The Learning With Errors Problem

Oded Regev
Tel Aviv University

(for more details, see survey prepared for CCC'2010)

Paris, 2010/5/29

Organization

Learning With Errors (LWE) Problem

- A secret vector **s** in $\mathbb{Z}_{17}^4$
- We are given an arbitrary number of equations, each correct up to ± 1
- Can you find **s**?

$$
\begin{aligned}
14s_1 + 15s_2 + 5s_3 + 2s_4 &\approx 8 \pmod{17} \\
13s_1 + 14s_2 + 14s_3 + 6s_4 &\approx 16 \pmod{17} \\
6s_1 + 10s_2 + 13s_3 + 1s_4 &\approx 3 \pmod{17} \\
10s_1 + 4s_2 + 12s_3 + 16s_4 &\approx 12 \pmod{17} \\
9s_1 + 5s_2 + 9s_3 + 6s_4 &\approx 9 \pmod{17} \\
3s_1 + 6s_2 + 4s_3 + 5s_4 &\approx 16 \pmod{17} \\
6s_1 + 7s_2 + 16s_3 + 2s_4 &\approx 3 \pmod{17}
\end{aligned}
$$

LWE's Claim to Fame

- ✓ Known to be as hard as worst-case lattice problems, which are believed to be exponentially hard (even against quantum computers)
- ✓ Extremely versatile
- ✓ Basis for provably secure and efficient cryptographic constructions

LWE's Origins

- The problem was first defined in [R05]
- Already (very) implicit in the first work on lattice-based public key cryptography [AjtaiDwork97] (and slightly more explicit in [R03])
  - See the survey paper for more details

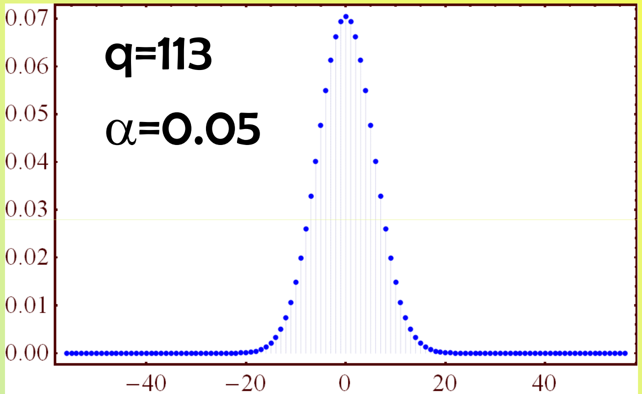
LWE – More Precisely

- There is a secret vector s in $\mathbb{Z}_q^n$
- An oracle (who knows s) generates a uniform vector a in $\mathbb{Z}_q^n$ and noise $e \in \mathbb{Z}$ distributed normally with standard deviation αq.
- The oracle outputs (a, $b = \langle a,s \rangle + e \bmod q$)
- This procedure is repeated with the same s and fresh a and e
- Our task is to find s

$$\begin{pmatrix} 2 & 13 & 7 & 3 \\ 4 & 7 & 9 & 1 \\ 6 & 14 & 5 & 11 \end{pmatrix} \cdot \begin{pmatrix} 8 \\ 3 \\ 12 \\ 5 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 13 \\ 12 \\ 3 \end{pmatrix}$$

LWE – Parameters: n, q, α

- The main parameter is n, the dimension
- The modulus q is typically poly(n)
  - Choosing exponential q increases size of input and makes applications much less efficient (but hardness is somewhat better understood)
  - (The case q=2 is known as Learning Parity with Noise (LPN))
- The noise element e is chosen from a normal distribution with standard deviation αq:



- The security proof requires $\alpha q > \sqrt{n}$
- The noise parameter α is typically 1/poly(n)
- The number of equations does not really matter

Algorithms

Algorithm 1: More Luck Than Sense

- Ask for equations until seeing several "$s_1 \approx \ldots$". E.g.,

$$\vdots$$

$$1s_1 + 0s_2 + 0s_3 + 0s_4 \approx 8 \pmod{17}$$

$$\vdots$$

$$1s_1 + 0s_2 + 0s_3 + 0s_4 \approx 7 \pmod{17}$$

$$\vdots$$

$$1s_1 + 0s_2 + 0s_3 + 0s_4 \approx 8 \pmod{17}$$

$$\vdots$$

- This allows us to deduce s_1 and we can do the same for the other coordinates
- Running time and number of equations is $2^{O(n\log n)}$

Algorithm 2: Maximum Likelihood

- Easy to show: After about O(n) equations, the secret s is the only assignment that approximately satisfies the equations (hence LWE is well defined)
- Hence we can find s by trying all possible q^n assignments
- We obtain an algorithm with running time $q^n=2^{O(n\log n)}$ using only O(n) equations

Algorithm 3: [BlumKalaiWasserman'03]

- Running time and number of equations is $2^{O(n)}$
- Best known algorithm for LWE (with usual setting of parameters)
- Idea:
  - First, find a small set S of equations (say, |S|=n) such that $\Sigma_S a_i=(1,0,\ldots,0)$. Do this by partitioning the n coordinates into logn blocks of size n/logn and construct S recursively by finding collisions in blocks
  - The sum of these equations gives a guess for s_1 that is quite good

Algorithm 4: [AroraGe'10]

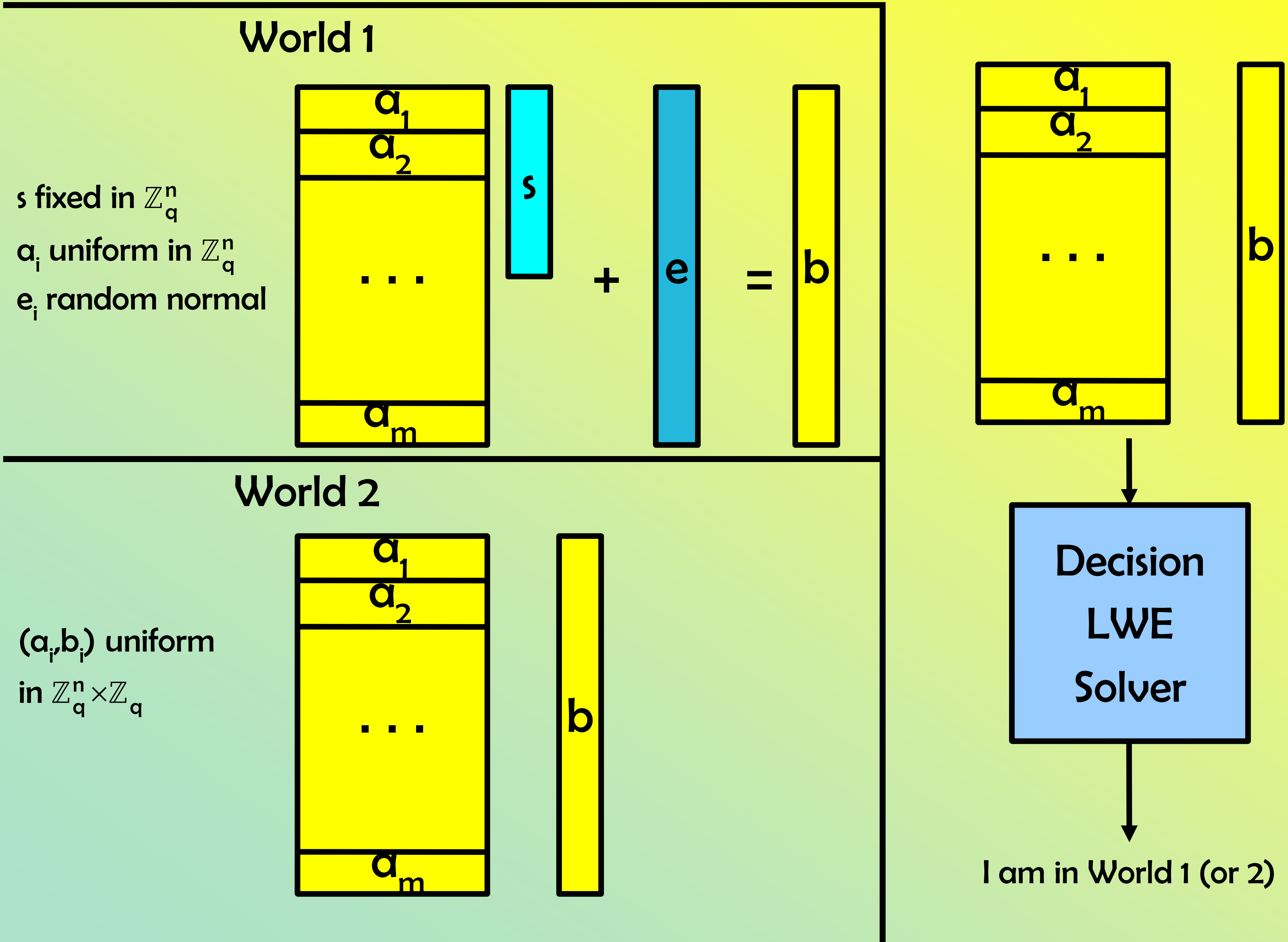
- Running time and number of equations is $2^{O((\alpha q)^2)}$
- So for $\alpha q<\sqrt{n}$, this gives a sub-exponential algorithm
- Interestingly, the LWE hardness proof [R05] requires $\alpha q>\sqrt{n}$; only now we 'know' why!
- Idea: apply a polynomial that zeroes the noise, and solve by linearization

Versatility

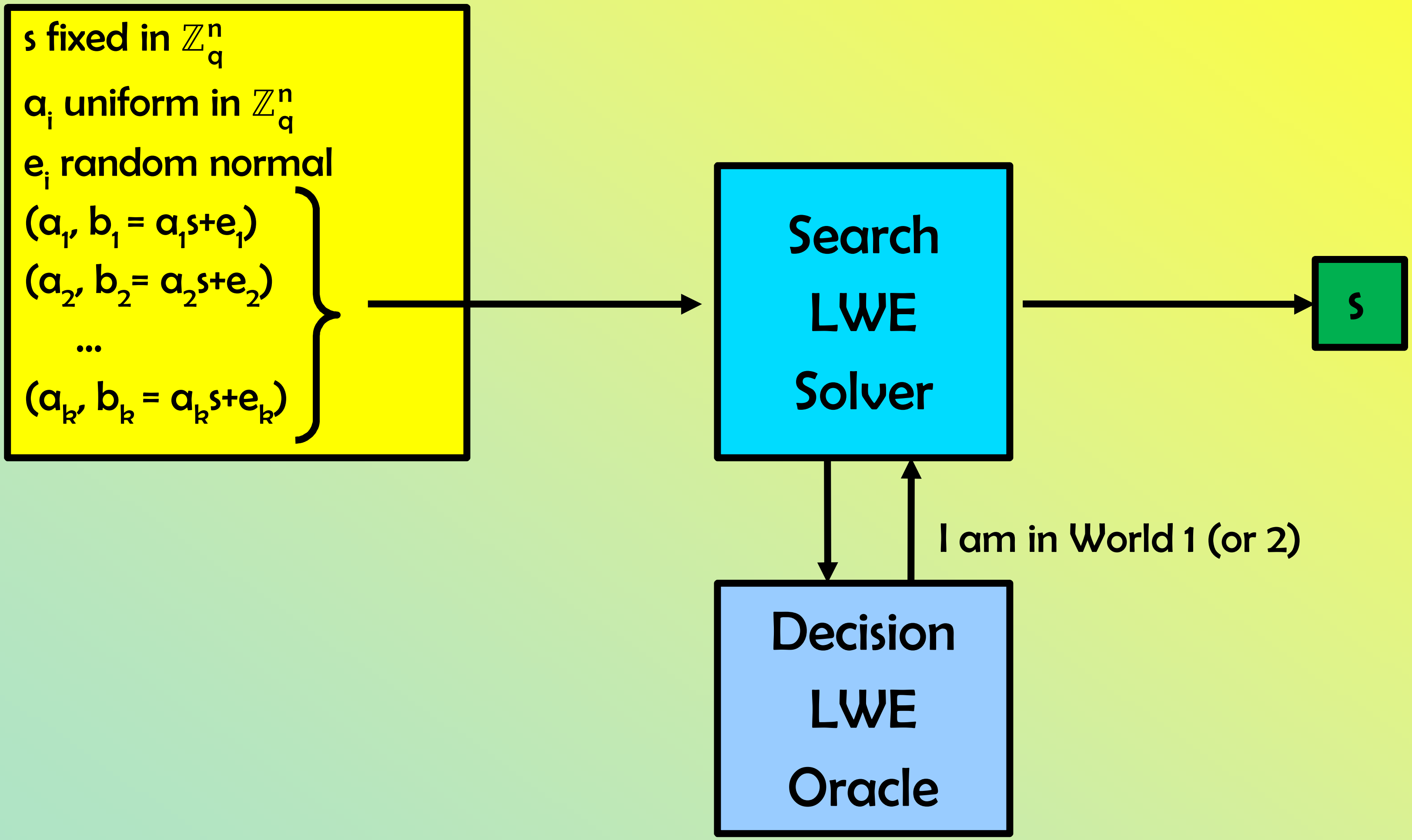
LWE is Versatile

- ✓ Search to decision reduction
- ✓ Worst-case to average-case reduction (i.e., secret can be uniformly chosen)
- The secret can be chosen from a normal distribution itself [ApplebaumCashPeikertSahai09], or from a weak random source [GoldwasserKalaiPeikertVaikuntanathan10]
- The normal error distribution is ‘LWE complete’
- The number of samples does not matter

Decision LWE Problem

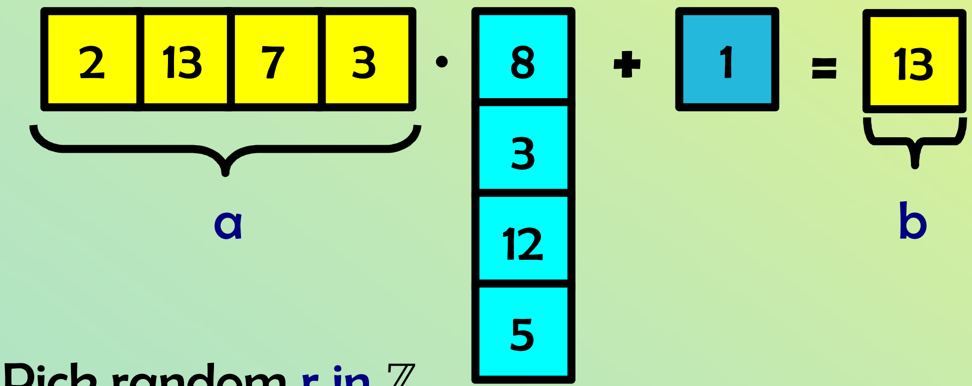


What We Want to Construct



Search LWE < Decision LWE

- Idea: Use the Decision oracle to figure out the coordinates of s one at a time
- Let g $\in \mathbb{Z}_q$ be our guess for the first coordinate of s
- Repeat the following:
  - Receive LWE pair (a,b)



  - Pick random r in $\mathbb{Z}_q$
  - Send (a+(r,0,...,0), b+rg) to the decision oracle:

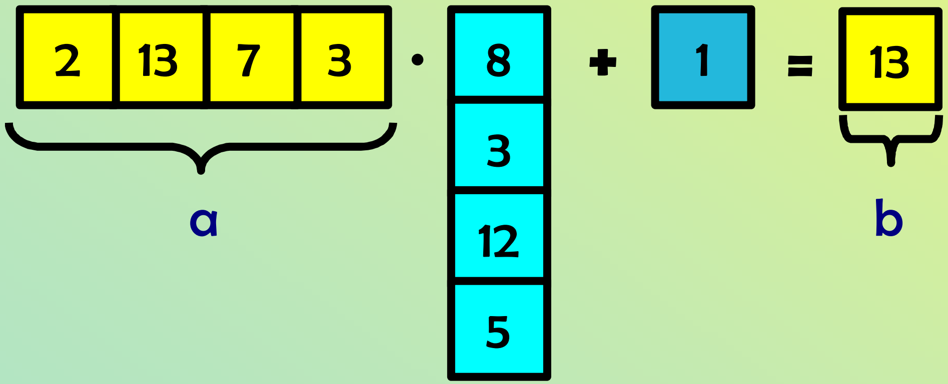
1. If g is right, then we are sending a distribution from World 1
2. If g is wrong, then we are sending a distribution from World 2 (here we use that q is prime)

- We will find the right g after at most q attempts
- Use the same idea to recover all coefficients of s one at a time

Worst Case to Average Case

- We are given an oracle that distinguishes World 1 from World 2 for a non-negligible fraction of secrets $s \in \mathbb{Z}_q^n$
- Our goal is to distinguish the two worlds for *all* secrets s

- Choose $t \in \mathbb{Z}_q^n$ uniformly
- Repeat the following:
  - Receive LWE pair (a,b)



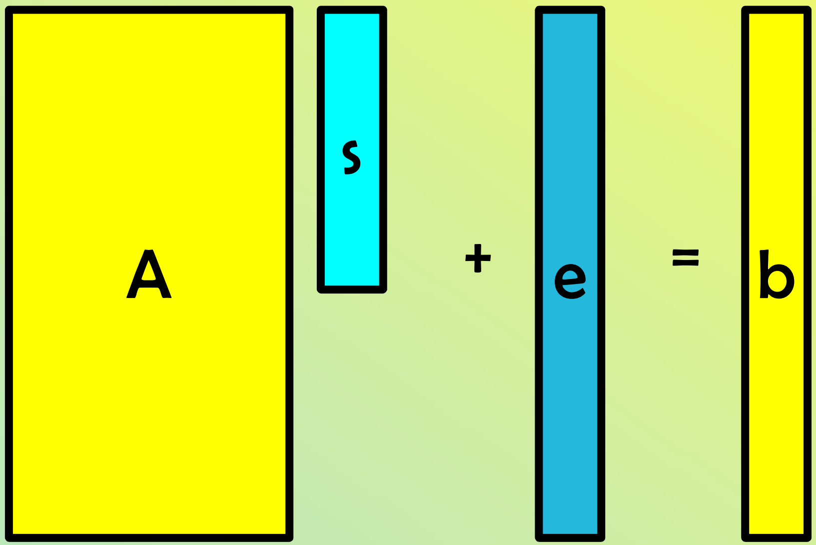
  - Send sample (a,b+⟨a,t⟩) to the oracle:

1. If our input is from World 1 with secret s, then our output is from World 1 with secret s+t
2. If out input is from World 2 then our output is also from World 2

- Since s+t is uniform in $\mathbb{Z}_q^n$, we will distinguish the two cases with non-negligible probability (over t)

Simple Cryptosystem

Public Key Encryption Based on LWE

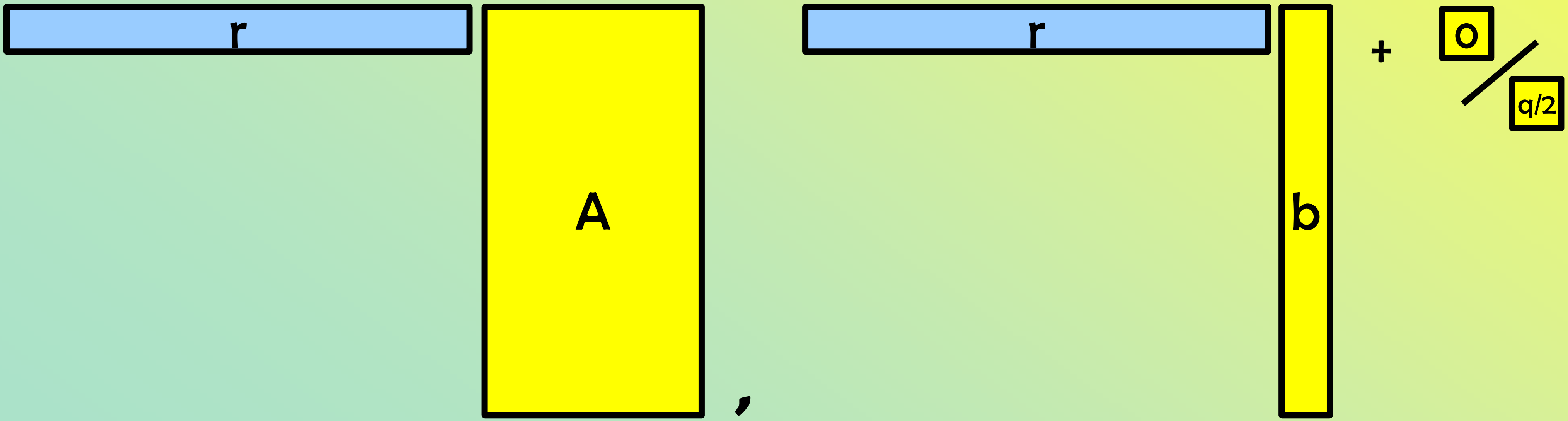


Secret Key: s in $\mathbb{Z}_q^n$

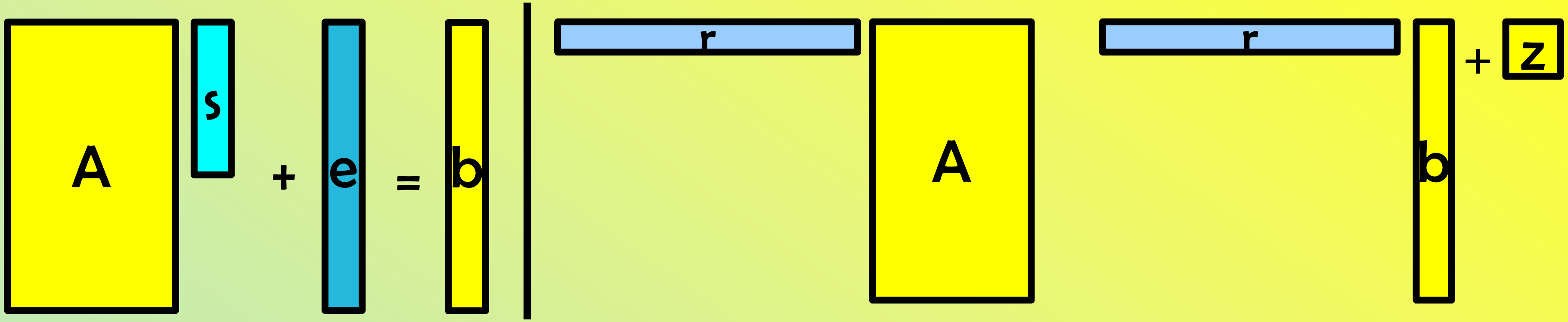
Public Key: A in $\mathbb{Z}_q^{m\times n}$, b=As+e

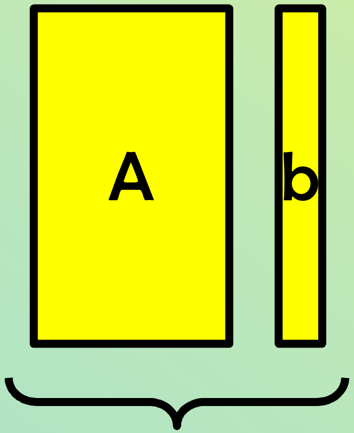
(where m=2n·logq)

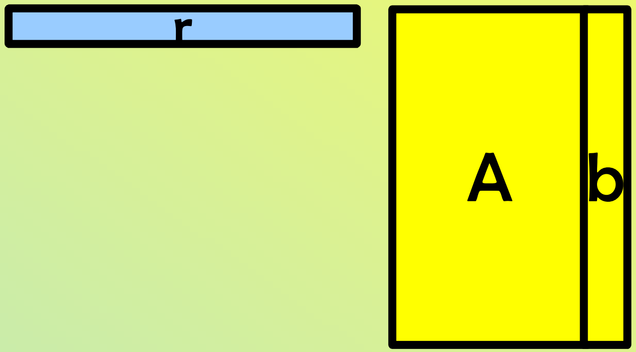
To encrypt a single bit $z\in\{0,1\}$: Pick r in $\{0,1\}^m$ and send $(rA,\ r\cdot b+z\cdot q/2)$



Proof of Semantic Security







1. The public key is pseudo-random: based on LWE

2. If A,b is truly random, then the distribution of (rA, r·b) (over r chosen from $\{0,1\}^m$) is statistically extremely close to uniform so decryption is impossible

Other Applications

- Public Key Encryption [R05, KawachiTanakaXagawa07, PeikertVaikuntanathanWaters08]
- CCA-Secure PKE [PeikertWaters08, Peikert09]
- Identity-Based Encryption [GentryPeikertVaikuntanathan08]
- Oblivious Transfer [PeikertVaikuntanathanWaters08]
- Circular-Secure Encryption [ApplebaumCashPeikertSahai09]
- Leakage Resilient Encryption [AkaviaGoldwasserVaikunathan09, DodisGoldwasserKalaiPeikertVaikuntanathan10, GoldwasserKalaiPeikertVaikuntanathan10]
- Hierarchical Identity-Based Encryption [CashHofheinzKiltzPeikert09, AgrawalBonehBoyen09]
- Learning Theory [KlivansSherstov06]
- And more…

Hardness

Hardness

- The best known algorithms run in exponential time
  - Even quantum algorithms don't do any better
- LWE is an extension of LPN, a central problem in learning theory and coding theory (decoding from random linear codes)

Hardness

- More importantly, LWE is as hard as worst-case lattice problems [R05, Peikert09]
- More precisely,
  - For $q=2^{O(n)}$, as hard as GapSVP [Peikert09]
  - For q=poly(n),
    - As hard as GapSVP given a somewhat short basis [Peikert09]
    - As hard as GapSVP and SIVP using a quantum reduction [R05]

The SIS problem

- The "Small Integer Solution" problem is a 'dual' problem to LWE:
  - Given a_1,a_2,... uniformly chosen from $\mathbb{Z}_q^n$, find a subset of them that sums to zero
- SIS is used for 'minicrypt' constructions, such as:
  - One-way functions [Ajtai96]
  - Collision resistant hash functions [GoldreichGoldwasserHalevi96]
  - Digital signatures [GentryPeikertVaikuntanathan'08, CashHofheinzKiltzPeikert09]
  - Identification schemes [MicciancioVadhan03, Lyubashevsky08, KawachiTanakaXagawa08]
- The hardness of SIS is well understood [MicciancioR04]:
  - For any q>poly(n) solving SIS implies a solution to standard lattice problems such as SIVP and GapSVP

Hardness of LWE

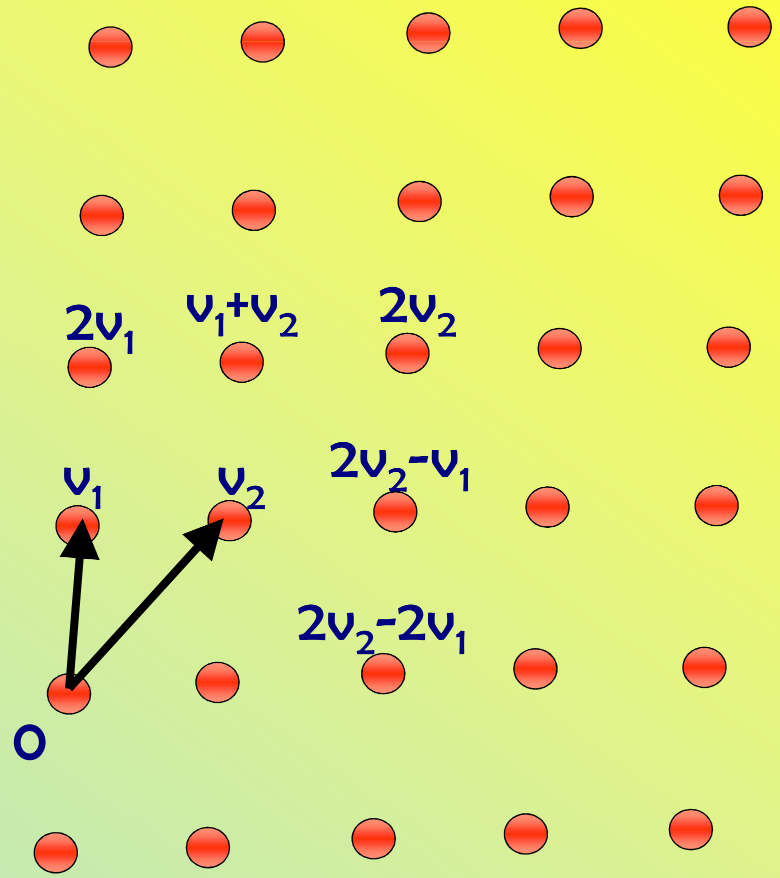
- We will present the hardness results of LWE [R05, Peikert09] including simplifications due to [LyubashevskyMicciancio09]
- Recently, [StehléSteinfeldTanakaXagawa09] gave an interesting alternative hardness proof by a (quantum) reduction from the SIS problem
  - Unfortunately leads to qualitatively weaker results
  - We will not describe it here

Lattices

- For vectors $v_1,\ldots,v_n$ in $\mathbb{R}^n$ we define the lattice generated by them as

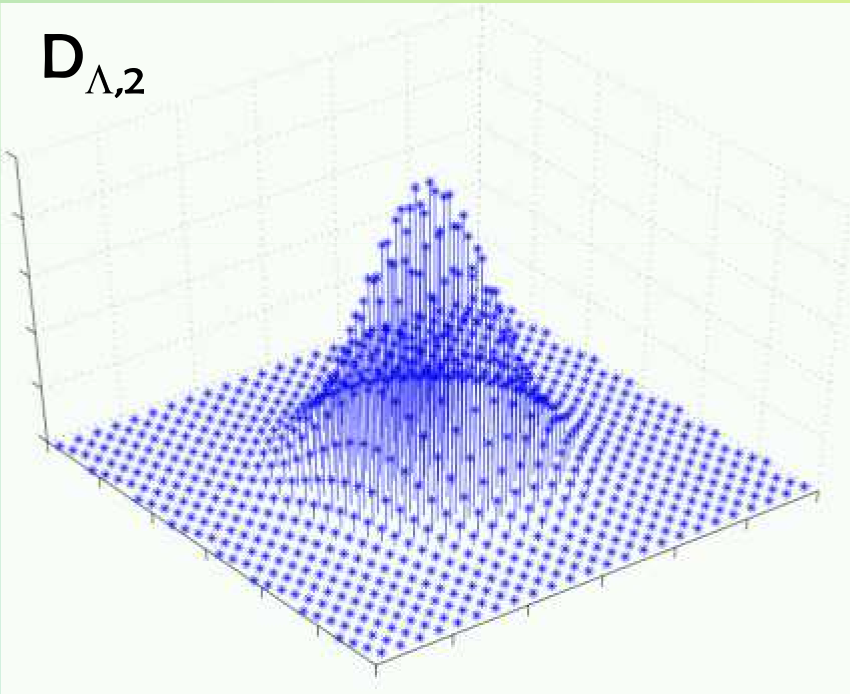
  $\Lambda=\{a_1v_1+\ldots+a_nv_n \mid a_i \text{ integers}\}$
- We call $v_1,\ldots,v_n$ a basis of Λ
- The dual lattice of Λ is

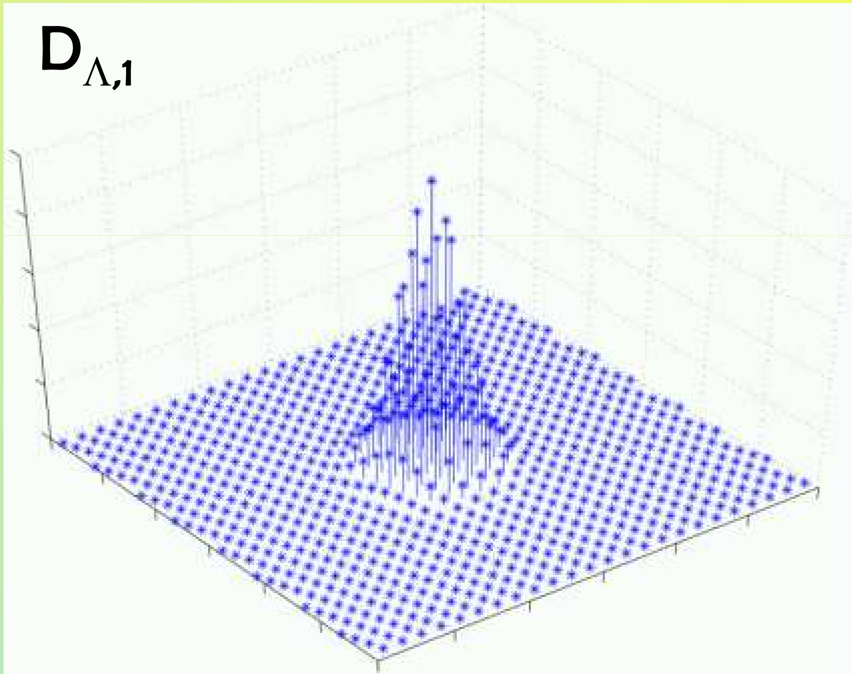
  $\Lambda^* = \{ x\in\mathbb{R}^n \mid \forall\, y\in\Lambda,\ \langle x,y\rangle \in \mathbb{Z} \}$
- For instance, $(\mathbb{Z}^n)^* = \mathbb{Z}^n$



Discrete Gaussian Distribution

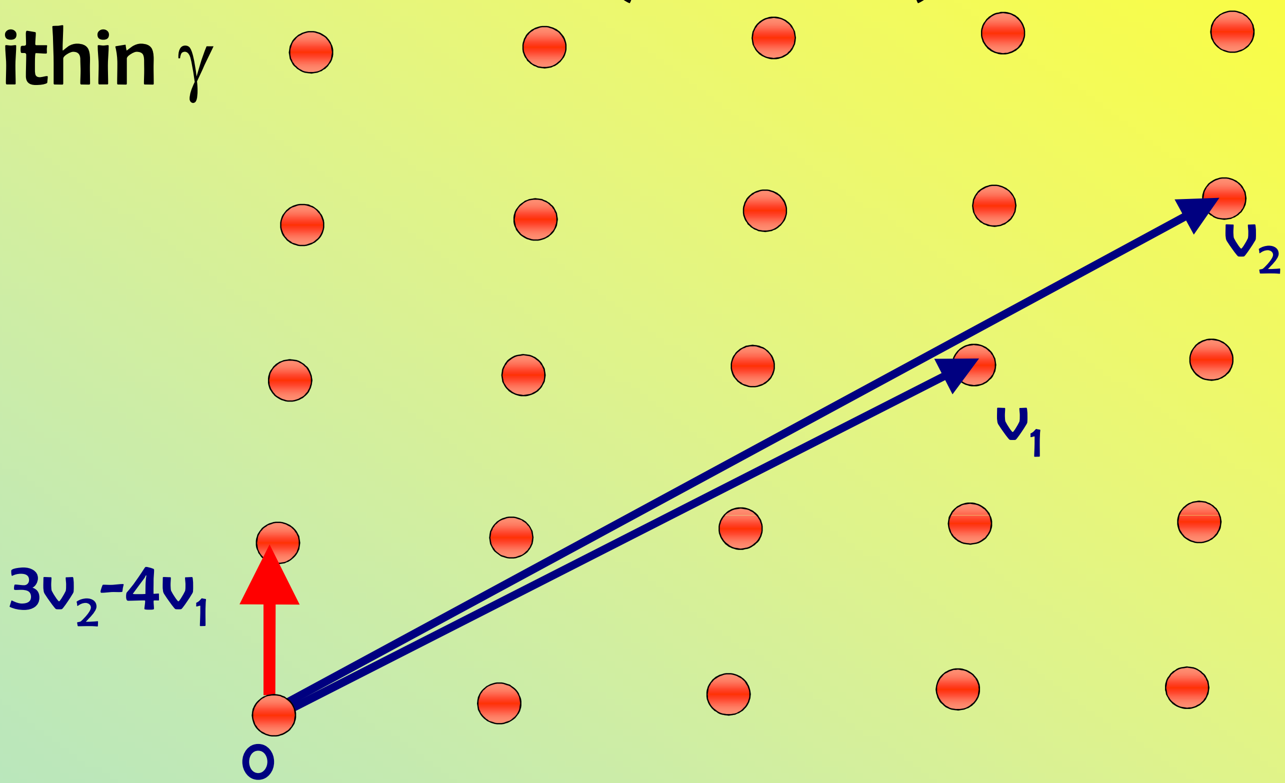
- For r>0, the distribution $D_{\Lambda,r}$ assigns mass proportional to $e^{-||x/r||^2}$ to each point $x \in \Lambda$
- Points sampled from $D_{\Lambda,r}$ are lattice vectors of norm roughly $r\sqrt{n}$





Computational Problems on Lattices

- 'Algebraic' lattice problems are easy; 'geometric' problems are hard
- Shortest Vector Problem (GapSVP_γ): given a lattice Λ, approximate length of shortest (nonzero) vector $\lambda_1(\Lambda)$ to within γ

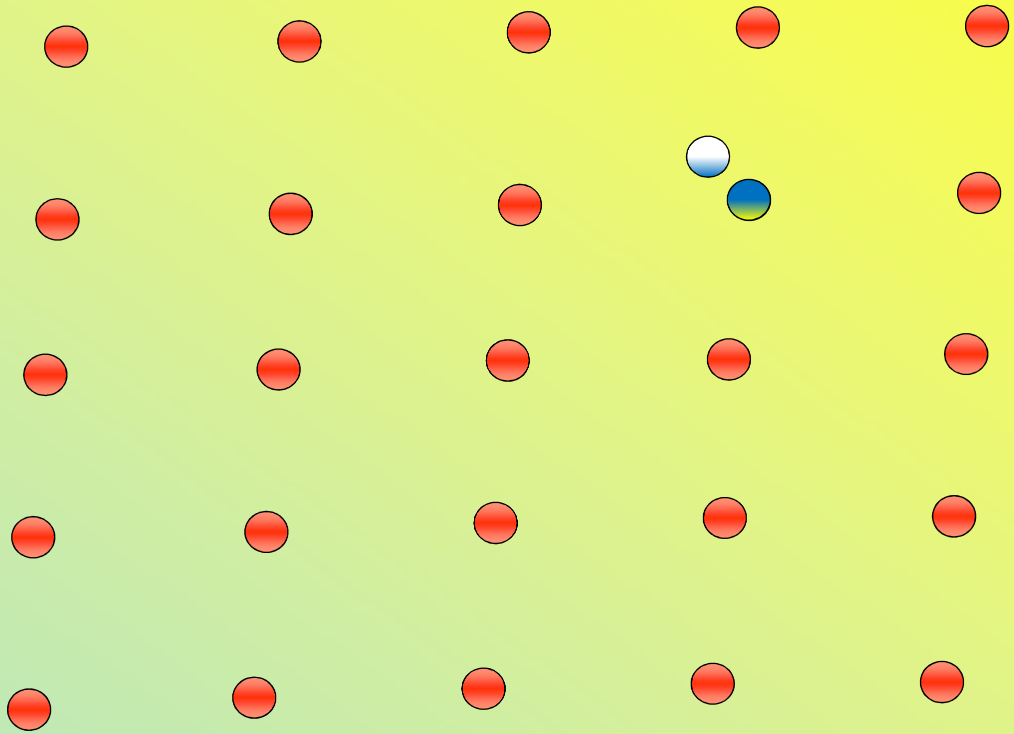


- Another lattice problem: SIVP_γ. Asks to find n short linearly independent lattice vectors.

Lattice Problems Are Hard

- Conjecture: for any γ=poly(n), $GapSVP_{\gamma}$ is hard
  - Best known algorithms run in time 2^n [AjtaiKumarSivakumar01, MicciancioVoulgaris10]
  - Quantum computation doesn't seem to help
  - On the other hand, not believed to be NP-hard [GoldreichGoldwasser00, AharonovR04]

Bounded Distance Decoding (BDD)



- BDD_d: given a lattice Λ and a point x within distance d of Λ, find the nearest lattice point

Solving BDD using Gaussian Samples

- The following was shown in [AharonovR04, LiuLyubashevskyMicciancio06]:

- <u>Proposition:</u>
  - Assume we have a polynomial number of samples from $D_{\Lambda^*,r}$ for some lattice Λ and a not too small r>0.
  - Then we can solve BDD on Λ to within distance 1/r

Core LWE Hardness Statement

- The core of the LWE hardness result is the following:
- Proposition [R05]:
  - Assume we have access to an oracle that solves LWE with modulus q and error parameter α.
  - Assume we also have a polynomial number of samples from $D_{\Lambda^*,r}$ for some lattice Λ and a not too small r>0.
  - Then we can solve BDD on Λ to within distance $\alpha q/r$
- This is already some kind of hardness result: without the LWE oracle, the best known algorithms for solving the above task require exponential time, assuming $\alpha q \geq \sqrt{n}$.

Getting a Cleaner Statement (1/2)

- [Peikert09] showed a reduction from GapSVP to solving BDD to within distance $\lambda_1(\Lambda)/\text{poly}(n)$
- Hence, if $\alpha q/r \geq \lambda_1(\Lambda)/\text{poly}(n)$ (i.e., $r \geq q \cdot \text{poly}(n)/\lambda_1(\Lambda)$) then we get a solution to the standard lattice problem GapSVP
- But how do we obtain samples from $D_{\Lambda^*,r}$?
- [GentryPeikertVaikuntanathan08] showed that such samples can be obtained from a basis with vectors of length r
  - So using the [LLL82] algorithm (that efficiently produces a basis of length $2^n/\lambda_1(\Lambda)$), we get hardness of LWE with $q=2^{O(n)}$ based on GapSVP
  - For polynomial q, we get hardness based on GapSVP given a somewhat short basis

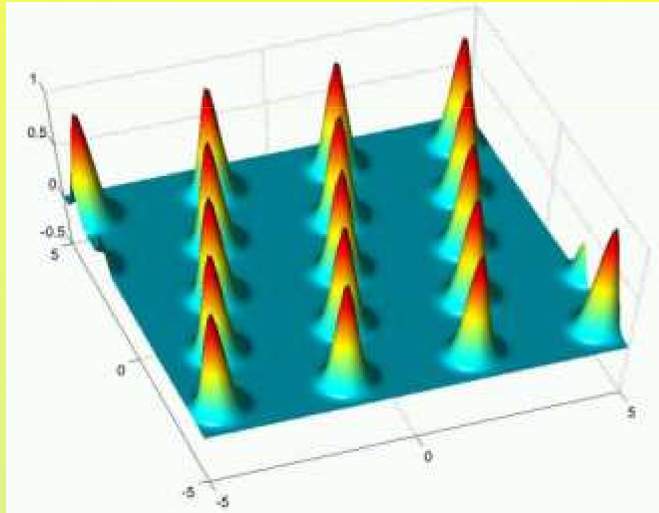
Getting a Cleaner Statement (1/2)

- [Peikert09] showed a reduction from GapSVP to solving BDD to within distance $\lambda_1(\Lambda)$/poly(n)
- Since sampling from $D_{\Lambda^*,r}$ for $r=2^n/\lambda_1(\Lambda)$ can be done efficiently, we obtain hardness of LWE for exponential moduli q

- Alternatively, we can use the sampler in [GentryPeikertVaikuntanathan08] to show hardness of LWE with polynomial moduli q based the assumption that GapSVP is hard even given a somewhat short vector

Getting a Cleaner Statement (2/2)

- Alternatively, [R05] showed a quantum reduction from sampling $D_{\Lambda^*,\sqrt{n}/d}$ to solving BDD in Λ with distance d.
- Assume $\alpha q \geq 2\sqrt{n}$, and combine with the core proposition:



Samples from $D_{\Lambda^*,r}$ → Solution to $BDD_{\Lambda,\alpha q/r}$

Samples from $D_{\Lambda^*,r/2}$ → Solution to $BDD_{\Lambda, 2\alpha q/r}$

Samples from $D_{\Lambda^*,r/4}$ → Solution to $BDD_{\Lambda, 4\alpha q/r}$

⋮

Proof of Core Proposition (1/2)

- For simplicity, assume $\Lambda=\mathbb{Z}^n$ (and ignore the fact that this lattice is 'easy')
- We are given:
  - An oracle that solves LWE with modulus q and parameter α
  - Samples from $D_{\mathbb{Z}^n,r}$
- Our input is a point $x\in\mathbb{R}^n$ within distance $\alpha q/r$ of some unknown $v\in\mathbb{Z}^n$
- Our goal is to output v
- We will show how to generate LWE samples with secret s=(v mod q)
- Using the LWE oracle, we can find v mod q; this allows to find v itself using a straightforward reduction
- Summarizing:
  - Given: samples from $D_{\mathbb{Z}^n,r}$
  - Input: a point $x\in\mathbb{R}^n$ within distance $\alpha q/r$ of some unknown $v\in\mathbb{Z}^n$
  - Goal: generate LWE samples with secret s=(v mod q)

Proof of Core Proposition (2/2)

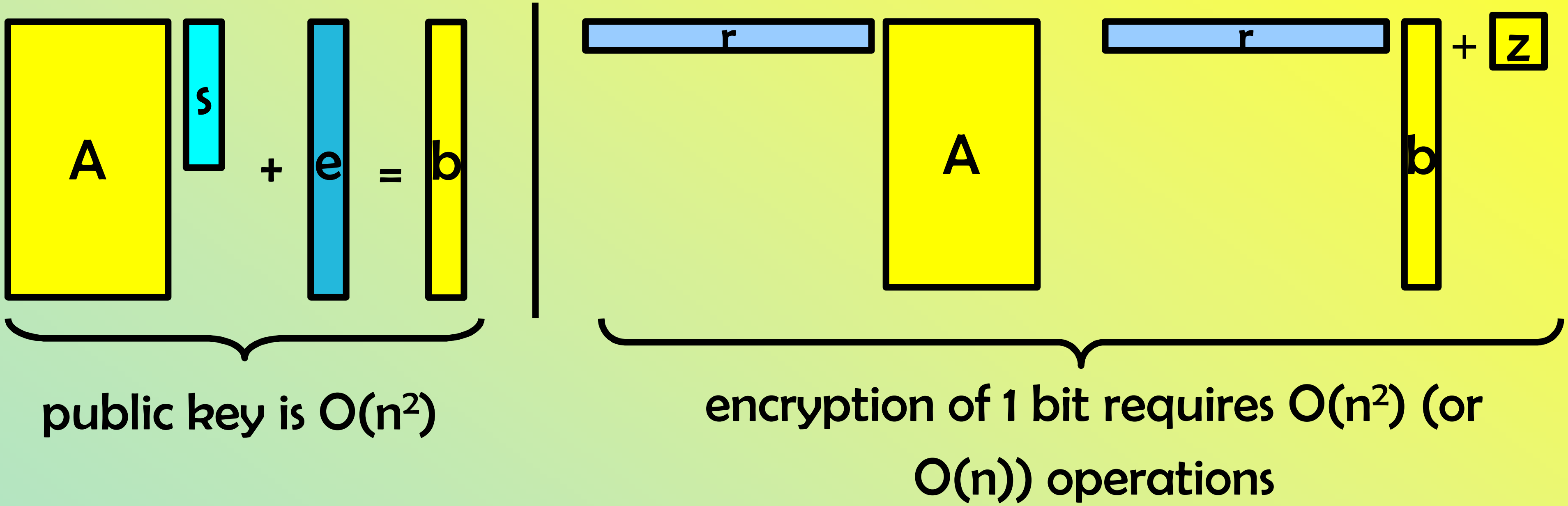
- This is done as follows:
  - Take a sample y from $D_{\mathbb{Z}^n,r}$
  - Output the pair

$$(a = y \bmod q,\ \ b = \lfloor \langle y,x \rangle \rceil \bmod q) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

- Analysis:
  - Since r is not too small, a is uniformly distributed in $\mathbb{Z}_q^n$
  - Now condition on any fixed value of a, and let's analyze the distribution of b.
  - y is distributed as a discrete Gaussian on $q\mathbb{Z}^n+a$
  - If x=v, then b is exactly $\langle a,s \rangle$, so we get LWE samples with no error
  - Otherwise, we get an error term of the form $\langle y,x-v \rangle$. Since x-v is a fixed vector of norm $<\alpha q/r$, and y is Gaussian of norm r, this inner product is normal with standard deviation $<\alpha q$.

LWE over Rings

Some Inefficiencies of LWE-Based Schemes



Source of Inefficiency

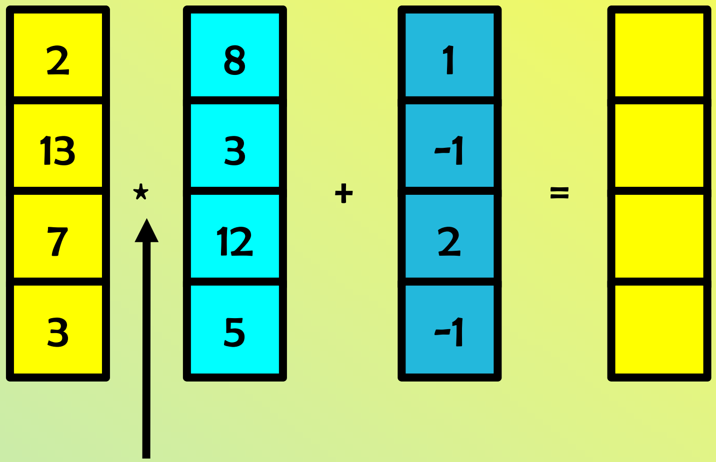
$$\begin{pmatrix} 2 & 13 & 7 & 3 \end{pmatrix} \cdot \begin{pmatrix} 8 \\ 3 \\ 12 \\ 5 \end{pmatrix} + \begin{pmatrix} 1 \end{pmatrix} = \begin{pmatrix} 13 \end{pmatrix}$$

- Getting just one extra random-looking number requires n random numbers!

- Wishful thinking: get n random numbers and produce O(n) pseudo-random numbers in “one shot”

$$\begin{pmatrix} 2 \\ 13 \\ 7 \\ 3 \end{pmatrix} * \begin{pmatrix} 8 \\ 3 \\ 12 \\ 5 \end{pmatrix} + \begin{pmatrix} 1 \\ -1 \\ 2 \\ -1 \end{pmatrix} = \begin{pmatrix} \; \\ \; \\ \; \\ \; \end{pmatrix}$$

Main Question

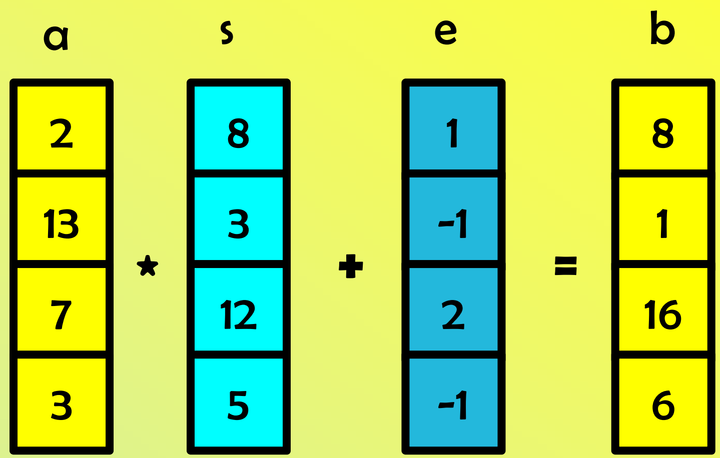


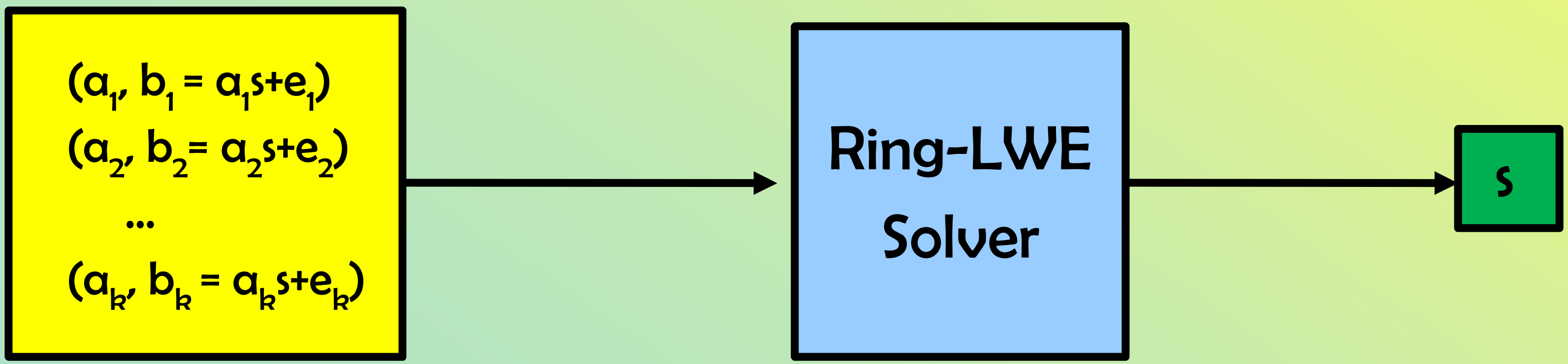
- How do we define multiplication so that the resulting distribution is pseudorandom? (Coordinate-wise multiplication is not secure)

- Answer: Define it as multiplication in a polynomial ring
  - Similar ideas used in the heuristic design of NTRU [HoffsteinPipherSilverman98], and in compact one-way functions [Micciancio02, PeikertRosen06, LyubashevskyMicciancio06,…].

The Ring-LWE Problem

- Let R be the ring $\mathbb{Z}_q[x]/\langle x^n+1\rangle$
- The secret s is now an element in R
- The elements a are chosen uniformly from R
- The coefficients of the noise polynomial e are chosen as small independent normal vars





Ring-LWE – Known Results

- [LyubashevskyPeikertR10] show that Ring-LWE is as hard as (quantumly) solving the standard lattice problem SIVP (on ideal lattices)
  - The proof is by adapting [R05]'s proof to rings; only the classical part needs to be changed
  - A qualitatively weaker result was independently shown by [Stehlé SteinfeldTanakaXagawa09] using different techniques of independent interest.
- [LPR10] also show that decision Ring-LWE is as hard as (search) Ring-LWE
  - Proof is quite non-trivial!
- Finally [LPR10] show how this can be used to construct very efficient cryptographic applications
- More details in the survey paper!

Open Questions

- Obtain the ultimate hardness result for LWE (as for SIS)
  - $500 prize
- Hardness of LPN?
  - Or is LPN easier?
  - $250 prize
- Understand practical parameters of LWE [RückertSchneider10]
- More algorithms for LWE
- Further cryptographic applications of LWE
  - Direct construction of efficient pseudorandom functions
  - Fully homomorphic encryption scheme (perhaps based on ring-LWE)?
- ‘Upgrade’ all existing constructions to ring-LWE
- Reduction from LWE to classical problems, similar to what was done in [Feige02]