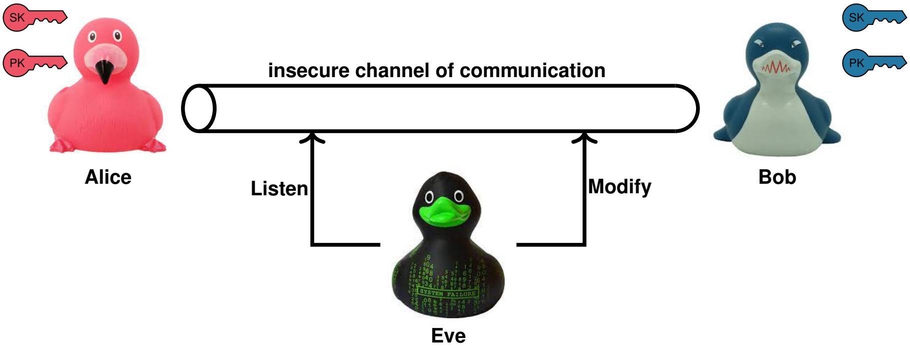
Quantum Algorithms for Lattice-based Cryptography

Yixin Shen

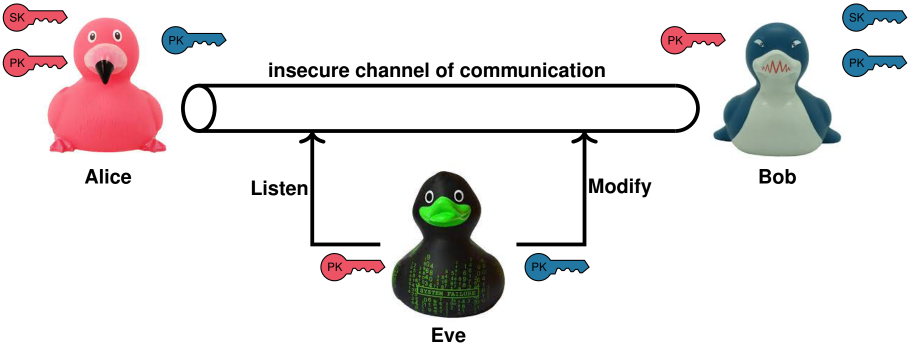
November 1, 2022

Hard Problems in Public Key Cryptography



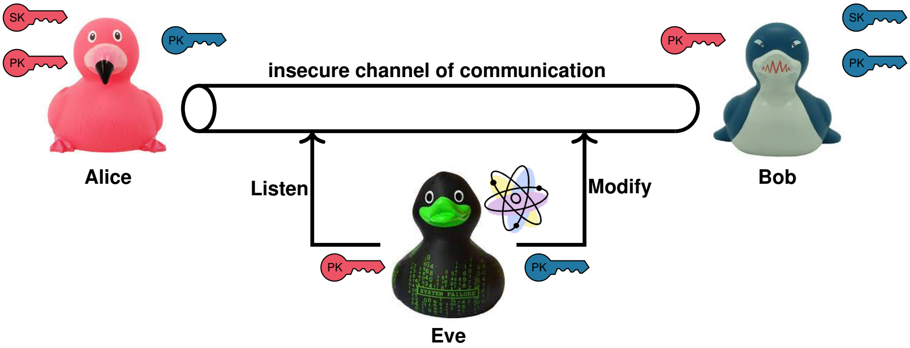
factorisation, discrete logarithm

Hard Problems in Public Key Cryptography



factorisation, discrete logarithm

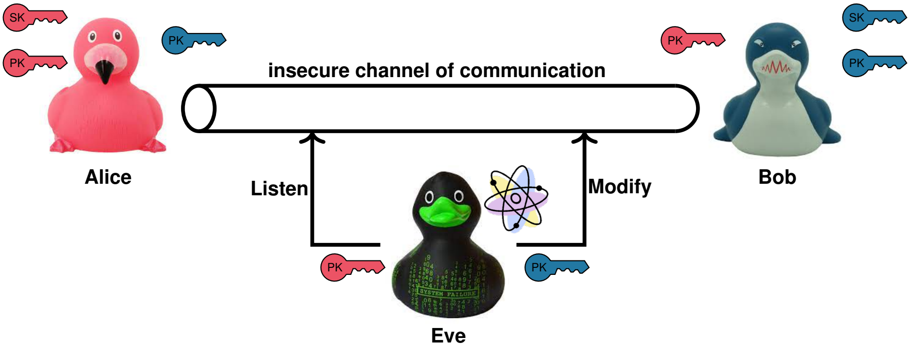
Hard Problems in Public Key Cryptography



~~factorisation, discrete logarithm~~

lattices, codes, ...

Hard Problems in Public Key Cryptography



~~factorisation, discrete logarithm~~

lattices, codes, ...

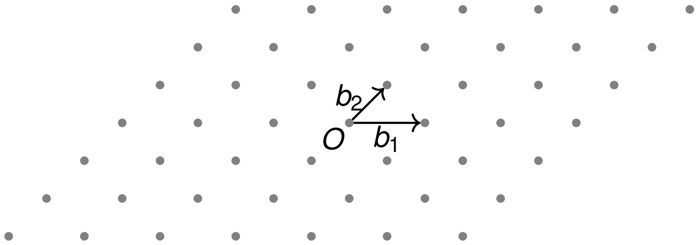
NIST selected algorithms:

- encryption: the only selected candidate is based on lattices
- signatures: 2 out of 3 based on lattices

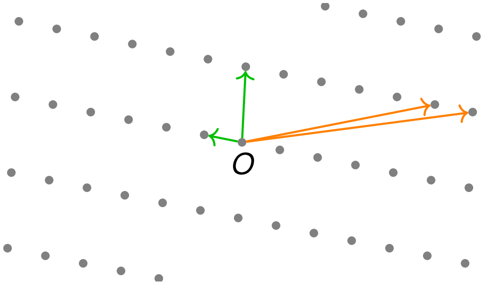
What is a (Euclidean) lattice?

Definition

$\mathcal{L}(\boldsymbol{b}_1, \dots, \boldsymbol{b}_n) = \left\{ \sum_{i=1}^{n} x_i \boldsymbol{b}_i : x_i \in \mathbb{Z} \right\}$ where $\boldsymbol{b}_1, \dots, \boldsymbol{b}_n$ is a basis of $\mathbb{R}^n$.

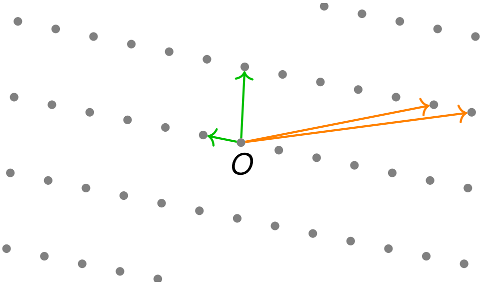


Lattice-based cryptography: fundamental idea



- good basis: private information, makes problem easy
- bad basis: public information, makes problem hard

Lattice-based cryptography: fundamental idea

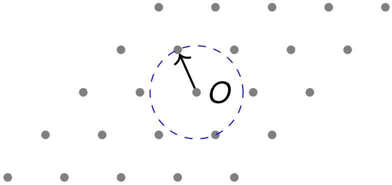


- good basis: private information, makes problem easy
- bad basis: public information, makes problem hard

Basis reduction: transform a bad basis into a good one
Main tool: BKZ algorithm and its variants

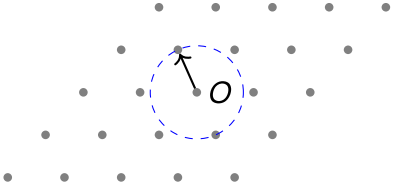
Requires to solve the (approx-)SVP problem in smaller dimensions.

The Shortest Vector Problem

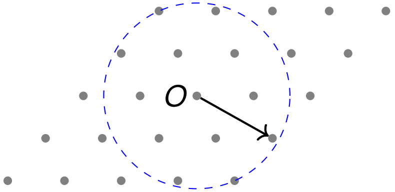


Shortest Vector Problem (SVP):
Given a basis for the lattice $\mathcal{L}$, find a shortest nonzero lattice vector.
$\lambda_1(\mathcal{L})$ = length of such a vector.

The Shortest Vector Problem



Shortest Vector Problem (SVP):
Given a basis for the lattice $\mathcal{L}$, find a shortest nonzero lattice vector.
$\lambda_1(\mathcal{L})$ = length of such a vector.



γ-approx-SVP ($\gamma > 1$):
Given a basis of $\mathcal{L}$, find a nonzero lattice vector of length at most $\gamma \cdot \lambda_1(\mathcal{L})$.
γ is approximation factor.

The Shortest Vector Problem

Depending on the dimension n:

- NP-Hardness (randomized reduction)
- NP $\cap$ co-NP
- Subexponential-time algorithms
- Poly-time algorithms

Approx factor:

- $O(1)$
- $\sqrt{n}$
- $2^{\sqrt{n}}$
- $2^{\frac{n \log \log n}{\log n}}$

The Shortest Vector Problem

Depending on the dimension n:

- NP-Hardness (randomized reduction)
- NP $\cap$ co-NP
- Subexponential-time algorithms
- Poly-time algorithms

Approx factor:

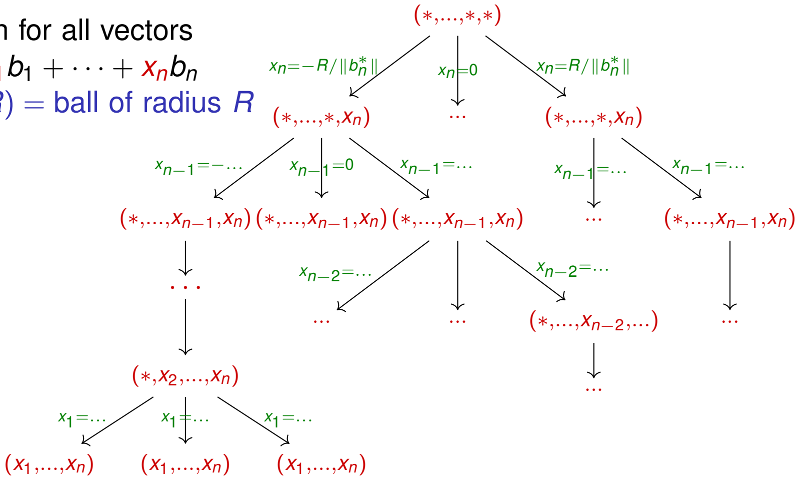
- $O(1)$
- $\sqrt{n}$
- $2^{\sqrt{n}}$
- $2^{\frac{n \log \log n}{\log n}}$

Main approaches for SVP:

- Enumeration: $2^{O(n \log(n))}$ time and $\text{poly}(n)$ space
- Sieving: $2^{O(n)}$ time and $2^{O(n)}$ space

Enumeration Algorithm

Search for all vectors
$X = x_1 b_1 + \cdots + x_n b_n$
in $B(R)$ = ball of radius R



Given $x_n, \ldots, x_{i+1}, \|x\| \leqslant R \Rightarrow$ the integer x_i belongs to an interval of small length

Quantum Speed-up for Enumeration

Quantum Backtracking [Montanaro15]

- blackbox access to a tree with marked nodes:
  - can only query the local structure of the tree
- tree of size T, depth n, constant max degree

$\Rightarrow$ $O^*(\sqrt{T})$ queries to find a marked node [ANS18]

Combine with clever ways of pruning the tree

Quantum Speed-up for Enumeration

Quantum Backtracking [Montanaro15]

- blackbox access to a tree with marked nodes:
  - can only query the local structure of the tree
- tree of size T, depth n, constant max degree

$\Rightarrow$ $O^*(\sqrt{T})$ queries to find a marked node [ANS18]

Combine with clever ways of pruning the tree

Variational Quantum Algorithm [APSW22]

- ignore the tree structure, encode SVP into a QUBO
- upper bound the x_is
- modify the classical loop of VQE to target directly the first excited state
- 1500 qubits suffice to solve SVP in dim 180
- time complexity? unknown...

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:

Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:

Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Idea: LLL reduced $\rightsquigarrow$ length $\ell \leqslant 2^{O(n)}\lambda_1$, sieve $O(n)$, solve SVP

Sieving Algorithms

Original idea [AKS01]:

- Reduce basis
- Generate random vectors
- Repeat many times:
  - Sieve vectors

Sieve:

Input: many vectors of length $\leqslant \ell$
Output: many vectors of length $\leqslant \frac{\ell}{2}$

Combine pairs of vectors to produce shorter vectors

Idea: LLL reduced $\rightsquigarrow$ length $\ell \leqslant 2^{O(n)}\lambda_1$, sieve $O(n)$, solve SVP

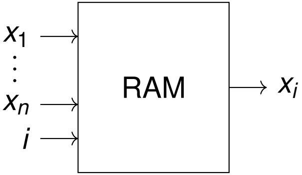
Many heuristic variants: local sensitive hash, tuple sieve, ...

Interlude: quantum memory models

classical access

quantum access

classical data



standard

quantum data

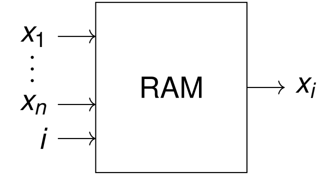
Assumption: $O(1)$ time cost

Interlude: quantum memory models

classical access

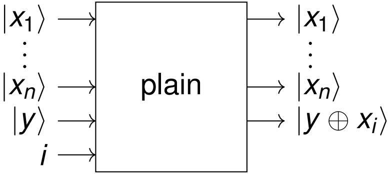
quantum access

classical data



standard

quantum data



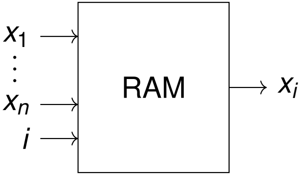
standard

Assumption: $O(1)$ time cost

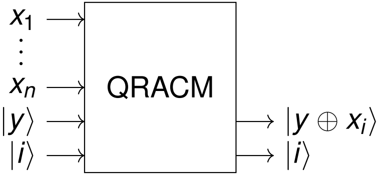
Interlude: quantum memory models

classical access | quantum access

classical data

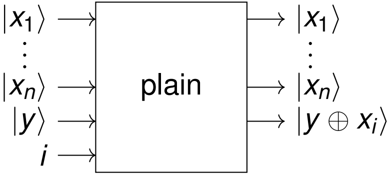


standard



potentially strong assumption

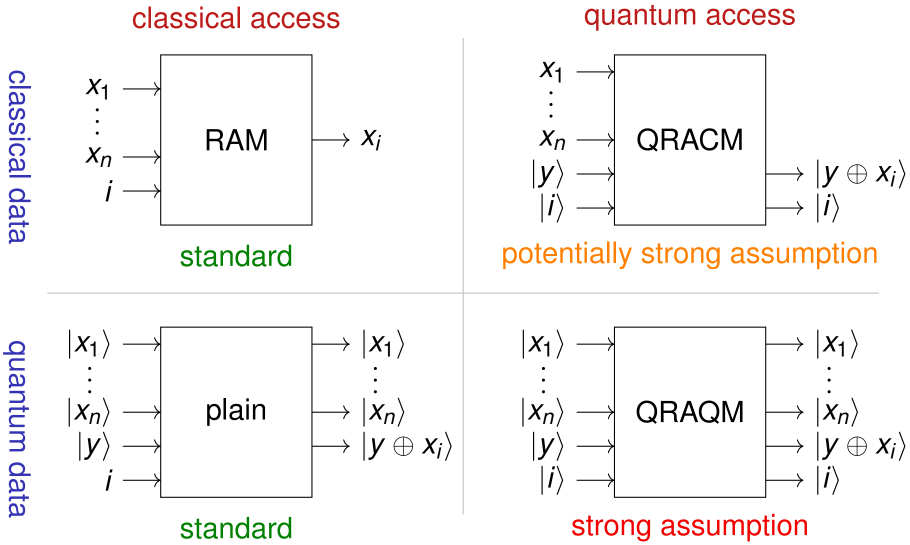
quantum data



standard

Assumption: $O(1)$ time cost

Interlude: quantum memory models



Assumption: $O(1)$ time cost

Results in the Classical and Quantum Setting

Time Complexity	PH	Space Complexity: Classical	Qubits	Model	Reference
$2^{n+o(n)}$	P	$2^{n+o(n)}$	N/A	N/A	[LMP15]
$2^{0.950n+o(n)}$	P	$2^{0.5n+o(n)}$	poly(n)	plain	[ACKS21]
$2^{0.835n+o(n)}$	P	$2^{0.5n+o(n)}$	poly(n)	QRACM	[ACKS21]
$2^{0.292n+o(n)}$	H	$2^{0.292n+o(n)}$	N/A	N/A	[BDGL16]
$2^{0.265n+o(n)}$	H	$2^{0.265n+o(n)}$	poly(n)	QRACM	[Laa15]
$2^{0.2563n+o(n)}$	H	$2^{0.2075n+o(n)}$	poly(n)	QRAQM	[BCSS22]

P=provable, H=heuristic

Tools: Quantum Amplitude Amplification/Estimation, Quantum Walks

Currently interested in: Heuristic algorithms for lattice sieving without QRACM. Such algorithm exists for collision finding: [CNS17].