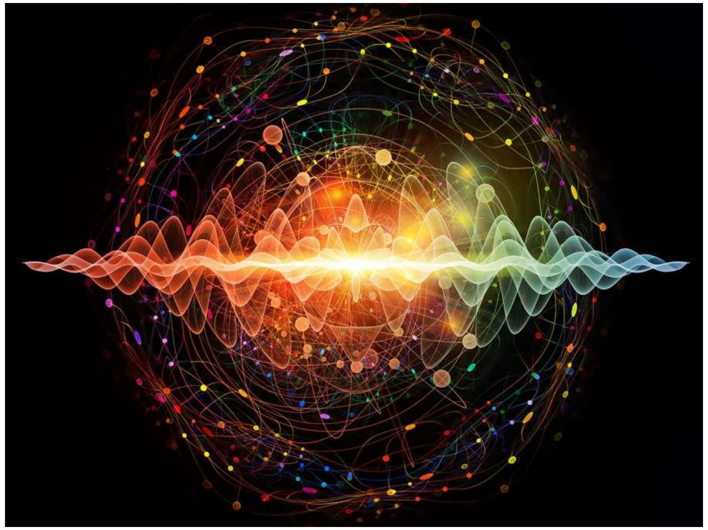
CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Digital Signatures from ID scheme: Lattice Challenges & Open Problems

Dipayan Das

Why Lattice-based cryptography???



1. Post-quantum candidate
2. Worst-case to average-case reduction
3. Advanced cryptographic primitives (like F.H.E)
4. 12 (9E+3S)/26 (17E+9S) second round candidates of the ongoing NIST post-quantum standardization process are lattice-based. 5(3E+2S)/7 finalist+2 (2E+0S)/5 alternative candidates in third round (updated on 22nd July, 2020)

For more details: https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions

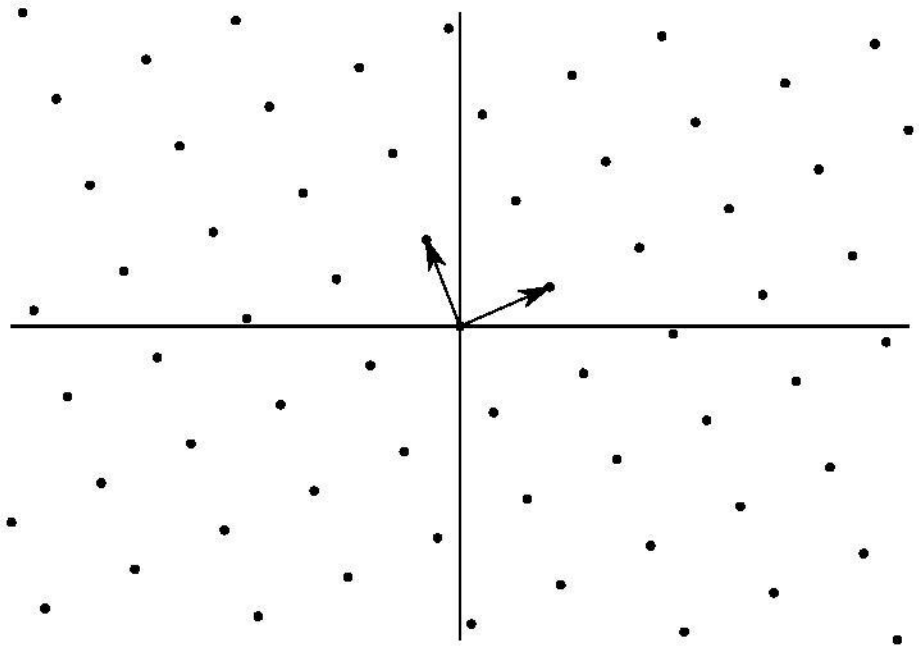
Other candidates are code-based, multivariate, hash-based, Zero knowledge proofs.

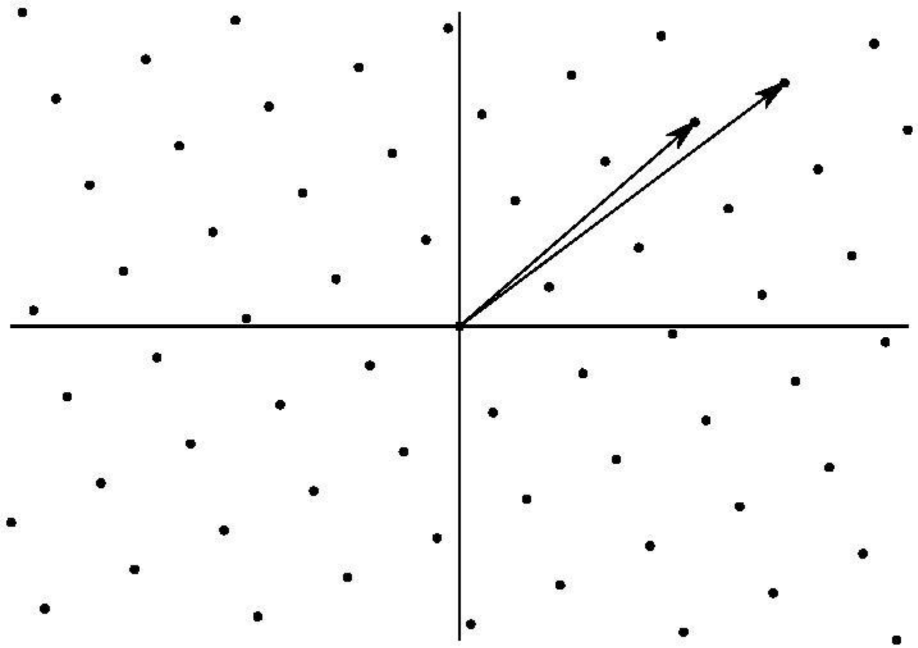
Conclusion Encryption schemes seems to be easy to construct than **Signature** schemes.

Pic courtesy: https://www.britannica.com/quiz/quantum-mechanics

Lattice

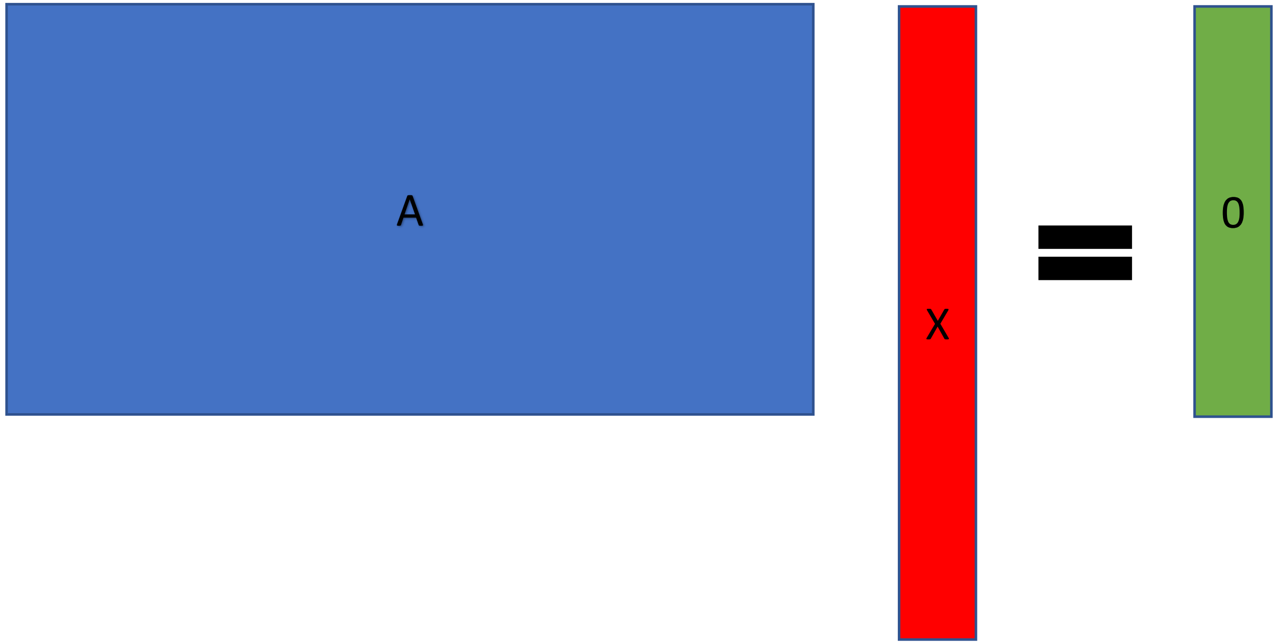
Given k linearly independent vectors $\boldsymbol{B} = \{\boldsymbol{b_1}, \boldsymbol{b_2}, \dots, \boldsymbol{b_k}\}$ in $\mathbb{Z}^n$, the ***Lattice*** $\boldsymbol{L}$ generated by the vectors $\boldsymbol{B}$ is defined as $\boldsymbol{L} = \boldsymbol{L}(\boldsymbol{B}) = \{\sum_{i=1}^{k} a_i \boldsymbol{b_i} : a_i \in \mathbb{Z}\}$





Short Integer Solution (SIS) problem [Ajt'96]

- Given uniform $A \in \mathbb{Z}_q^{n \times m}$, find non-zero x such that $Ax = 0 \ mod\ q$

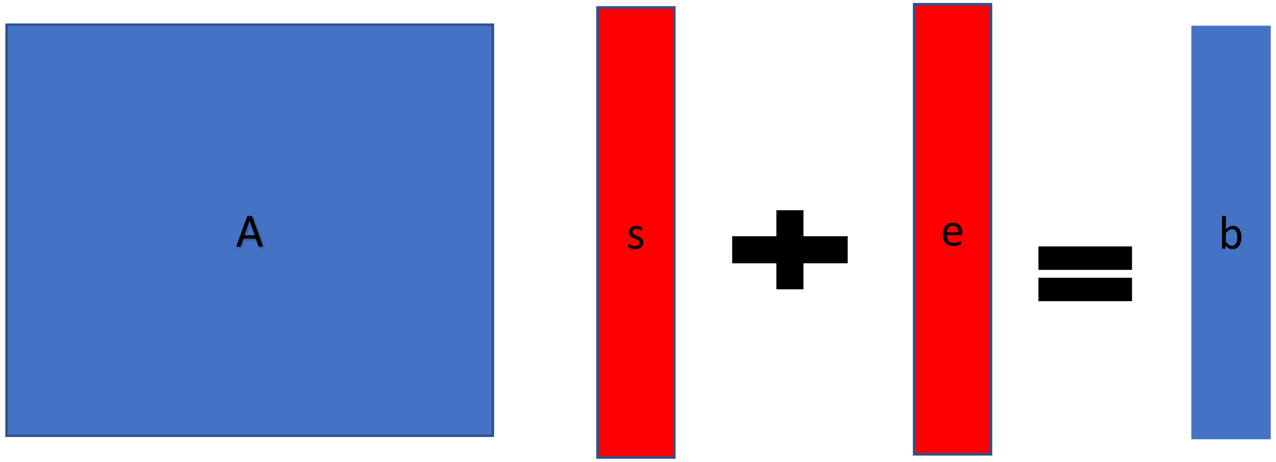


Some Observations

- The SIS problem without the norm constraint is “easy” to solve.
- We also can have inhomogenous version (ISIS): $A\mathbf{s} = t$
- SIS $<_{poly}$ ISIS

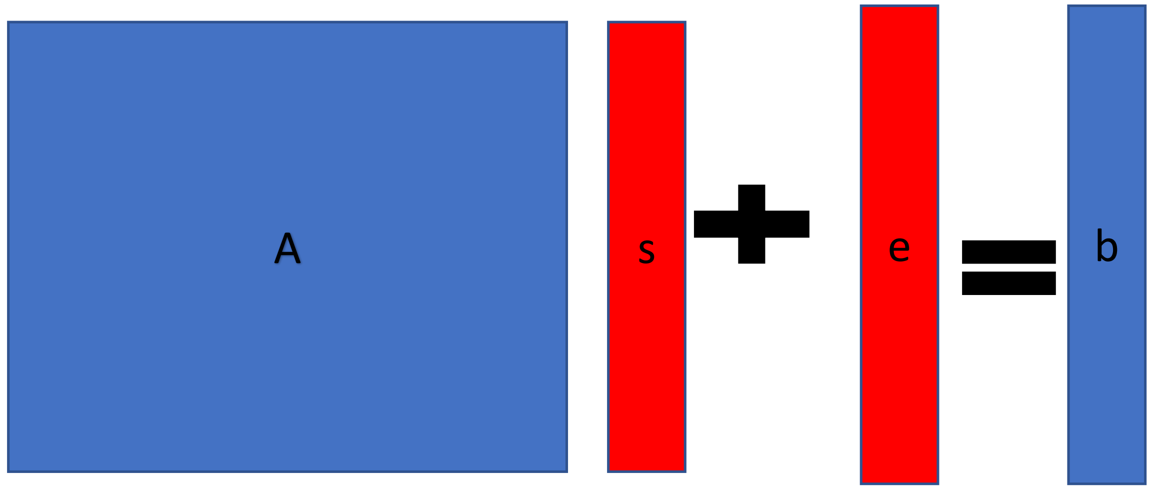
Learning with Errors (LWE) problem [Reg'05]

- Given uniform $A \in \mathbb{Z}_q^{n\times n}, b,$ find non-zero (s, e) such that $As + e = b \bmod q$

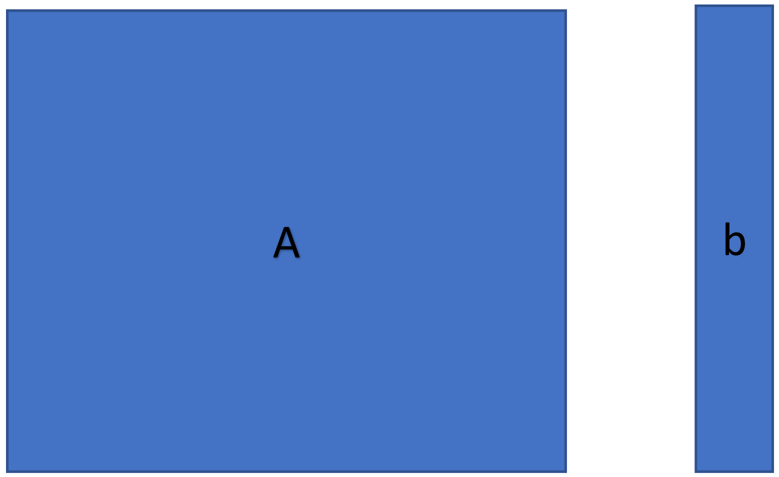


Decision Learning with Errors(LWE) problem [Reg05,ACPS09]

- Given uniform $(\mathrm{A}, b) \in \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_{\mathrm{q}}^{m}$, decide if $\mathrm{b} = As + e = b \; mod \; q$ or b is uniform



LWE Distribution



Uniform distribution

Search LWE to Decision LWE

$$\begin{bmatrix} a_0 + l & a_1 & \cdots \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ \vdots \end{bmatrix} + e_0 = b + l s_0$$

- Let q be a prime
- For any small $\mathrm{k} \in Z_q$, transform $(\boldsymbol{a} + (l, 0, \dots, 0), b + lk)$ for $l \leftarrow Z_q$
- If $k = s_0$: then LWE samples map to LWE samples
- Otherwise uniform sample maps to uniform!
- Since $\boldsymbol{s}$ is small, we have the right guess in a small number of guesses.
- Repeat it for all coordinates to recover $\boldsymbol{s} = (s_0, \dots s_{n-1})$

Some Observations

- Search LWE $<_{poly}$ Decision LWE

The reduction is non-tight in both running time and advantage [Reg'05,MM'11,BLP+13].

SIS/LWE as a lattice problem

- $L^{\perp}_{A,q}(x) = \{x: Ax = 0 \; mod \; q\}$

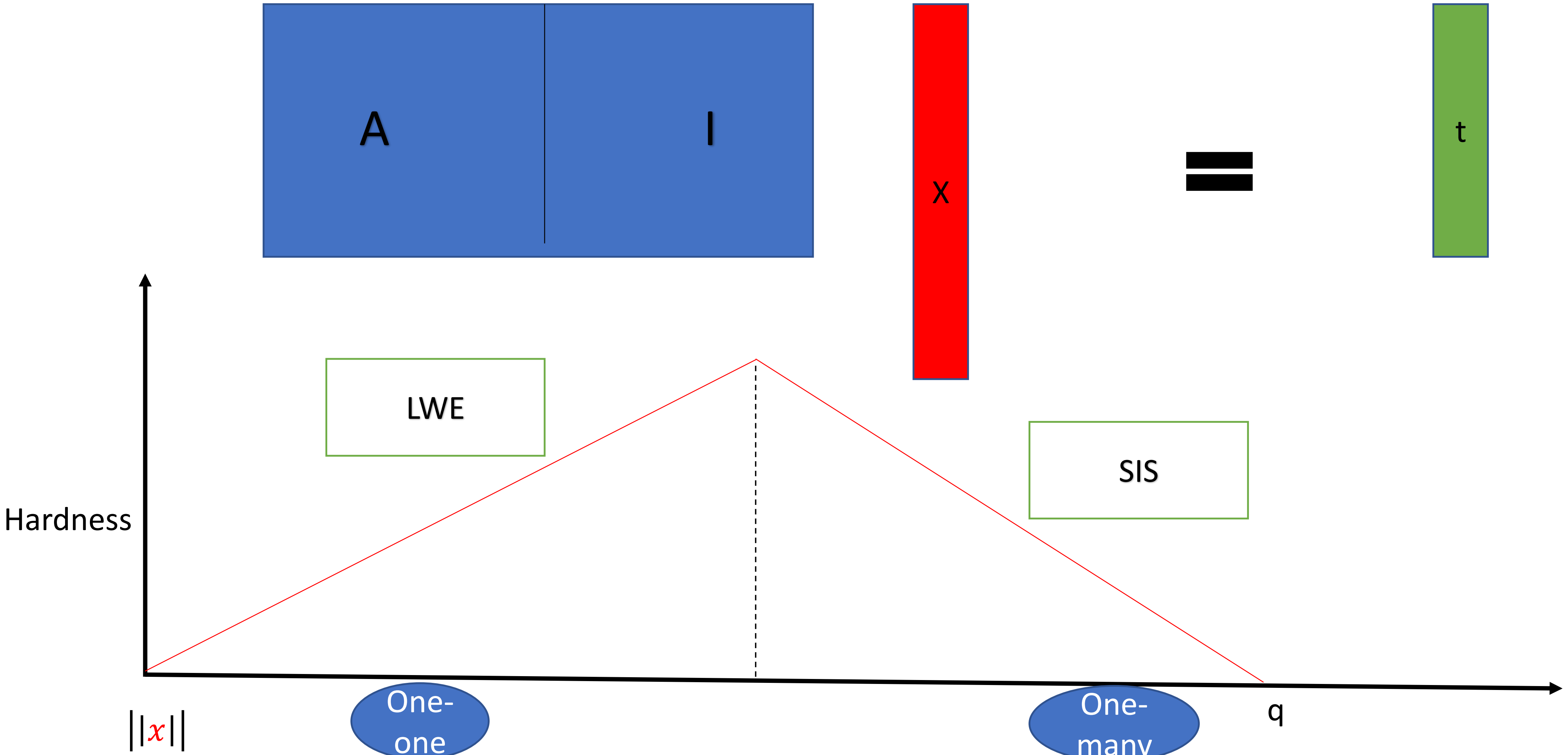
SVP on $L^{\perp}_{A,q}$ implies a solution to the SIS problem

- $\mathrm{L_{A,q}}(x) = \{x: x = As \; mod \; q\}$ for some fixed s

CVP on $L_{A,q}$ implies solution to the LWE problem

Not exactly CVP since e is small. We call it BDD problem in lattice terminology

SIS/LWE in a nutshell



(Dis)Advantages of SIS/LWE based constructions

- Asymptotic worst-case security
- Only linear operations required for crypto constructions

- Storing A requires mn elements of $\mathbb{Z}_q$
- Long keys
- Matrix multiplication is slow
- Inefficient crypto constructions

Some Algebraic Variants

- Polynomial Ring LWE/SIS [LPR'10] (Pros: Storage & operations, Cons: Slower ring multiplication, worst-case hardness, Probably more algebraic): Rotation matrix
- Middle-product LWE [RSSS'17,B**D**H+20](Pros: Based on the hardness of exponentially many Ring LWE, Cons: large dimension, slower multiplication): Large Toeplitz matrix
- Module LWE/SIS[LS'16]: Repeated small Circular matrix (Pros: Faster ring multiplication, Cons: Challenge space)

Replace elements of Z_q by $Z_q[x]/(x^{2^k}+1)$

Polynomial ring $Z_q[x]/(x^4+1)$

- Addition: Coordinate wise
- Multiplication: Rotation wise (mod x^4+1)

Example:

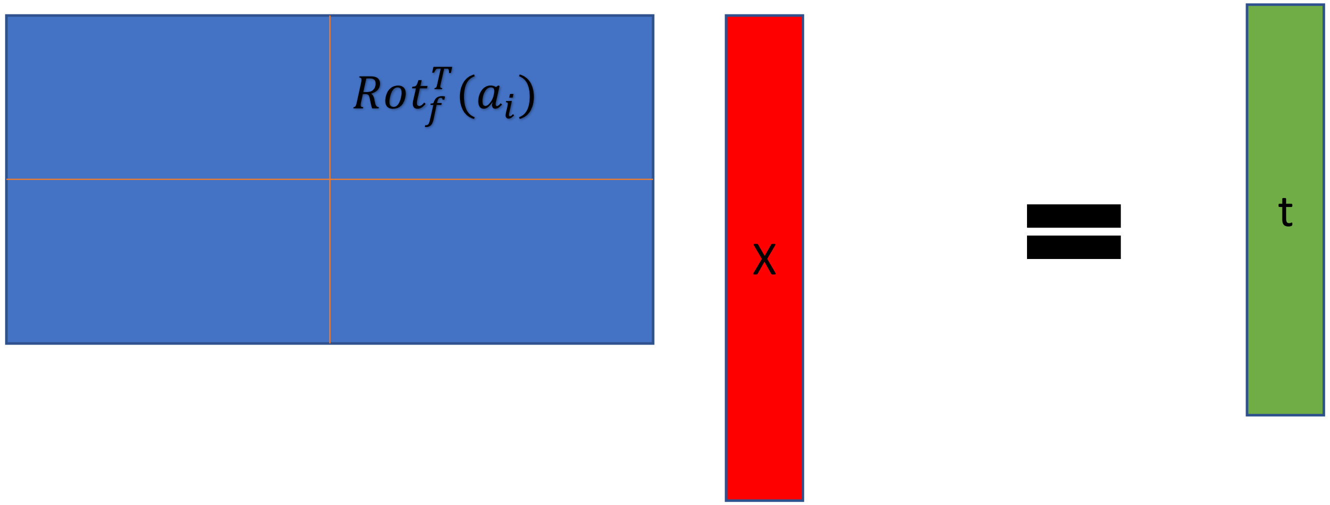
$$\left(2x^2+3x+1\right)*\left(x^2+2\right)=3x^3+5x^2+6x+1$$

Note: Reducing by x^4+1 doesn't change the coefficients by much, but for some polynomials it can change a lot. Technically this is called EXPANSION FACTOR.

Such polynomials are not useful for crypto!!

Module SIS/LWE problem [LS'16]

- Let $R_q = Z_q[x]/(f = x^d + 1)$
- Given uniform $A \in R_q^{n \times m}$, $t \in R_q^n$, find non-zero $x \in R_q^m$ such that $Ax = t \bmod q$



- Worst-case to average-case connection over $Z[x]/(x^d + 1)$

Some notes on the ring $\mathrm{R} = Z[x]/(f)$

The polynomial $f(x)$ must satisfy

- Irreducibility over Z
- Bounded Expansion factor

What else?

Could there be some $f'(x)$ that is easier for solving SIS/LWE?

What is the Hardest instantiation?

Expansion factor comparison

```
f(x)=x^128+1

e(x)=x^127 - x^124 + x^123 + x^121 + x^119 + x^118 + x^117 - x^116 + x^115 + x^112 +
x^111 - x^109 - x^108 - x^106 + x^105 + x^104 - x^102 - x^101 - x^99 + x^98 + x^97 -
x^96 + x^94 + x^92 - x^91 - x^87 + x^86 - x^84 + x^83 - x^82 + x^81 - x^79 - x^78 +
x^75 + x^74 - x^72 - x^71 - x^69 + x^68 + x^65 - x^63 - x^62 - x^60 - x^57 + x^56 +
 x^55 + x^52 - x^51 + x^50 - x^48 - x^47 + x^44 - x^43 + x^41 - x^40 - x^36 - x^35 +
x^33 + x^32 + x^31 - x^29 - x^26 - x^25 - x^23 + x^22 + x^20 + x^19 + x^18 - x^17 +
x^16 - x^15 - x^14 - x^13 - x^12 + x^10 - x^9 + x^8 - x^7 + x^5 - x^4 - x^3 + x^2 -
x + 1
e^2(x)=6*x^127 - 8*x^126 + 12*x^125 - 9*x^124 + 8*x^123 - 6*x^122 - 10*x^121 + 8*x^120 +
 2*x^119 - 13*x^118 + 6*x^117 - 4*x^116 + 10*x^114 + 6*x^113 - 9*x^112 - 4*x^111 - 8*x^110
- 8*x^109 + 11*x^108 + 2*x^107 - 5*x^106 + 16*x^105 - 16*x^104 - 6*x^103 + 4*x^102 - 16*x^101 +
7*x^100 + 10*x^99 - 8*x^98 + 8*x^97 + 8*x^96 - 8*x^95 + 10*x^94 + 2*x^93 + 6*x^91 - 7*x^90 -
10*x^89 + 2*x^88 - 16*x^87 + 27*x^86 + 16*x^85 - 7*x^84 + 20*x^83 - 6*x^82 - 20*x^81 - 10*x^80 -
16*x^79 + 2*x^78 + 6*x^77 + 3*x^76 + 10*x^75 + 11*x^74 + 2*x^73 + 13*x^72 + 10*x^71 - 2*x^70 -
8*x^69 - 3*x^68 - 22*x^67 + 12*x^66 - 2*x^65 + 4*x^64 + 10*x^63 + x^62 + 16*x^61 + 13*x^60 -
18*x^59 - 15*x^58 - 6*x^57 - 7*x^56 + 10*x^55 + 15*x^54 - 2*x^53 + 15*x^52 - 2*x^51 - 3*x^50 +
14*x^49 + 8*x^48 - 6*x^47 + 14*x^46 - 32*x^45 - 4*x^44 - 8*x^43 + 4*x^42 + 6*x^40 + 18*x^38 +
6*x^37 + 2*x^36 - 6*x^35 - 10*x^34 - x^32 - 16*x^31 + 14*x^30 + 10*x^29 - 6*x^28 + 14*x^27 +
11*x^26 - 14*x^25 + 25*x^24 - 14*x^23 - 11*x^22 - 10*x^21 - 4*x^20 - 2*x^19 + 23*x^18 - 8*x^17 +
8*x^16 + 8*x^15 - 16*x^14 - 2*x^13 + 22*x^12 - 16*x^11 + 2*x^10 - 6*x^9 - 2*x^8 - 6*x^7 + 17*x^6 -
4*x^5 + 15*x^4 - 16*x^3 + 6*x^2 - 6*x + 19

e^3(x)=396*x^127 - 279*x^126 - 11*x^125 + 79*x^124 - 23*x^123 - 54*x^122 + 80*x^121 - 286*x^120 +
96*x^119 + 31*x^118 - 33*x^117 + 326*x^116 + 89*x^115 - 210*x^114 + 163*x^113 - 135*x^112 - 249*x^111 +
243*x^110 - 233*x^109 - 175*x^108 + 241*x^107 - 176*x^106 + 269*x^105 + 324*x^104 - 144*x^103 + 123*x^102 +
 x^101 - 366*x^100 - 77*x^99 - 5*x^98 - 280*x^97 + 139*x^96 - 110*x^95 + 260*x^94 + 337*x^93 + 83*x^92 +
 9*x^91 - 57*x^90 - 452*x^89 + 148*x^88 - 127*x^87 - 117*x^86 + 139*x^85 - 171*x^84 + 15*x^83 + 258*x^82 -
24*x^81 + 139*x^80 + 175*x^79 - 295*x^78 - 194*x^77 + 20*x^76 - 409*x^75 + 210*x^74 + 66*x^73 - 129*x^72 +
308*x^71 + 222*x^70 - 271*x^69 + 299*x^68 - 172*x^67 - 125*x^66 + 15*x^65 - 36*x^64 - 255*x^63 + 71*x^62 -
200*x^61 + 10*x^60 + 103*x^59 + 67*x^58 + 214*x^57 + 229*x^56 - 345*x^55 + 82*x^54 - 291*x^53 - 143*x^52 +
35*x^51 - 7*x^50 + 75*x^49 + 271*x^48 - 256*x^47 + 261*x^46 - 16*x^45 - 66*x^44 + 16*x^43 + 47*x^42 -
394*x^41 + 134*x^40 - 157*x^39 - 80*x^38 + 155*x^37 - 37*x^36 + 40*x^35 + 438*x^34 - 240*x^33 + 35*x^32 -
105*x^31 - 263*x^30 - 6*x^29 + 227*x^28 - 85*x^27 + 259*x^26 - 101*x^25 - 14*x^24 - 90*x^23 + 83*x^22 -
196*x^21 + 307*x^20 - 306*x^19 + 6*x^18 + 71*x^17 - 50*x^16 + 55*x^15 + 204*x^14 - 338*x^13 + 167*x^12 -
12*x^11 - 114*x^10 + 44*x^9 + 3*x^8 - 186*x^7 + 280*x^6 + 8*x^5 + 158*x^4 + 29*x^3 - 128*x^2 - 164*x + 169
```

```
f(X)=-X^128 + X^127 + X^121 + X^120 - X^119 - X^118 - X^117 - X^116 + X^115 + X^114 - X^113 + X^111 - X^110 +
 X^107 - X^105 + X^103 + X^99 + X^96 - X^94 - X^93 + X^92 + X^91 + X^90 + X^89 + X^88 - X^86 - X^85 + X^84 +
X^83 - X^81 - X^79 - X^78 - X^75 - X^73 + X^71 - X^70 - X^69 + X^67 + X^64 + X^63 - X^62 - X^61 - X^59 - X^58 +
 X^57 + X^55 + X^54 - X^53 + X^52 + X^51 - X^50 - X^49 + X^48 + X^46 + X^42 - X^41 - X^39 + X^38 + X^37 + X^36 +
 X^35 + X^34 - X^32 + X^31 + X^30 + X^27 - X^26 + X^25 - X^23 + X^22 + X^21 + X^20 - X^19 + X^17 + X^16 - X^14 +
X^12 + X^9 - X^3 - X^2 - X - 1

e(x)=x^127 + x^126 - x^125 - x^124 + x^123 + x^122 - x^121 + x^119 - x^115 + x^113 - x^110 - x^109 + x^108 + x^105 -
 x^104 + x^103 - x^102 - x^101 - x^100 - x^98 - x^97 + x^95 + x^94 - x^93 + x^92 - x^91 - x^90 + x^87 + x^86 + x^84 -
x^83 + x^81 + x^78 - x^77 + x^74 + x^73 - x^72 - x^71 - x^70 - x^69 - x^68 - x^67 - x^66 + x^65 - x^64 + x^62 + x^61 -
x^59 + x^58 - x^57 + x^56 + x^53 - x^52 - x^51 + x^50 + x^47 + x^46 + x^45 - x^43 + x^42 + x^41 + x^40 - x^39 - x^38 +
 x^37 - x^36 + x^34 - x^32 + x^31 - x^30 - x^28 + x^27 - x^26 + x^25 - x^24 + x^20 + x^18 + x^15 - x^14 - x^13 + x^12 -
x^11 - x^10 - x^6 - x^5 - x^4 + x^2 + 1

e^2(x)=46632246162*x^127 - 18898871904*x^126 - 49562601274*x^125 - 56407630916*x^124 - 22742104862*x^123 + 44650096586*x^122 +
109564145870*x^121 + 73797285658*x^120 - 31360378766*x^119 - 64035418485*x^118 - 67319547845*x^117 - 24699911812*x^116 + 52576415938*x^115 +
 27709364714*x^114 - 25689665788*x^113 + 13559429119*x^112 + 9502017365*x^111 - 50528512331*x^110 - 1807222257*x^109 + 20043538483*x^108 + 36721813731*x^107
- 13837470066*x^106 - 26282322303*x^105 + 22368775808*x^104 + 25282887382*x^103 - 36798400741*x^102 - 48519578695*x^101 - 28256314494*x^100 + 23514417740*x^99 +
34992412114*x^98 + 76227639273*x^97 + 77895298910*x^96 - 26850666345*x^95 - 109311057170*x^94 - 99719479125*x^93 - 13177425619*x^92 + 26492858320*x^91 +
69171363455*x^90 + 83223678850*x^89 + 44143439953*x^88 - 43018695823*x^87 - 86827879064*x^86 - 37421658356*x^85 + 59790880675*x^84 + 58835263927*x^83 +
 16686134879*x^82 - 6857207250*x^81 + 10536832243*x^80 - 23269579172*x^79 - 8309243731*x^78 + 15533832933*x^77 - 12703043870*x^76 - 44444444786*x^75 -
10068338680*x^74 - 2894180676*x^73 + 56668226180*x^72 + 49181043663*x^71 - 46691331238*x^70 - 53870592802*x^69 - 27519747279*x^68 - 18807297103*x^67 -
 38058876207*x^66 + 14535594063*x^65 + 80954642663*x^64 + 69609573571*x^63 + 3770424807*x^62 + 1894101741*x^61 - 7096896270*x^60 - 66335149521*x^59 -
 55695148490*x^58 - 1669691631*x^57 - 18856863060*x^56 + 21782123170*x^55 + 18154854113*x^54 + 3510613158*x^53 + 77299312542*x^52 + 28096988219*x^51 -
 61551572483*x^50 - 48019552731*x^49 + 4358663340*x^48 - 19899330928*x^47 + 11148107327*x^46 + 1653884427*x^45 + 37835534594*x^44 + 61898127714*x^43 +
49355741014*x^42 - 54192328563*x^41 - 66988162862*x^40 - 85516919055*x^39 - 31557193*x^38 + 27873629853*x^37 + 52082505244*x^36 + 52077395858*x^35 +
16347848131*x^34 - 44372359630*x^33 - 48182556576*x^32 + 30161401486*x^31 + 24189554550*x^30 + 1339917119*x^29 + 17989003036*x^28 + 28550087041*x^27 -
24738948520*x^26 + 13836031912*x^25 - 38185204943*x^24 - 33979790782*x^23 + 45142110920*x^22 + 28645419424*x^21 - 13263306058*x^20 - 60124020330*x^19 +
15413029687*x^18 + 63413785029*x^17 + 38314725749*x^16 - 23735440860*x^15 - 45501648511*x^14 + 4277235253*x^13 + 22893974248*x^12 - 14123723263*x^11 -
6386434996*x^10 + 9972376824*x^9 - 20253215198*x^8 + 2397677478*x^7 + 34116149536*x^6 + 54642003260*x^5 + 42731763673*x^4 - 7820019089*x^3 - 29549588824*x^2 -
43378676795*x - 34858383112
```

Computation done in SageMath

Popular choice of $\mathrm{R_q} = Z_q[x]/(f)$

The hardness assumption holds for any q and meaningful f

For practical purpose:

- $f = x^{2^k} + 1$ and any q that factors f in small degree factors (e.g. Linear factors when $q = 1 \; mod \; 2n$).
- Fast NTT operation

Evaluation attack on RLWE for $f(x) = x^n - 1$

Proof sketch

- $f(1) = 0 \; mod \; q$
- Let $s(x), e_i(x) \leftarrow \chi$
- Let $(a_i(x), b_i(x) = a_i(x)s(x) + e_i(x)) \in R_q = Z_q[x]/(f)$ be a RLWE sample
- Evaluate $a_i(1), b_i(1) \in Z_q$
- Now $b_i(1) = a_i(x)s(x)_{x=1} + e_i(1) \; mod \; q$
- $= a_i(1)s(1) + e_i(1) \; mod \; q$ [since $f(1) = 0 mod \; q$]
- Then $b_i(1) - a_i(1)s(1) = e_i(1)$

Check the right $s(1)$ from the support of χ.

If such $s(1)$ exits, you will get small $e_i(1)$

Digital Signatures

Key Generation Algorithm→ (Pub,Sec) = Gen(k)

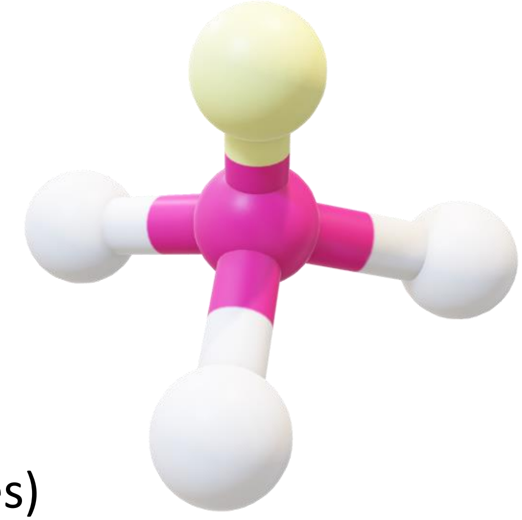
Signing Algorithm → S=Sign(Sec,M)

Verification Algorithm→ Verify(S,M,Pub)= Yes/No

Digital Signatures

- Correctness: Verify(Pub,M,Sign(Sec,M))= Yes
- Security: Unforgeability

Lattice-based signatures



- Trapdoor-based signatures[GPV'08]

(Pros: Compact signatures, Cons: Gaussian sampling over lattices)

- Fiat-Shamir transformation from ID schemes (like Schnorr protocol)[Lyu'09,12,BG'14,…]

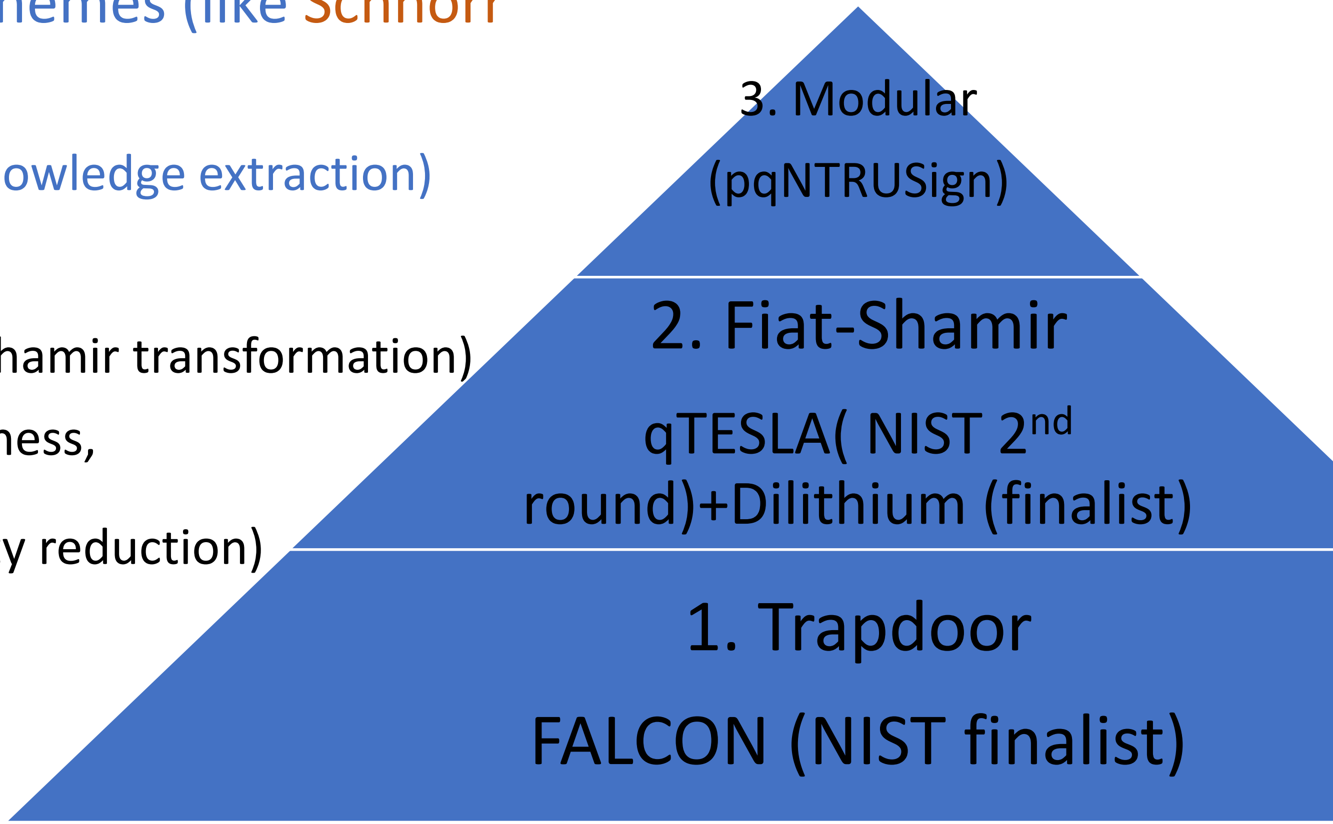
(Pros: Fast, Cons: Rejection Sampling & exact knowledge extraction)

- Modular lattice signatures [**D**HP+20]

(successor of NTRUSign: Trapdoor-based+Fiat-Shamir transformation)

(Pros: Tradeoff between compactness and fastness,

Cons: Rejection sampling, Unforgeability security reduction)



3- round ID schemes

Prover (sk)

Commit

$w \leftarrow P_1\ (sk)$

Verifier (pk)

Challenge

$c \in C \leftarrow P_2(w)$

Response

$z \leftarrow P_3(w, c, sk)$

Accept/Reject

Properties

- Correctness

- Honest Verifier Zero Knowledge (HVZK): A simulator can produce the transcript (w, c, z) using only pk with same distribution as in the real protocol.

No information of sk is leaked

- Special Soundness (SS): A verifier can extract the knowledge sk using a prover who wins the protocol in two different runs on the same commitment (rewinding technique).

Prover indeed holds sk

ID to Digital Signatures (Fiat-Shamir Transformation)

Signer (sk, M) Verifier (pk)

$w \leftarrow P_1(sk)$

$c = H(w, M)$

$z \leftarrow P_3(sk, c, M)$ $\longrightarrow$ Accept/Reject

Theorem: If the ID is HVZK+SS $\Rightarrow$ Signature scheme is UF-CMA secure in (Q)ROM.

Schnorr protocol (using discrete log)

Prover sk: (g, s)

Verifier pk: $(g, g^s = h)$

$y \leftarrow Z_q$

$w = g^y$

$c \leftarrow Z_q$

$z = sc + y$

Accept if $g^z = h^c w$

Correctness: $g^z = g^{sc+y} = h^c w$

Schnorr protocol

- HVZK:

$c \leftarrow Z_q, z \leftarrow Z_q$ and set $w = g^z / h^c$

(w, c, z) has the original distribution as in the original protocol.

- SS:

Let $(w, c, z), (w, c', z')$ be two valid transcript from the prover.

$g^z = h^c w$, $g^{z'} = h^{c'} w$, then $\frac{g^z}{h^c} = \frac{g^{z'}}{h^{c'}}$,

Hence $g^{s'(=\frac{z-z'}{c-c'})} = h$

Lattice Analogue of Schnorr Protocol

Prover sk: $(A \in R_q^{n\times m}, s)$

Verifier pk: $(A, As = t)$

$y \leftarrow R_q^m$

$w = Ay$ $\longrightarrow$

$\longleftarrow$ $c \leftarrow R_q$

$z = sc + y$ $\longrightarrow$

Accept if $\mathrm{Az} - \mathrm{tc} = w$

Correctness: $A(sc + y) = tc + w$

Lattice Analogue of Schnorr Protocol

Prover sk: $(A \in R_q^{n\times m}, s)$ Verifier pk: $(A, As = t)$

$y \leftarrow R_q^m$

$w = Ay$ $\longrightarrow$

$\longleftarrow$ $c \leftarrow R_q$

$z = sc + y$ $\longrightarrow$

Accept if $\mathrm{Az} - \mathrm{tc} = w$

Challenges:

- If z is not small, forging z is easy.
- Sample small y, c & add smallness condition in the Verification step.
- $z = sc + y$ leaks information about secret s (learning parallelepiped type attacks)

Additional care is required!!

Lattice Analogue of Schnorr Protocol

Prover sk: $(A \in R_q^{n \times m}, s)$

Verifier pk: $(A, As = t)$

$y \leftarrow R_q^m$

$w = Ay$ $\longrightarrow$

$\longleftarrow$ $c \leftarrow R_q$

$z = sc + y$ $\longrightarrow$

Accept if z is small and $\mathrm{A}\mathrm{z} - \mathrm{tc} = w$

Challenges:

- At high level, we want the distribution of $z \approx_s$ some distribution independent of s
- Use rejection sampling to achieve it.

Rejection sampling [Lyu'12]

- Let $\boldsymbol{f}, \boldsymbol{g}$ be two distributions such that $\boldsymbol{f}(x) \leq M\ \boldsymbol{g}(x)$ for "almost" all x

Suppose we have access to $\boldsymbol{g}$(depends on s), but we want to output according to $\boldsymbol{f}$ (independent of s)

- $z \leftarrow \boldsymbol{g}$ and output with probability $\frac{\boldsymbol{f}(z)}{M\boldsymbol{g}(z)} \approx_s z \leftarrow \boldsymbol{f}$ and output with probability $1/M$

We can aim for the distribution of $\boldsymbol{f}$ as

- Uniform distribution in a small interval [Lyu'09]
- Discrete Gaussian distribution [Lyu'12]
- Bimodal Gaussian distribution [DDLL'13]

Lattice Analogue of Schnorr Protocol

Prover sk: $(A \in R_q^{n\times m}, s)$

Verifier pk: $(A, As = t)$

$y \leftarrow R_q^m$

$w = Ay$

$c \leftarrow R_q$

$z = sc + y$

Apply Rejection sampling

& Re-run (if required)

Accept if z is small and $\mathrm{A}z - \mathrm{tc} = w$

Expected number of re-run: M times

Lattice Analogue of Schnorr Protocol

- HVZK

Sample small c, z and make $w = Az - tc$ and output the transcript with probability $1/M$.

The distribution of (w, c, z) is identical to the original protocol.

Lattice Analogue of Schnorr Protocol

- SS

$\mathrm{A}z - \mathrm{tc} = w = Az' - tc'$

Then $A(z - z') = t(c - c')$

We can chose q such that $c - c'$ is invertible in R_q

$\mathrm{A}\frac{(z-z')}{c-c\prime} = t$, but…

$\frac{(z-z')}{c-c\prime}$ is not small anymore.

So we couldn’t Extract small s' such that $As' = t$

Lattice Analogue of Schnorr Protocol

- Still a meaningful extraction.

$A(z - z') = t(c - c')$

Put $t = As$

$$A\big((z - z') - s(c - c')\big) = 0$$

This is a solution of the SIS problem.

Quotient of small elements in $R_q = Z_q[x]/(f)$

```
f(x)=x^128+1
q=32771

a(x)=x^127 + x^126 + x^125 + x^123 + 32770*x^122 + 32770*x^121 + x^119 + 32770*x^118 + 32770*x^116 + x^115 + 32770*x^114 + x^113 + 32770*x^112 +
32770*x^111 + 32770*x^110 + x^109 + x^107 + 32770*x^104 + 32770*x^103 + 32770*x^101 + x^100 + 32770*x^98 + x^96 + 32770*x^95 + x^94 + x^93 + x^92 +
x^91 + 32770*x^90 + x^89 + x^88 + x^87 + x^85 + x^84 + x^83 + x^79 + x^78 + 32770*x^77 + 32770*x^76 + 32770*x^75 + x^73 + x^70 + 32770*x^69 + 32770*x^68 +
x^67 + x^66 + 32770*x^65 + 32770*x^63 + 32770*x^62 + 32770*x^60 + 32770*x^59 + 32770*x^57 + 32770*x^55 + 32770*x^54 + 32770*x^53 + x^50 + 32770*x^49 +
32770*x^48 + x^47 + 32770*x^46 + 32770*x^45 + x^44 + x^43 + 32770*x^42 + 32770*x^41 + 32770*x^40 + 32770*x^39 + 32770*x^36 + 32770*x^34 + 32770*x^33 +
x^32 + 32770*x^29 + x^26 + 32770*x^24 + x^23 + x^22 + x^21 + 32770*x^20 + x^15 + 32770*x^13 + x^12 + 32770*x^11 + 32770*x^10 + x^8 + x^7 + 32770*x^6 +
x^5 + 32770*x^4 + 32770*x^3 + 32770*x^2 + 32770

b(x)=32770*x^127 + x^125 + x^124 + 32770*x^123 + 32770*x^122 + 32770*x^121 + 32770*x^120 + 32770*x^119 + x^116 + 32770*x^115 + 32770*x^114 + 32770*x^113 +
 32770*x^112 + 32770*x^110 + x^109 + 32770*x^106 + 32770*x^105 + x^102 + 32770*x^101 + 32770*x^100 + 32770*x^98 + x^97 + x^96 + x^95 + 32770*x^91 + 32770*x^90 +
 32770*x^89 + x^88 + 32770*x^87 + x^86 + x^85 + x^84 + 32770*x^83 + x^82 + 32770*x^81 + 32770*x^80 + x^79 + x^76 + x^74 + 32770*x^73 + 32770*x^72 + 32770*x^71 +
x^70 + x^69 + 32770*x^68 + 32770*x^66 + 32770*x^64 + 32770*x^63 + x^62 + 32770*x^61 + 32770*x^60 + 32770*x^59 + 32770*x^56 + 32770*x^54 + x^53 + x^51 + 32770*x^48 +
 x^47 + x^46 + x^44 + 32770*x^41 + x^40 + x^38 + x^36 + x^35 + 32770*x^33 + 32770*x^32 + x^27 + x^26 + 32770*x^25 + 32770*x^21 + x^20 + x^18 + x^17 + x^16 + x^15 +
32770*x^14 + x^13 + x^12 + x^11 + 32770*x^10 + x^6 + 32770*x^5 + x^4 + 32770*x^2 + x + 32770

a(x)/b(x)=25890*x^127 + 4597*x^126 + 17063*x^125 + 23762*x^124 + 22492*x^123 + 6247*x^122 + 22526*x^121 + 22963*x^120 + 18046*x^119 + 1376*x^118 + 32123*x^117 +
30559*x^116 + 1342*x^115 + 22769*x^114 + 32767*x^113 + 21477*x^112 + 17226*x^111 + 4687*x^110 + 13623*x^109 + 11901*x^108 + 7292*x^107 + 31694*x^106 + 15593*x^105 +
3025*x^104 + 15518*x^103 + 23889*x^102 + 27148*x^101 + 4607*x^100 + 8485*x^99 + 30044*x^98 + 29788*x^97 + 30406*x^96 + 8870*x^95 + 8665*x^94 + 32301*x^93 + 17070*x^92 +
22749*x^91 + 10346*x^90 + 31477*x^89 + 20225*x^88 + 22687*x^87 + 17007*x^86 + 22075*x^85 + 22892*x^84 + 29728*x^83 + 31327*x^82 + 354*x^81 + 908*x^80 + 14965*x^79 +
11289*x^78 + 1513*x^77 + 27035*x^76 + 12816*x^75 + 14768*x^74 + 1680*x^73 + 18875*x^72 + 17602*x^71 + 25220*x^70 + 1819*x^69 + 15900*x^68 + 25915*x^67 + 31731*x^66 +
 21266*x^65 + 26048*x^64 + 28131*x^63 + 31734*x^62 + 29460*x^61 + 21226*x^60 + 9652*x^59 + 32446*x^58 + 15884*x^57 + 24280*x^56 + 13287*x^55 + 31702*x^54 + 29256*x^53 +
 26124*x^52 + 24267*x^51 + 11764*x^50 + 9689*x^49 + 3806*x^48 + 12617*x^47 + 611*x^46 + 13251*x^45 + 6273*x^44 + 25829*x^43 + 32342*x^42 + 20197*x^41 + 22019*x^40 +
19593*x^39 + 24284*x^38 + 17893*x^37 + 10664*x^36 + 3381*x^35 + 7943*x^34 + 11733*x^33 + 17210*x^32 + 6763*x^31 + 10411*x^30 + 21797*x^29 + 10748*x^28 + 23081*x^27 +
6255*x^26 + 2333*x^25 + 3759*x^24 + 19664*x^23 + 4827*x^22 + 22681*x^21 + 7112*x^20 + 9816*x^19 + 27028*x^18 + 7906*x^17 + 21108*x^16 + 19800*x^15 + 30792*x^14 +
 14339*x^13 + 3018*x^12 + 26773*x^11 + 29410*x^10 + 10146*x^9 + 13327*x^8 + 32548*x^7 + 27105*x^6 + 4952*x^5 + 16658*x^4 + 19916*x^3 + 21174*x^2 + 28148*x + 17163
```

Computation done in SageMath

Open Problems

- Special Soundness property (important for other applications, like proof of proper cipher-text)
- Eliminate/better understanding of the rejection sampling technique
- Rejection sampling in other metric (e.g. Hamming metric)
- Tight security reductions from search lattice problems
- Lower bound the success probability of small invertible elements in some R_q