H(m) hash the message
  - Sample s ∈ L⋂B_r(x) (small radius r)

[GPV08] Gentry, Peikert,Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC

Preventing the attack [GPV08]



- Idea: do not decode deterministically but randomly

- Sign(m,sk):
  - x = H(m) hash the message
  - Sample $s \in L \cap B_r(x)$ (small radius r)

How can we sample lattice point around the point x ?

[GPV08] Gentry, Peikert,Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC

NTRU Scheme

NTRU [HPS98]

- Parameters:
  - q prime and large (e.g. q=16411)
  - C≪√q integer (e.g. C=5)
- NTRU instance:
  - Sample f,g random integers in [-C,C] (e.g. f=-2, g=3)
  - Return h=g/f mod q (e.g. h=-5471 mod 16411)
- NTRU Assumption: given h, there is no efficient algorithm that can find u and v s.t. |u|,|v|≤ C and h=u/v mod q
- Careful: for integer it is easy, replace them by polynomials and compute in $Z_q[X]/(X^d+1)$ for d a power of 2

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS

NTRU Cryptosystem

- $q=128$ and $p=3$: p is small and q security parameter (not too large !)
- $R_q=Z_q[X]/(x^d+1)$ where d is a power of 2

- sk: $(f, g) \in R_q$ are polynomials with small coefficients s.t. f is invertible mod p and q, ($f=1+pf'$)
- pk: $h=pg/f$ in R_q
- Encryption: $c=m+hr$ where $m,r \in R_q$ with small L2-norm (OTP $hr \approx U(Z_q)$)
- Decryption:
  - Compute $M = fc \bmod q = mf+rpg \bmod q$
  - As m, f, r and g are short, the equation is true over the Z: $M = mf+rpg$
  - Compute $M \bmod p = m$

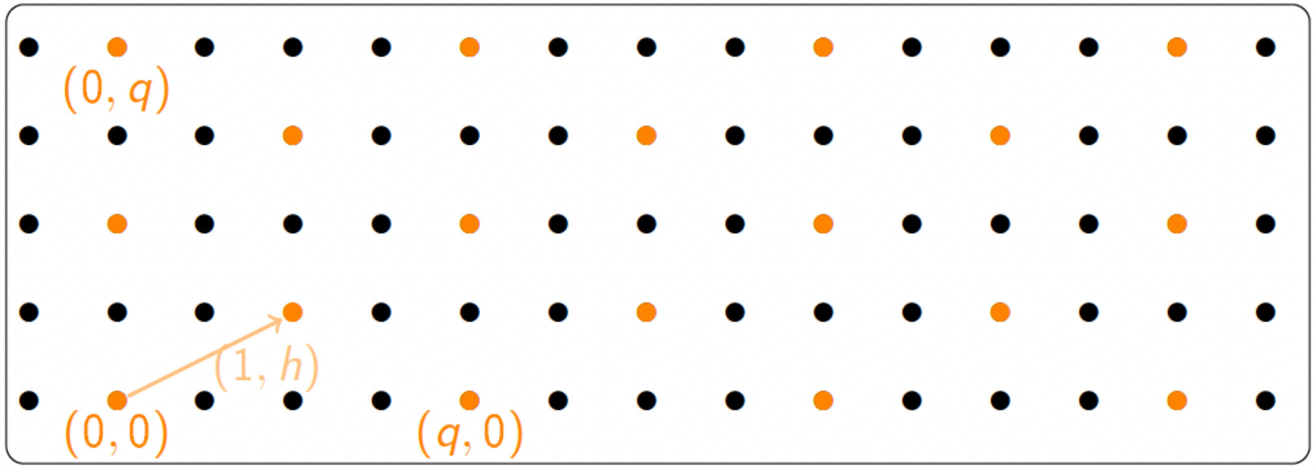
The secret key is composed of f only

Hard lattice from NTRU assumption

- NTRU Assumption: given h, there is no efficient algorithm that can find u and v s.t. |u|,|v|≤ C and h=u/v mod q
- How do we get a hard lattice from this ?
- Let h=f/g mod q an NTRU instance.

$$B_h = \begin{pmatrix} 1 & 0 \\ h & q \end{pmatrix} \quad \text{and} \quad \mathcal{L}_h = \mathcal{L}(B_h) \quad \text{(spanned by the columns of } B_h\text{)}$$

Property:
$(u,v)^T \in L_h$ iff h=v/u (mod q)



Finding a short vector in B_h
⇒ Solving NTRU [CS97]

NTRU Assumption ⇒ no adversary can compute a short basis

NTRU: wrapping up

- We have seen: under the NTRU Assumption, we can generate random hard lattices:
  - Sample f, g polynomials with short coefficients in [-C,C]
  - Compute $h=f/g \bmod (q,X^d+1)$
  - Return L_h spanned by the vectors $(1,h)^T$ and $(0,q)^T$

- L_h has dimension 2, but coefficients are polynomials
- If we write them in some basis $(1,X,\ldots,X^{d-1})$ the matrice is of dimension d and for q and 1, they become qId and Id: Module case (rank=2)
- The dimension of the lattice is 2d (1024 and 2048) for d=512 / 1024

Falcon scheme

- KeyGen(): (pk=A, sk=B)
  1. BA = 0
  2. B has small coefficients
- Sign(m,sk=B):
  1. Compute c s.t. cA=H(m)
  2. v ∈ L(B) close to c
  3. s = c-v
- Verify(m,s,pk):
  - s is short
  - sA=H(m) (exists u s.t. v=uB; sA=(c-v)A=cA-vA=cA-uBA=cA=H(m))

From [DLP14] to Falcon

Two improvements:

- Key Generation [PP19]
- Gaussian Sampler [DP16]

[DLP14] Ducas, Lyubashevsky, Prest. Efficient Identity-Based Encryption over NTRU Lattices. ASIACRYPT 2014

[DP16] Fast Fourier Orthogonalization. ISSAC 2016

[PP19] Pornin, Prest. More Efficient Algorithms for NTRU Key Generation. PKC 2019

Key Generation

Key Generation

$$\mathbf{B} = \left[\begin{array}{c|c} g & -f \\ \hline G & -F \end{array}\right]$$

- Given A=[1,h], find B short s.t. $BA^T=0$ mod (X^d+1,q)
  - Half of the basis: B=[f,-g] satisfies $[f,-g][1,h]^T=f-gh=0$ as h=f/g
  - Full Trapdoor problem:
    - Given $f,g \in Z[X]/(X^d+1)$, find $F,G \in Z[X]/(X^d+1)$ s.t. fG-gF=q mod (X^d+1) det(B)=q
  - Previous techniques: Resultants, HNF, … $O(n^3)$ time and $O(n^2)$ space

- Tower of Rings: $Z \subseteq Z[X]/(X^2+1) \subseteq \ldots \subseteq Z[X]/(X^{d/2}+1) \subseteq Z[X]/(X^d+1)$
  - Field norm: navigate along this tower: $Q_d=Q[X]/(X^d+1)$

$N: Q_d \rightarrow Q_{d/2}$

$f \mapsto ff^x$, where $f^x(x)=f(-x)$

Algorithm for solving NTRU equation

Problem: Given $f, g \in \mathbb{Z}[x]/(x^d + 1)$, find $F, G \in \mathbb{Z}[x]/(x^d + 1)$ such that:

$$f \cdot G - g \cdot F = q$$

If we can solve the problem projected over $\mathcal{Z}_{d/2}$, i.e.:

$$\mathrm{N}_{\mathcal{Z}_d/\mathcal{Z}_{d/2}}(f) \cdot G' - \mathrm{N}_{\mathcal{Z}_d/\mathcal{Z}_{d/2}}(g) \cdot F' = 1$$

for some F', G', then we have this relationship over $\mathcal{Z}_d$:

$$f \cdot (f^{\times} G') - g \cdot (g^{\times} F') = 1$$

Recursive Algorithm

$$
\begin{array}{ccc}
\mathcal{Z}_d & \ni & f, g \\
\cup\!\nmid & & \downarrow \\
\mathcal{Z}_{d/2} & \ni & \mathrm{N}_{\mathcal{Z}_d/\mathcal{Z}_{d/2}}(f), \mathrm{N}_{\mathcal{Z}_d/\mathcal{Z}_{d/2}}(g) \\
\cup\!\nmid & & \downarrow \\
\mathcal{Z}_{d/4} & \ni & \mathrm{N}_{\mathcal{Z}_d/\mathcal{Z}_{d/4}}(f), \mathrm{N}_{\mathcal{Z}_d/\mathcal{Z}_{d/4}}(g) \\
\cup\!\nmid & & \downarrow \\
\vdots & \vdots & \vdots \\
\cup\!\nmid & & \downarrow \\
\mathbb{Z} & \ni & \mathrm{N}_{\mathcal{Z}_d/\mathbb{Z}}(f), \mathrm{N}_{\mathcal{Z}_d/\mathbb{Z}}(g) \quad \to \quad F^{[\ell]}, G^{[\ell]}
\end{array}
$$

From f to N(f): the coefficients Of N(f) are twice larger, but the degree is half

At the bottom of the recursion, we solve the equation over Z with very large coefficients

Then, we go up the tower

Recursive algorithm

Gain: x100 in memory (3MB => 30KB)

Gain: x100 in time (2s => 20ms)

Algorithm 1 $\mathsf{TowerSolverR}_{n,q}(f,g)$

Require: $f, g \in \mathbb{Z}[x]/(x^n+1)$ with n a power of two

Ensure: Polynomials F, G such that the equation 1 is verified

if $n = 1$ **then**

  Compute $u, v \in \mathbb{Z}$ such that $uf - vg = 1$ ← Extended Euclidean Algorithm over Z

  $(F, G) \leftarrow (v, u)$

  return (F, G)

else

  $f' \leftarrow \mathrm{N}(f)$ ▷ $f', g', F', G' \in \mathbb{Z}[x]/(x^{n/2}+1)$

  $g' \leftarrow \mathrm{N}(g)$

  $(F', G') \leftarrow \mathsf{TowerSolverR}_{n/2,q}(f', g')$ ← Recursive Call: go down in the tower

  $F \leftarrow g^{\times}(x)F'(x^2)$ ▷ $F, G \in \mathbb{Z}[x]/(x^n+1)$ ← Go up in the tower

  $G \leftarrow f^{\times}(x)G'(x^2)$

  Reduce (F, G) with respect to (f, g) ← Lattice reduction step

return (F, G)

If $\|(f,g)\|_\infty$ <C=20, $\|(F,G)\|_\infty$ <120

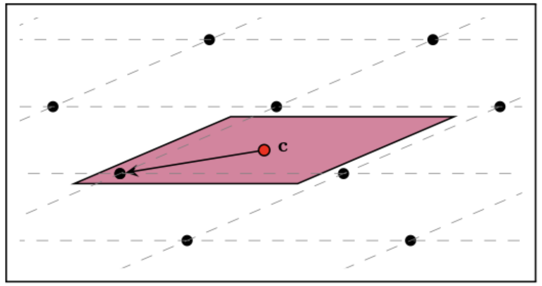
Gaussian Sampling algorithms

Closest vector problem: 2 algorithms

Round-Off Algorithm:

1. $\mathbf{t} \leftarrow \mathbf{c} \cdot \mathbf{B}^{-1}$
2. $\mathbf{z} \leftarrow \lfloor \mathbf{t} \rceil$
3. Output $\mathbf{v} \leftarrow \mathbf{z} \cdot \mathbf{B}$

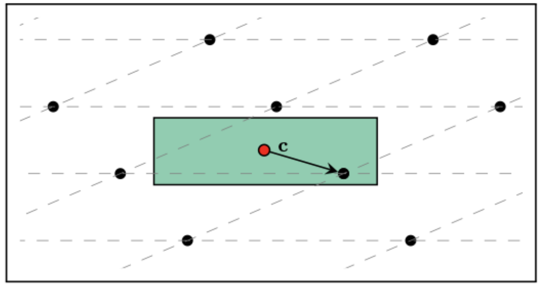
Output:



Nearest Plane Algorithm:[1]

1. $\mathbf{t} \leftarrow \mathbf{c} \cdot \mathbf{B}^{-1}$
2. For $j = n$ down to 1:
   1. $\hat{t}_j \leftarrow t_j + \sum_{i>j} (t_i - z_i) \cdot L_{i,j}$
   2. $z_j \leftarrow \lfloor \hat{t}_j \rceil$
3. Output $\mathbf{v} \leftarrow \mathbf{z} \cdot \mathbf{B}$

Output:



[1] Requires precomputing the Gram-Schmidt orthogonalisation (GSO) of **B**: $\mathbf{B} = \mathbf{L} \cdot \tilde{\mathbf{B}}$.

Gaussian Sampling

Given $\mathbf{B} \in \mathcal{Z}_d^{n\times n}$ and $\mathbf{t} \in \mathcal{Q}_d^n$, compute $\mathbf{z} \in \mathcal{Z}_d^n$ such that $||(\mathbf{z} - \mathbf{t}) \cdot \mathbf{B}||$ is small.

Algorithm 3 $\mathsf{Klein}_{\mathbf{L},\sigma}(\mathbf{t})$

Require: $\sigma \geq \eta_\epsilon(\mathbb{Z}^n) \cdot ||\mathbf{B}||_{\mathsf{GS}}$, the Gram-Schmidt orthogonalization $\mathbf{B} = \mathbf{L} \cdot \tilde{\mathbf{B}}$, the values $\sigma_j = \sigma/||\tilde{\mathbf{b}}_j||$ and a target $\mathbf{t}$

Ensure: A vector $\mathbf{z}$ such that $\mathbf{zB} \leftarrow D_{\Lambda(\mathbf{B}),\sigma,\mathbf{tB}}$

for $j = n, \ldots, 1$ **do**

$c_j \leftarrow t_j + \sum_{i>j}(t_j - z_j)L_{ij}$

$z_j \leftarrow D_{\mathbb{Z},\sigma_j,c_j}$

return z

<u>3 algorithmic techniques:</u>

- Randomized nearest plane [Bab85, GPV08]: High quality, $O((nd)^2)$ operations
- Randomized round-off [Bab85, Pei10]: Lower quality, $O(n^2 d \log d)$ operations
- Fast Fourier orthogonalization [DP16]: High quality, $O(n^2 d \log d)$ operations

Fast Fourier Orthogonalization [DP16]

Consider the simplified case where we want this to be small:

$$(z - t) \cdot b$$

Using the ring isomorphism $\mathcal{Q}_d \cong (\mathcal{Q}_{d/2})^2$, this is equivalent to:

$$\left[\; z_e - t_e \;\middle|\; z_o - t_o \;\right] \cdot \underbrace{\left[\begin{array}{c|c} b_e & b_o \\ \hline x b_o & b_e \end{array}\right]}_{\mathbf{B}}$$

Why this is nice:

- We can orthogonalize the second row of **B** w.r.t. to the first one:

$$\tilde{\mathbf{b}}_2 \leftarrow \mathbf{b}_2 - \underbrace{\frac{\langle \mathbf{b}_2, \mathbf{b}_1 \rangle}{\mathbf{b}_2, \mathbf{b}_1}}_{L_{2,1}} \cdot \mathbf{b}_1$$

- We can apply this “break and orthogonalize” trick recursively.

GSO **B** = **LB**’ is equivalent to **BB*** = **LB’B’*L***

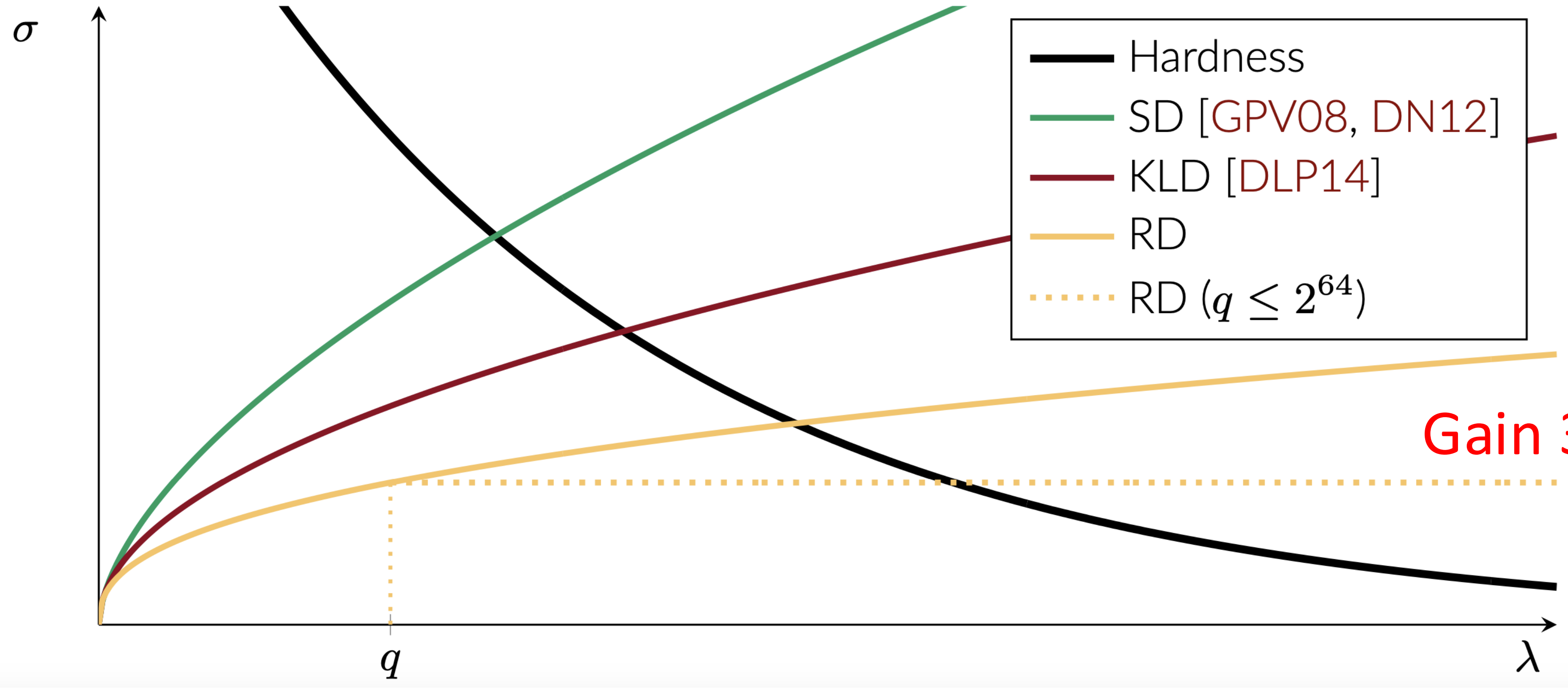
Recursive strategy O(dlog^{2}d)

Working in the FFT domain reduces the complexity to O(d log d)

Security [P17]

Rényi divergence is more efficient than SD/KLD, interesting when q is smaller

1. σ too large => Klein does not solve a hard problem and it is useless in crypto
2. σ too small => Klein does not behave like a perfect Gaussian oracle



Gain 30 bits of security

[DN12] Faster Gaussian Lattice Sampling Using Lazy Floating-Point Arithmetic. ASIACRYPT 2012

[P17] Prest. Sharper Bounds in Lattice-Based Cryptography Using the Rénui Divergence. ASIACRYPT 2017

Pros and cons of Falcon

- Falcon signature:
  - Post-quantum security
  - Fast (comparable to RSA / EdDSA)
  - Relatively compact (about 3 times RSA, 15 times EdDSA)

- But there are still important open question and many improvements
- Requires floating point
  - Not all devices have floating-point units
- Difficult to mask
  - Can be subject to SCA
  - See Raccoon if you want a masking-friendly signature scheme
- Antrag and Mitaka avoid FFO and simpler sampler without floating

[Raccoon] del Pino, Katsumata, Maller, Mouhartem, Prest, Rossi, Saarinen. Technical report, NIST
[Mitaka] Espitau, F, Gerard, Rossi, Takahashi, Tibouchi, Wallet, Yu, EUROCRYPT 2022
[Anrtag] Espitau, Nguyen, Sun, Tibouchi, Wallet. ASIACRYPT 2023

Conclusion

- **FALCON:** ***Fa**st Fourier **l**attice-based **co**mpact signatures over **N**TRU*
- Other uses of Falcon: IBE scheme
- **Careful use of tower of subrings in $Z_q[X]/(X^d+1)$**
  - Other use: LLL over $Z_q[X]/(X^d+1)$ [KEF20]

- Security Results: NTRUSign based on R-SIS [SS11]
- Interesting results show connection between the NTRU Assumption and hard lattice problems [PS21, FPS22]

[KEF20] Kichner, Espitau, Fouque. Fast Reduction of Algebraic Lattices over Cyclotomic Fields, CRYPTO 2020
[SS11] Stehlé, Steinfeld. Making NTRU NTRUEncrypt and NTRUSign as Secure as Standard Worst-Case Problems Over Ideal Lattices.EUROCRYPT 2011
[PS21] Pellet—Mary, Stehlé. On the hardness of the NTRU Problem, ASIACRYPT 2021
[FPS22] Felderhoff, Pellet—Mary, Stehlé, On Module Unique-SVP and NTRU, ASIACRYPT 22