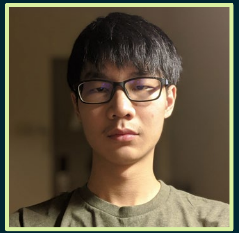
Worst-case to Average-case Hardness of LWE:

An Alternative Perspective


Divesh Aggarwal


Leong Jin Ming


Alexandra Veliche

Cryptographic from LWE

LWE

IND-CPA-secure
Public-Key Encryption
[Regev, 2005]

CCA-secure
Encryption
[Peikert, 2009]

…

Key Exchange Scheme
[Ding-Xie-Lin, 2012],
[Peikert, 2014]

Cryptographic Significance

worst-case problems → average-case problem: **LWE**

LWE →

- IND-CPA-secure Public-Key Encryption [Regev, 2005]
- CCA-secure Encryption [Peikert, 2009]
- …
- Key Exchange Scheme [Ding-Xie-Lin, 2012], [Peikert, 2014]

Cryptographic Significance

worst-case hardness $\Longrightarrow$ average-case hardness $\Longrightarrow$ cryptosystems

GapSVP BDD $\longrightarrow$ LWE

Cryptographic Significance

worst-case hardness $\Longrightarrow$ average-case hardness $\Longrightarrow$ Post-quantum cryptosystems

GapSVP BDD $\longrightarrow$ LWE

no known efficient quantum algorithms for these problems

Learning With Errors

$\mathbf{LWE}_{n,p,\phi}$: n dimension, p modulus, $\phi \sim \mathbb{R}/\mathbb{Z}$ error distribution

Given noisy samples $(\mathbf{a},\ \langle \mathbf{a}, \mathbf{s} \rangle + e)$, where

$\mathbf{a} \leftarrow \mathbb{Z}_p^n$ uniformly random, $\mathbf{s} \in \mathbb{Z}_p^n$ unknown, $e \leftarrow \phi$ small error,

(search-LWE) output $\mathbf{s}$.

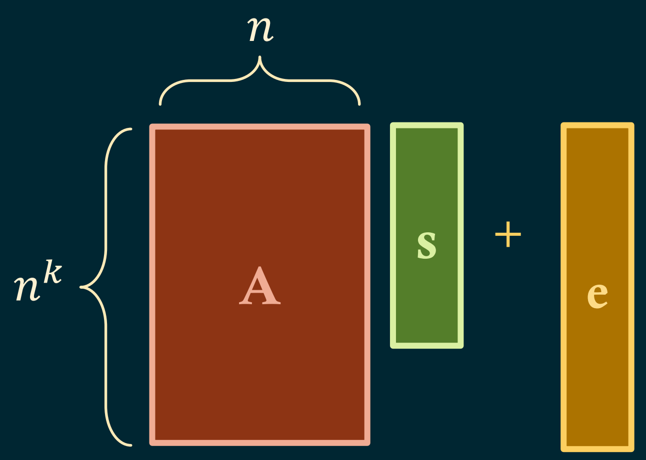
Learning With Errors

LWE$_{n,p,\phi}$: n dimension, p modulus, $\phi \sim \mathbb{R}/\mathbb{Z}$ error distribution

Given noisy samples $(\mathbf{a},\ \langle \mathbf{a}, \mathbf{s} \rangle + e)$, where

$\mathbf{a} \leftarrow \mathbb{Z}_p^n$ uniformly random, $\mathbf{s} \in \mathbb{Z}_p^n$ unknown, $e \leftarrow \phi$ small error,

(search-LWE) output $\mathbf{s}$.



Learning With Errors

$\mathbf{LWE}_{n,p,\phi}$: n dimension, p modulus, $\phi \sim \mathbb{R}/\mathbb{Z}$ error distribution

Given noisy samples $(\mathbf{a}, b)$, where

$\mathbf{a} \leftarrow \mathbb{Z}_p^n$ uniformly random and $b \in \mathbb{Z}_p$,

(decision-LWE) output

- YES if samples are from the LWE distribution for $\mathbf{s}$ and ϕ,
- NO if samples are uniformly random.

Lattices

Lattice:

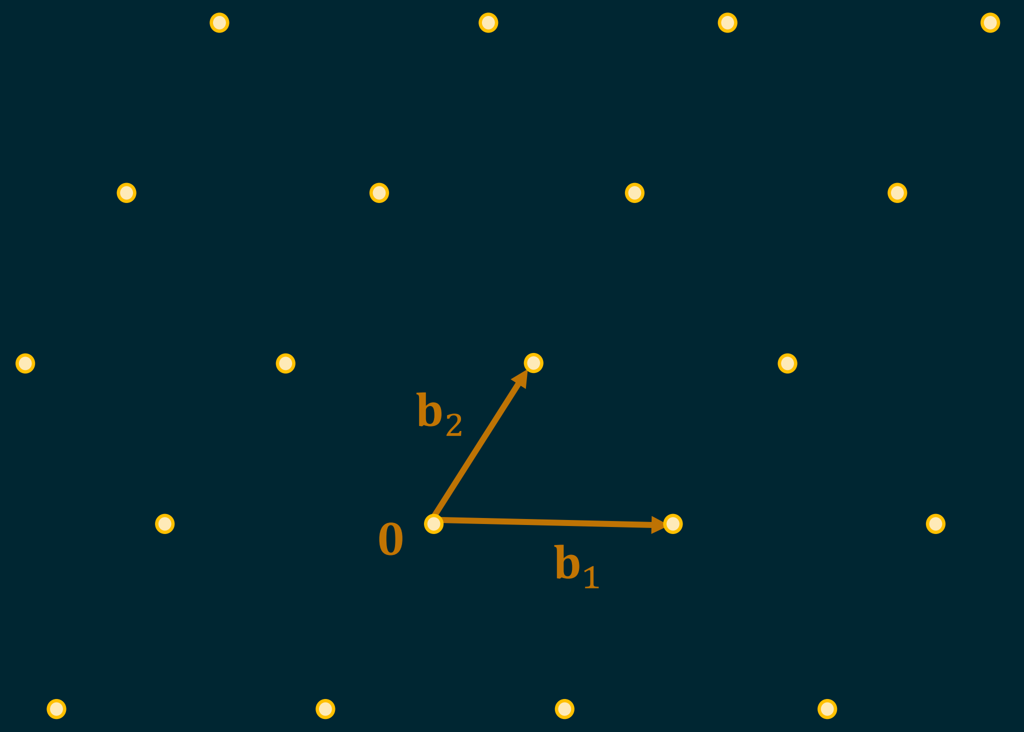
An infinite discrete set of vectors in $\mathbb{R}^n$

consisting of all integer linear combinations

$$\mathcal{L} = \{a_1\mathbf{b}_1 + \cdots + a_k\mathbf{b}_k \;:\; a_1, \dots, a_k \in \mathbb{Z}\}$$

of some linearly independent vectors $\mathbf{b}_1, \dots, \mathbf{b}_k \in \mathbb{R}^n$.

The set $\{\mathbf{b}_1, \dots, \mathbf{b}_k\}$ is called a *basis*.



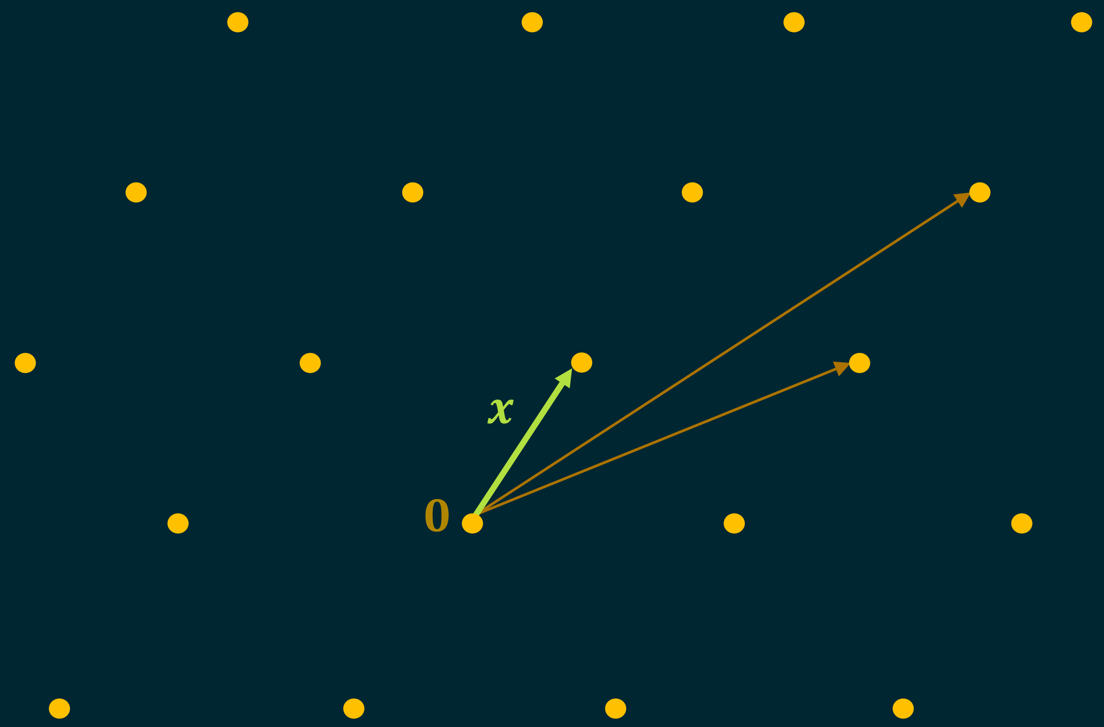
Shortest Vector Problem

SVP :

Given a basis $\mathcal{B}$ for lattice $\mathcal{L} \subset \mathbb{R}^n$,

find a shortest non-zero lattice vector $\boldsymbol{x}$,

i.e. $\boldsymbol{x} \in \mathcal{L} \setminus \{\mathbf{0}\}$, such that $\|\boldsymbol{x}\| = \lambda_1(\mathcal{L})$.



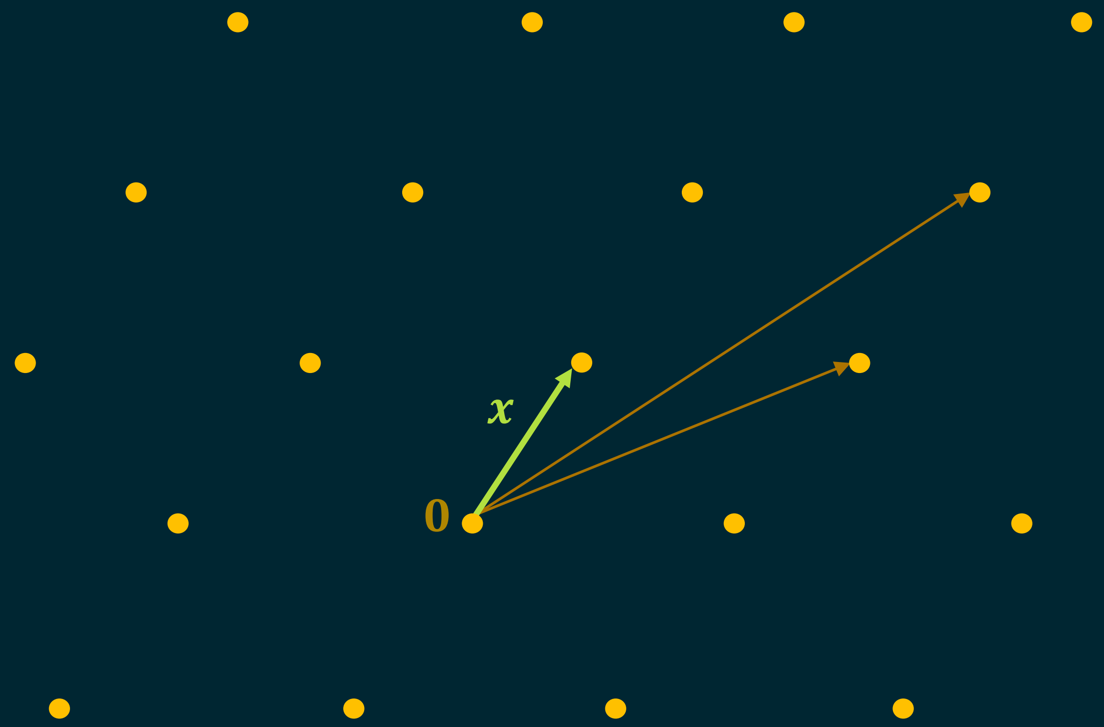
Shortest Vector Problem

SVP :

Given a basis $\mathcal{B}$ for lattice $\mathcal{L} \subset \mathbb{R}^n$,

find a shortest non-zero lattice vector $\boldsymbol{x}$,

i.e. $\boldsymbol{x} \in \mathcal{L} \setminus \{\mathbf{0}\}$, such that $\|\boldsymbol{x}\| = \lambda_1(\mathcal{L})$.



GapSVP$_\gamma$ is an approximate decision variant.

Bounded Distance Decoding

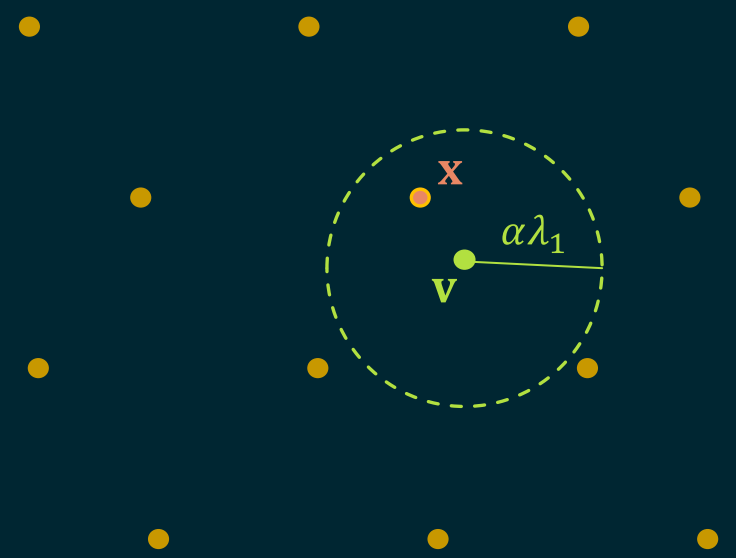
$\mathbf{BDD}_{\alpha}$: $\alpha > 0$ distance approximation factor

Given a basis $\mathcal{B}$ for a full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$

and a target vector $\mathbf{v} \in \mathbb{R}^n$ close to the lattice,

find a lattice vector $x \in \mathcal{L}$ closest to $\mathbf{v}$,

i.e. $\mathbf{x} \in \mathcal{L}$ such that $\|\mathbf{v} - \mathbf{x}\|_2 < \alpha \cdot \lambda_1(\mathcal{L})$.



Hardness of LWE

[Regev, 2009] – quantum reduction from worst-case lattice problems to decision-LWE

Hardness of LWE

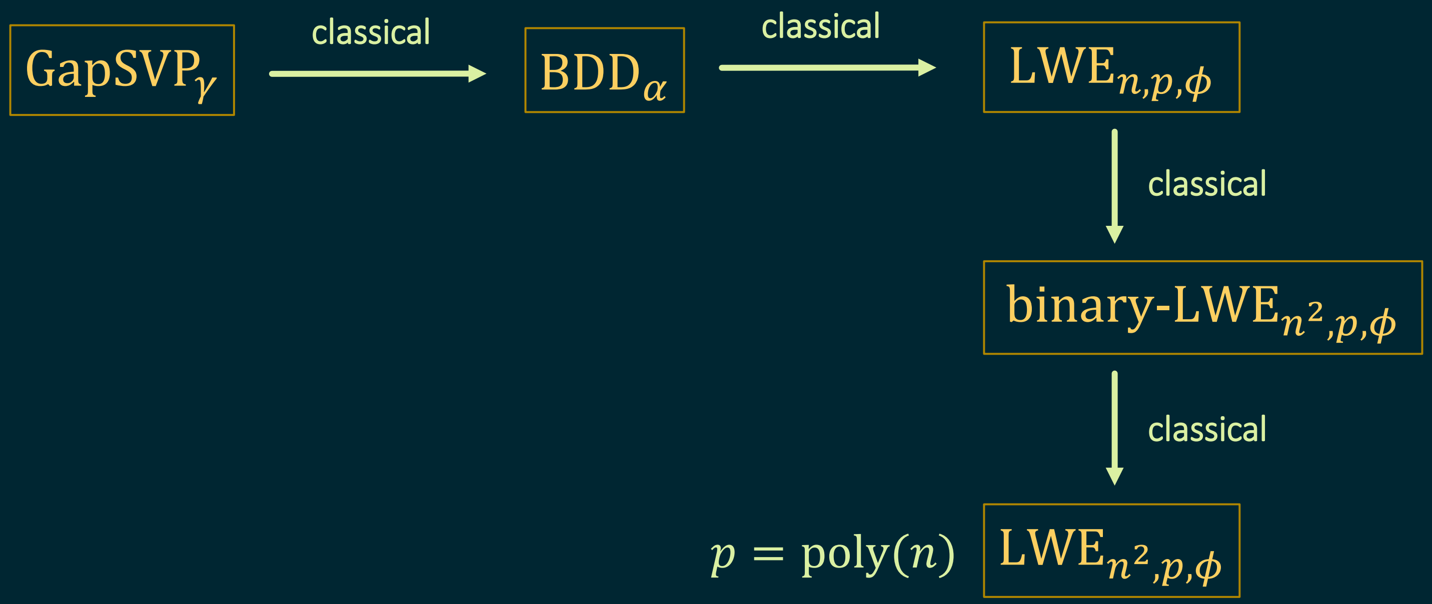
[Peikert, 2009] – classical reduction, but modulus becomes exponential

$$\mathrm{GapSVP}_{\gamma} \xrightarrow{\text{classical}} \mathrm{BDD}_{\alpha} \xrightarrow{\text{classical}} \mathrm{LWE}_{n,p,\phi}$$

$p = \exp(n)$

Hardness of LWE

[Brakerski, Peikert, Langlois, Regev, Stehle, 2013] – classical reduction with polynomial modulus



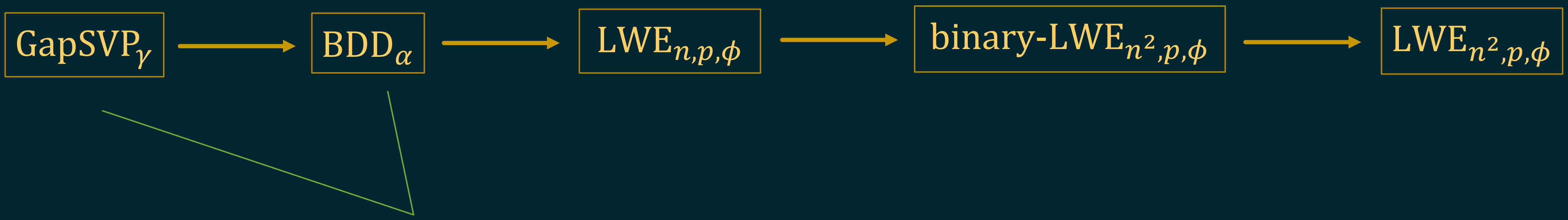
Hardness of LWE

$\mathrm{GapSVP}_\gamma \rightarrow \mathrm{BDD}_\alpha \rightarrow \mathrm{LWE}_{n,p,\phi} \rightarrow \text{binary-}\mathrm{LWE}_{n^2,p,\phi} \rightarrow \mathrm{LWE}_{n^2,p,\phi}$

$\mathrm{LWE}_{n,p,\phi}$: $p = \exp(n)$

$\mathrm{LWE}_{n^2,p,\phi}$: $p = \mathrm{poly}(n)$

Algorithms for Lattice Problems



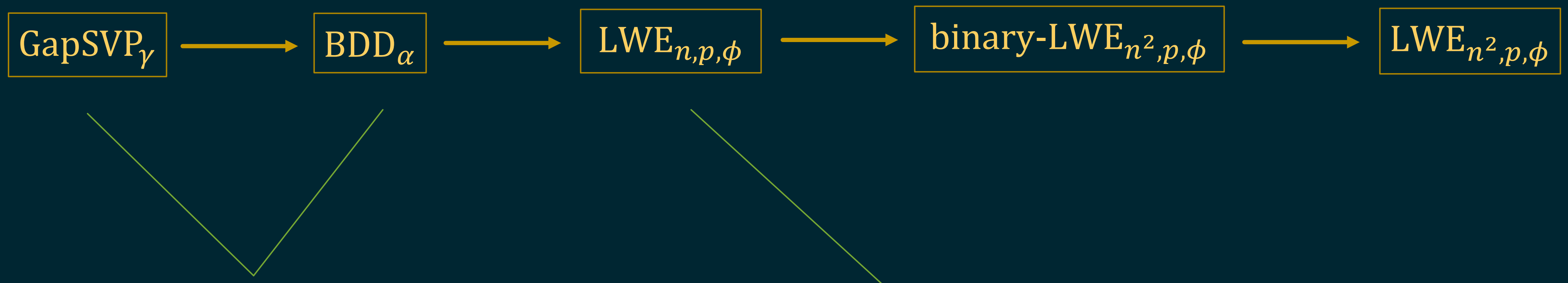
Fastest algorithms for these
problems run in $2^{\Theta(n)}$ time
(for polynomial approximation factor).

What the Reduction says about LWE Algorithms

$\text{GapSVP}_\gamma \longrightarrow \text{BDD}_\alpha \longrightarrow \text{LWE}_{n,p,\phi} \longrightarrow \text{binary-LWE}_{n^2,p,\phi} \longrightarrow \text{LWE}_{n^2,p,\phi}$

Conjecture: known algorithms are the best possible

What the Reduction says about LWE Algorithms



Conjecture: known algorithms are the best possible $\Longrightarrow$ LWE cannot be solved in less than $2^{\Omega(\sqrt{n})}$ time.

What the Reduction says about LWE Algorithms

[Blum-Kalai-Wasserman, 2000] – Best known algorithm for $\mathrm{LWE}_{n,p,\phi}$ runs in $2^{O(n \log p \,/\, \log n)}$ time.

$\mathrm{GapSVP}_\gamma \longrightarrow \mathrm{BDD}_\alpha \longrightarrow \mathrm{LWE}_{n,p,\phi} \longrightarrow \text{binary-}\mathrm{LWE}_{n^2,p,\phi} \longrightarrow \mathrm{LWE}_{n^2,p,\phi}$

Conjecture: known algorithms are the best possible $\Longrightarrow$ LWE cannot be solved in less than $2^{\Omega(\sqrt{n})}$ time.

What the Reduction says about LWE Algorithms

[Blum-Kalai-Wasserman, 2000] – Best known algorithm for $\mathrm{LWE}_{n,p,\phi}$ runs in $2^{O(n \log p \,/\, \log n)}$ time.

$\mathrm{GapSVP}_\gamma \rightarrow \mathrm{BDD}_\alpha \rightarrow \mathrm{LWE}_{n,p,\phi} \rightarrow \text{binary-}\mathrm{LWE}_{n^2,p,\phi} \rightarrow \mathrm{LWE}_{n^2,p,\phi}$

Big Gap!

Conjecture: known algorithms are the best possible $\Longrightarrow$ LWE cannot be solved in less than $2^{\Omega(\sqrt{n})}$ time.

Closing the Gap

How to close this gap?

We change our perspective!

Security in Practice

What does it mean for a cryptosystem to be 256-bit secure?

Security in Practice

What does it mean for a cryptosystem to be 256-bit secure?

(a) The fastest algorithm for breaking the cryptosystem runs in 2^{256} time.

(b) No reasonably efficient algorithm can break the cryptosystem with probability $> 2^{-256}$.

Security in Practice

What does it mean for a cryptosystem to be 256-bit secure?

(a) The fastest algorithm for breaking the cryptosystem runs in 2^{256} time.

(b) No reasonably efficient algorithm can break the cryptosystem with probability $> 2^{-256}$.

This is what we usually want for cryptographic security

An Alternative Perspective

An alternative measure of computational hardness:

The maximum success probability of any PPT algorithm that finds a solution.

An Alternative Perspective

An alternative measure of computational hardness:

The maximum success probability of any PPT algorithm that finds a solution.

Can we study worst-case to average-case reductions under this framework?

An Alternative Perspective

An alternative measure of computational hardness:

The maximum success probability of any PPT algorithm that finds a solution.

Can we study worst-case to average-case reductions under this framework?

Yes (this talk!)

Success Probability of Solving LWE

Trivial algorithm (guess the error): Success probability for solving $\mathrm{LWE}_{n,p,\phi}$ is $p^{-\Omega(n)}$.

Success Probability of Solving LWE

Trivial algorithm (guess the error): Success probability for solving $\mathrm{LWE}_{n,p,\phi}$ is $p^{-\Omega(n)}$.

All other algorithms are not PPT, so it is unlikely that we can achieve better than this.

Success Probability of Solving Lattice Problems

LLL / Slide Reduction + guess coefficients: Success probability of solving GapSVP_γ is $2^{-\Theta(n^2/\log n)}$.

Success Probability of Solving Lattice Problems

LLL / Slide Reduction + guess coefficients: Success probability of solving GapSVP_γ is $2^{-\Theta(n^2/\log n)}$.

Known techniques do not seem to improve this when restricted to PPT algorithms,

so it is unlikely that we can achieve much better than this.

Success Probability of Solving Lattice Problems

LLL / Slide Reduction + guess coefficients: Success probability of solving GapSVP_γ is $2^{-\Theta(n^2/\log n)}$.

Known techniques do not seem to improve this when restricted to PPT algorithms,

so it is unlikely that we can achieve much better than this.

BDD_α is closely related to GapSVP_γ for $\gamma = \mathrm{poly}(n) = \frac{1}{\alpha}$,

so it is unlikely we can achieve better than known algorithms.

A Natural Conjecture

Conjecture:

(informal) No algorithm can solve BDD_α on an arbitrary n-rank lattice for $\alpha = 1/\mathrm{poly}(n)$ in polynomial time with success probability better than $2^{-n^2/\log n}$.

What We Show

Trivial algorithm: Success probability for efficiently solving $\mathrm{LWE}_{n,p,\phi}$ is $p^{-\Omega(n)}$.

Conjecture $\Longrightarrow$ Maximum success probability for efficiently solving $\mathrm{LWE}_{n,p,\phi}$ is $p^{-\Omega(n/\log^2 n)}$.

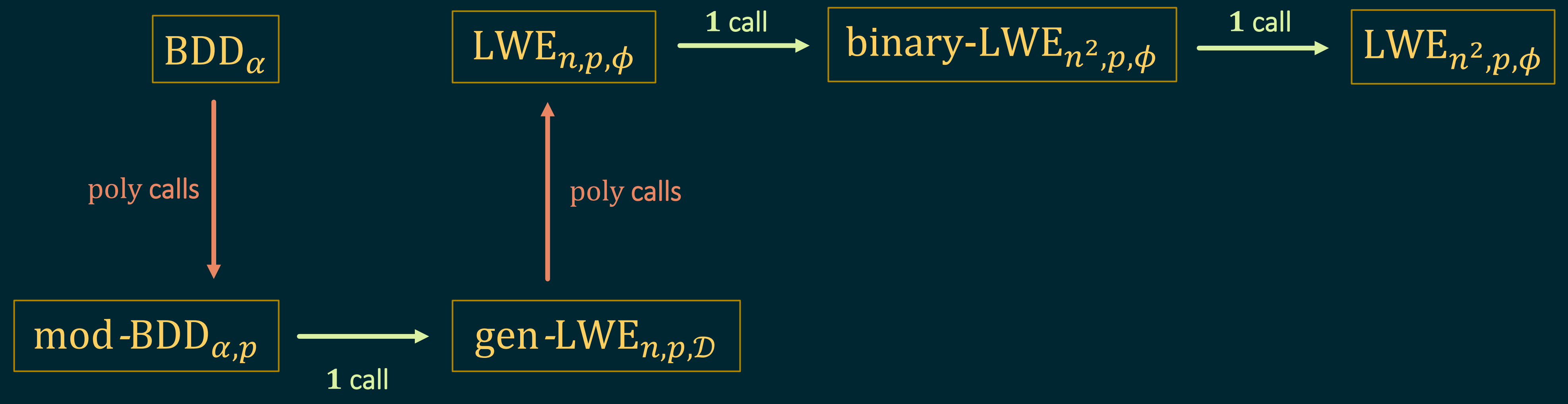
What We Show

Trivial algorithm: Success probability for efficiently solving $\mathrm{LWE}_{n,p,\phi}$ is $p^{-\Omega(n)}$.

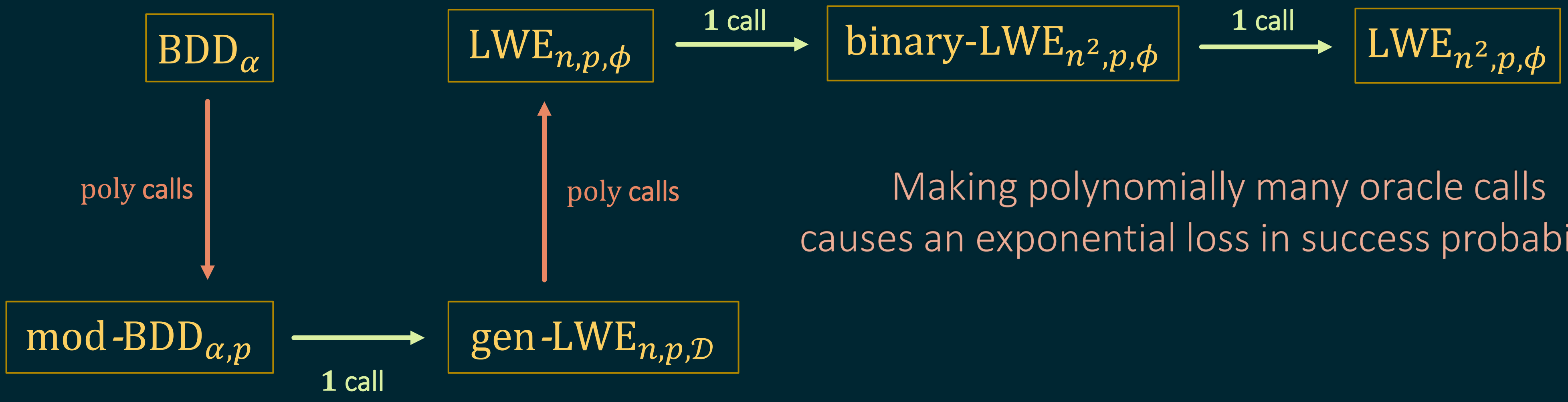
Tight!

Conjecture $\Longrightarrow$ Maximum success probability for efficiently solving $\mathrm{LWE}_{n,p,\phi}$ is $p^{-\Omega(n/\log^2 n)}$.

Limitations of the Original Reduction



Limitations of the Original Reduction



Making polynomially many oracle calls
causes an exponential loss in success probability!

Limitations of the Original Reduction

Reduction algorithm for $\mathcal{P} \to \mathcal{Q}$ makes k calls to oracle for $\mathcal{Q}$.

Success probability of solving $\mathcal{Q}$ is $\geq \epsilon \implies$ success probability of solving $\mathcal{P}$ is $\geq \epsilon^k$.

Limitations of the Original Reduction

Reduction algorithm for $\mathcal{P} \to \mathcal{Q}$ makes k calls to oracle for $\mathcal{Q}$.

Success probability of solving $\mathcal{Q}$ is $\geq \epsilon \implies$ success probability of solving $\mathcal{P}$ is $\geq \epsilon^k$.

Success probability of solving $\mathcal{P}$ is $\leq \delta \implies$ success probability of solving $\mathcal{Q}$ is $\leq \delta^{1/k}$.

Limitations of the Original Reduction

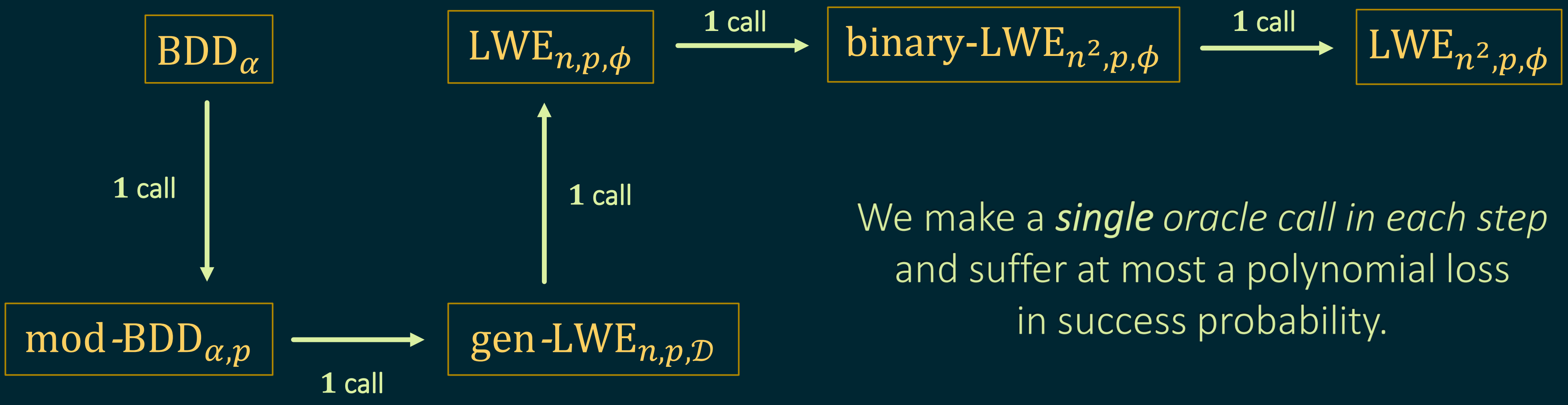
Reduction algorithm for $\mathcal{P} \to \mathcal{Q}$ makes k calls to oracle for $\mathcal{Q}$.

Success probability of solving $\mathcal{Q}$ is $\geq \epsilon \implies$ success probability of solving $\mathcal{P}$ is $\geq \epsilon^k$.

Success probability of solving $\mathcal{P}$ is $\leq \delta \implies$ success probability of solving $\mathcal{Q}$ is $\leq \delta^{1/k}$.

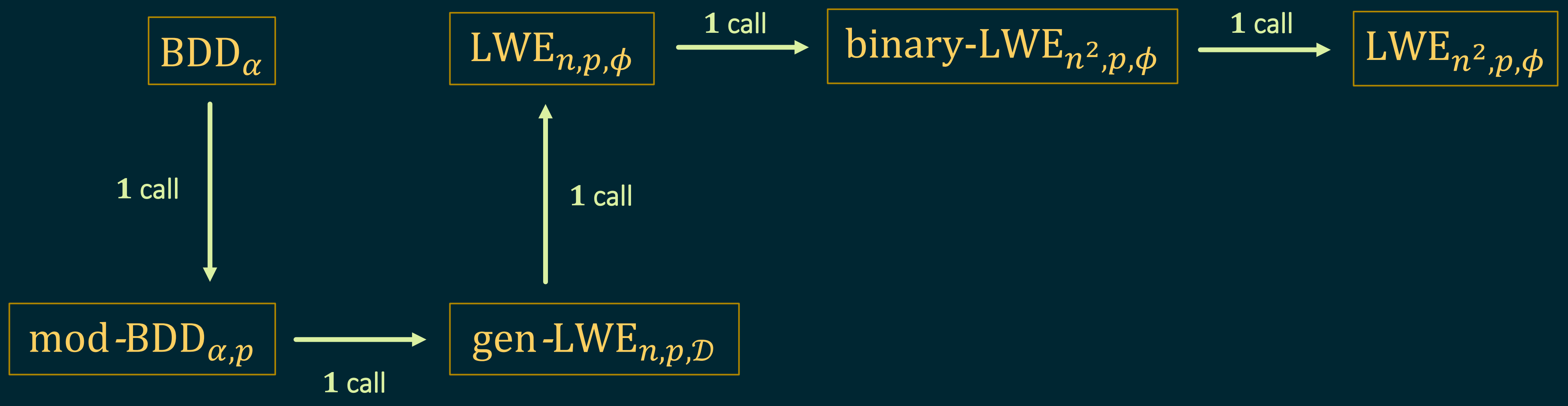
We want just $O(1)$ oracle calls to get a meaningful conclusion.

Our Reduction



We make a ***single*** *oracle call in each step* and suffer at most a polynomial loss in success probability.

Our Reduction



We use the same techniques as [Regev, 2005] and [Brakerski+, 2013],
but with great care to the *explicit loss in success probability* and *number of oracle calls*.

Our Main Result

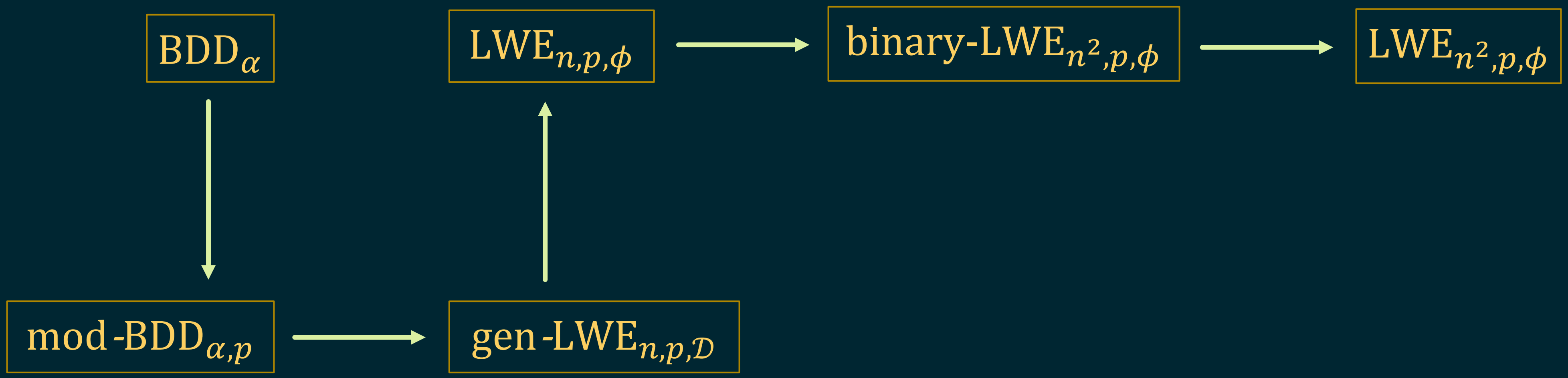
Theorem 1: *(informal)* If no PPT algorithm can solve BDD_{α} for $\alpha \in (0, 1/2)$

with success probability greater than $2^{-\Omega\left(\frac{n^2}{\log n}\right)}$,

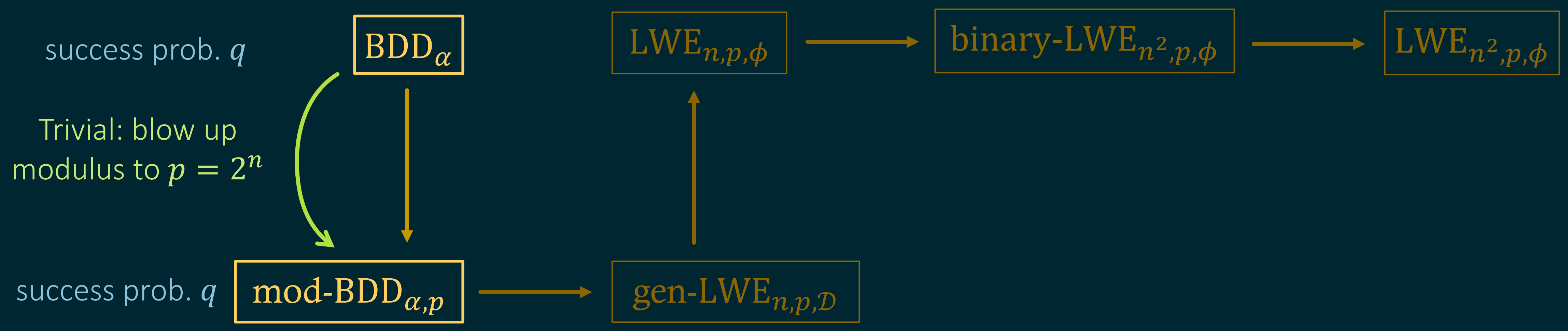
then no PPT algorithm can solve search-$\mathrm{LWE}_{n,p,\phi}$ (even for binary secret)

for dimension n, and modulus $p = \mathrm{poly}(n)$ with success probability $2^{-\frac{n}{\log n}}$.

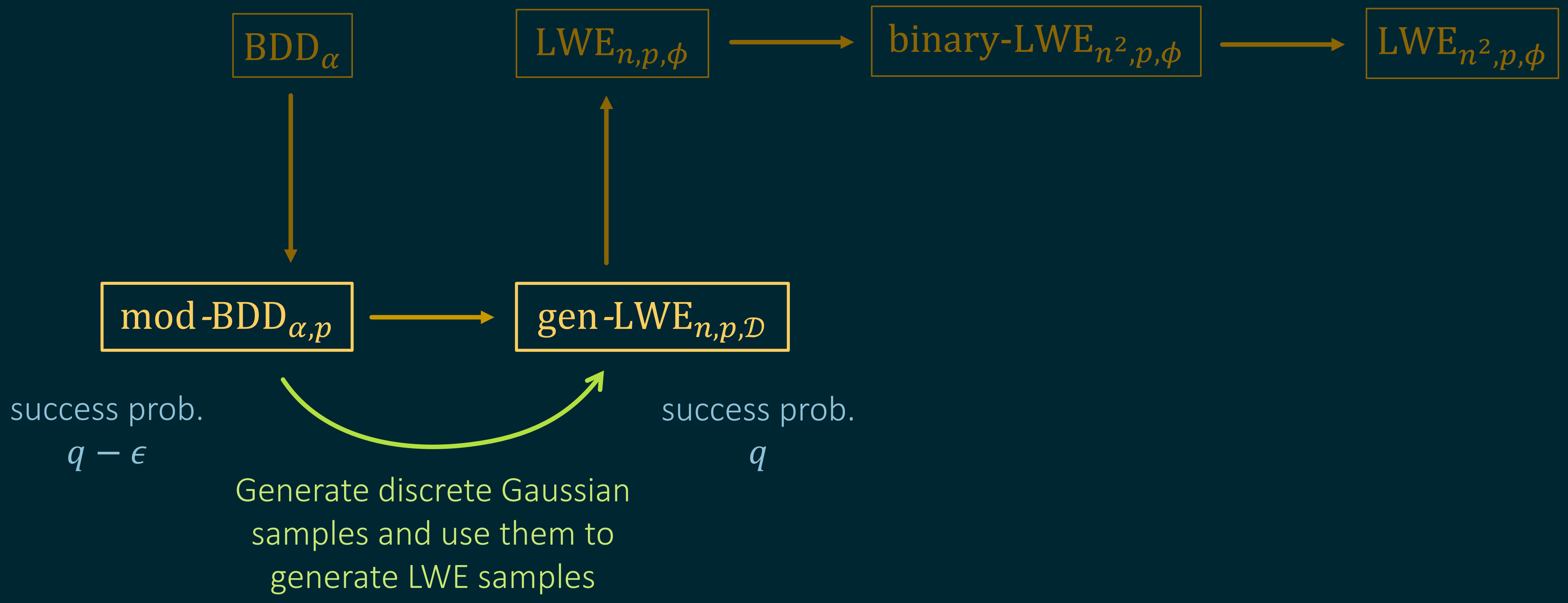
Our Reduction



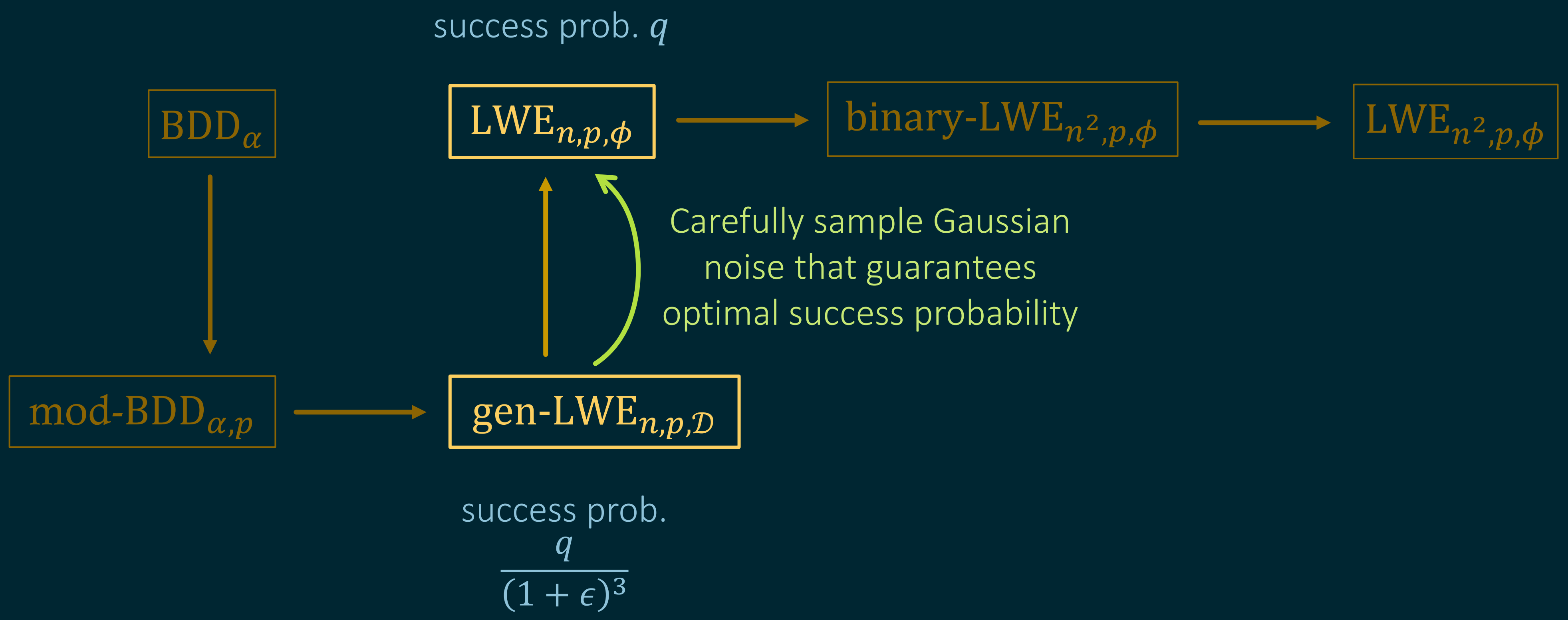
Our Proof Techniques



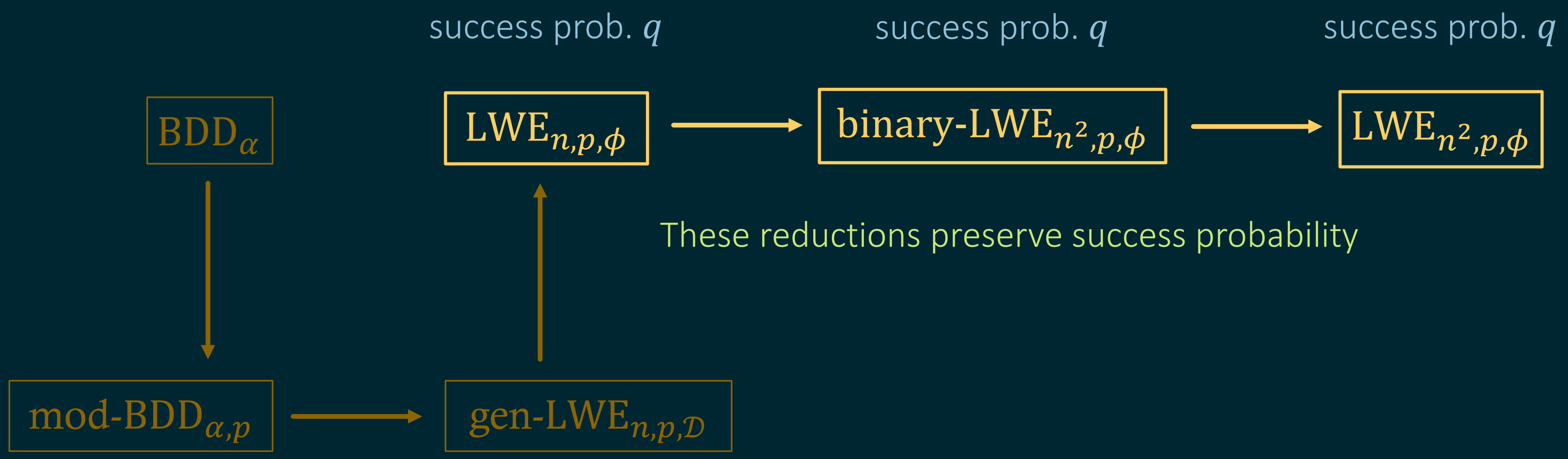
Our Proof Techniques



Our Proof Techniques



Our Proof Techniques



Our Second Result

Theorem 2: *(informal)* If no algorithm can solve search-$\mathrm{LWE}_{n,p}$ for polynomial modulus

with success probability $\boldsymbol{\alpha}$ in *expected* polynomial time,

then no PPT algorithm can "solve" decision-$\mathrm{LWE}_{n,p}$

with probability $\approx \boldsymbol{\alpha}$.

Future Directions

- Establish a similar result for $\mathrm{GapSVP} \to \mathrm{BDD}$ (or prove impossibility).
- Reductions $\mathrm{BDD} \to$ search-LWE and search-$\mathrm{LWE} \to$ decision-LWE are disconnected, because *expected* polynomial-time is a fundamental part of the second reduction.

  Is a workaround possible?
- Use this alternative framework to study the complexity of other computational problems relevant to cryptography or learning.

Future Directions

- Establish a similar result for GapSVP → BDD (or prove impossibility).
- Reductions BDD → search-LWE and search-LWE → decision-LWE are disconnected, because *expected* polynomial-time is a fundamental part of the second reduction. Is a workaround possible?
- Use this alternative framework to study the complexity of other computational problems relevant to cryptography or learning.

Thank you! Questions?