On the Success Probability of Solving Unique SVP via BKZ

Eamonn W. Postlethwaite, Fernando Virdia

Information Security Group,
Royal Holloway, University of London,
United Kingdom

PKC 2021

Overview

We investigate the cost of solving Learning With Errors (LWE) via lattice reduction.

Overview

We investigate the cost of solving Learning With Errors (LWE) via lattice reduction.

Contributions

- Extend [ADPS16, DDGR20] to estimate the cost of low-probability attacks.

Overview

We investigate the cost of solving Learning With Errors (LWE) via lattice reduction.

Contributions

- Extend [ADPS16, DDGR20] to estimate the cost of low-probability attacks.
- Experimentally verify the resulting model against Gaussian, binary and ternary secrets and errors.

Overview

We investigate the cost of solving Learning With Errors (LWE) via lattice reduction.

Contributions

- Extend [ADPS16, DDGR20] to estimate the cost of low-probability attacks.
- Experimentally verify the resulting model against Gaussian, binary and ternary secrets and errors.
- Explain low-probability attacks reported in [AGVW17].

Overview

We investigate the cost of solving Learning With Errors (LWE) via lattice reduction.

Contributions

- Extend [ADPS16, DDGR20] to estimate the cost of low-probability attacks.
- Experimentally verify the resulting model against Gaussian, binary and ternary secrets and errors.
- Explain low-probability attacks reported in [AGVW17].
- Determine effect on security estimates for cryptosystems.

Overview

We investigate the cost of solving Learning With Errors (LWE) via lattice reduction.

Contributions

- Extend [ADPS16, DDGR20] to estimate the cost of low-probability attacks.
- Experimentally verify the resulting model against Gaussian, binary and ternary secrets and errors.
- Explain low-probability attacks reported in [AGVW17].
- Determine effect on security estimates for cryptosystems.

Preliminaries

Search-LWE

Let $\boldsymbol{s} \leftarrow \chi_s^n$, $\boldsymbol{e} \leftarrow \chi_e^m$ and $\boldsymbol{A} \leftarrow U(\mathbb{Z}_q^{m \times n})$. Given $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \bmod q)$, recover $\boldsymbol{s}$.

Preliminaries

Search-LWE

Let $\boldsymbol{s} \leftarrow \chi_s^n$, $\boldsymbol{e} \leftarrow \chi_e^m$ and $\boldsymbol{A} \leftarrow U(\mathbb{Z}_q^{m \times n})$. Given $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \bmod q)$, recover $\boldsymbol{s}$.

Primal attack strategy

Construct a lattice containing $\boldsymbol{t} := (\boldsymbol{s}, \boldsymbol{e}, 1)$ as the unique shortest vector. Use lattice reduction to recover $\boldsymbol{t}$. This reduces LWE to the unique Shortest Vector Problem (uSVP).

Preliminaries

Search-LWE

Let $\boldsymbol{s} \leftarrow \chi_s^n$, $\boldsymbol{e} \leftarrow \chi_e^m$ and $\boldsymbol{A} \leftarrow U(\mathbb{Z}_q^{m \times n})$. Given $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \bmod q)$, recover $\boldsymbol{s}$.

Primal attack strategy

Construct a lattice containing $\boldsymbol{t} := (\boldsymbol{s}, \boldsymbol{e}, 1)$ as the unique shortest vector. Use lattice reduction to recover $\boldsymbol{t}$. This reduces LWE to the unique Shortest Vector Problem (uSVP).

Immediate questions:

- How do we cost the attack?
- How does the cost vary if we change χ_e, χ_s?

Preliminaries

Lattice basis

Given d linearly independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d \in \mathbb{R}^d$, we say they form a basis for the (full-rank) lattice $\Lambda = \{\sum x_i \boldsymbol{b}_i : x_i \in \mathbb{Z}\}$.

Basis profile

Given a basis $\boldsymbol{B}$ of a lattice, we define the profile of $\boldsymbol{B}$ as the set $\{\|\boldsymbol{b}_i^*\|^2\}_{i=1}^d$ of its orthogonal Gram–Schmidt vectors.

Orthogonal projections

Given a basis $\boldsymbol{B}$ of $\mathbb{R}^d$ we denote by $\pi_{\boldsymbol{B},k} : \mathbb{R}^d \to \mathbb{R}^d$ the linear operator projecting vectors orthogonally to the subspace $\operatorname{span}_{\mathbb{R}}(\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{k-1}\})$.

Preliminaries

Given a lattice basis and a reduction algorithm $\mathcal{A}$, it is possible to predict the resulting reduced basis profile.

Preliminaries

Given a lattice basis and a reduction algorithm $\mathcal{A}$, it is possible to predict the resulting reduced basis profile.

- The Geometric Series Assumption (GSA) [Sch03] predicts that

$$\|\boldsymbol{b}_i^*\| = \alpha \cdot \|\boldsymbol{b}_{i-1}^*\| \ \forall i, \text{ for some } \alpha = \alpha(\mathcal{A}) \in (0,1).$$

Preliminaries

Given a lattice basis and a reduction algorithm $\mathcal{A}$, it is possible to predict the resulting reduced basis profile.

- The Geometric Series Assumption (GSA) [Sch03] predicts that
$$\|\boldsymbol{b}_i^*\| = \alpha \cdot \|\boldsymbol{b}_{i-1}^*\| \ \forall i, \text{ for some } \alpha = \alpha(\mathcal{A}) \in (0,1).$$
- BKZ simulators (such as [CN11]) can predict the effect of running τ tours of BKZ on a given basis.

Preliminaries

Given a lattice basis and a reduction algorithm $\mathcal{A}$, it is possible to predict the resulting reduced basis profile.

- The Geometric Series Assumption (GSA) [Sch03] predicts that
$$\|\boldsymbol{b}_i^*\| = \alpha \cdot \|\boldsymbol{b}_{i-1}^*\| \ \forall i, \text{ for some } \alpha = \alpha(\mathcal{A}) \in (0,1).$$
- BKZ simulators (such as [CN11]) can predict the effect of running τ tours of BKZ on a given basis.

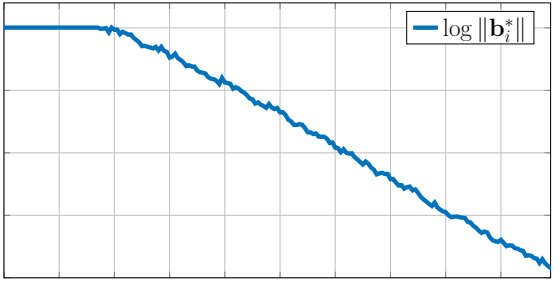
We now look at the main step in the lattice reduction algorithms we will use.

The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 1$
  $j \leftarrow \min(i + \beta - 1, d)$
  $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
    $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
    LLL on $\boldsymbol{B}$ to remove linear dependencies
  return *whether an insertion was made*

Lattice basis profile



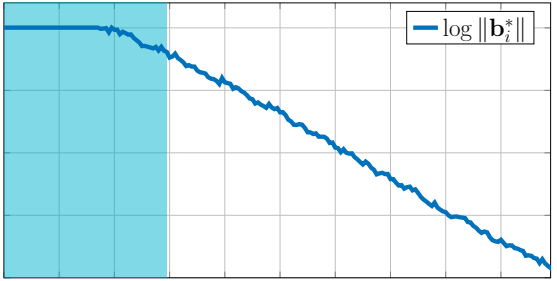
The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 1$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies
- **return** *whether an insertion was made*

Lattice basis profile



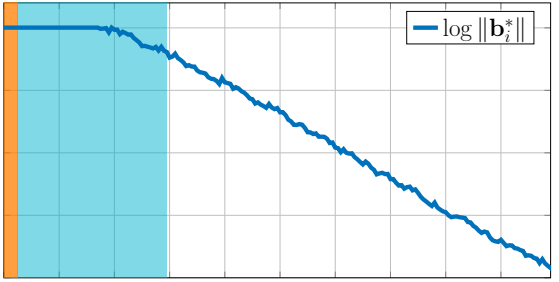
The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 1$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies
- **return** *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 11$

$j \leftarrow \min(i + \beta - 1, d)$

$\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathsf{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

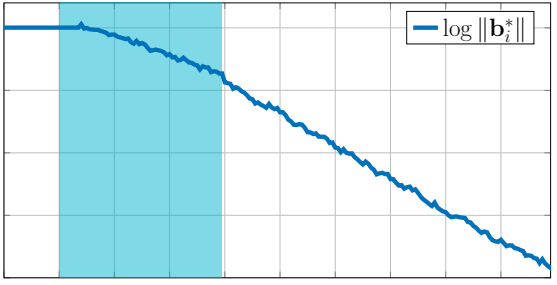
$\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$

extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i

LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



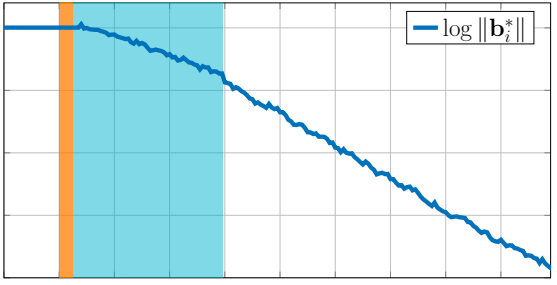
The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 11$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies
- **return** *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

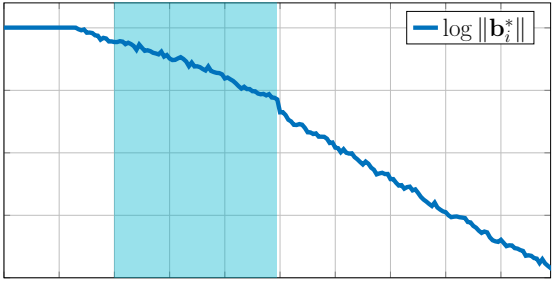
BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 21$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathsf{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 21$

$j \leftarrow \min(i + \beta - 1, d)$

$\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

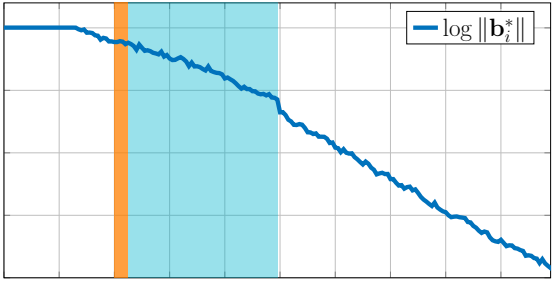
$\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$

extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i

LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 31$

$j \leftarrow \min(i + \beta - 1, d)$

$\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

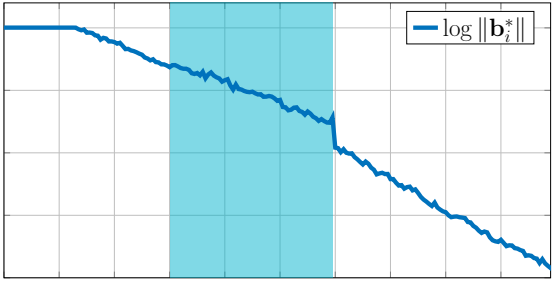
$\boldsymbol{v}' \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$

extend $\boldsymbol{B}$ by inserting $\boldsymbol{v}'$ into $\boldsymbol{B}$ at index i

LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



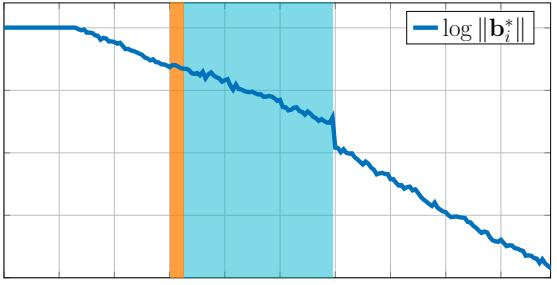
The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 31$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies
- **return** *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

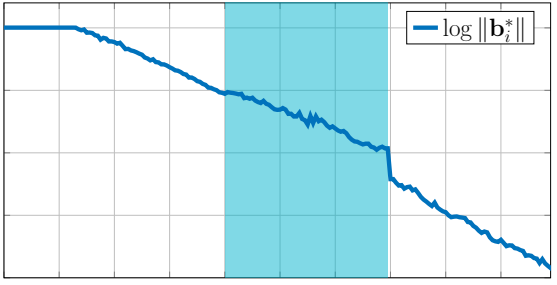
BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 41$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathsf{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 41$

  $j \leftarrow \min(i + \beta - 1, d)$

  $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

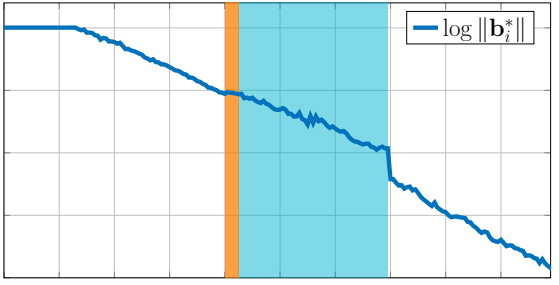
    $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$

    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i

    LLL on $\boldsymbol{B}$ to remove linear dependencies

  return *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

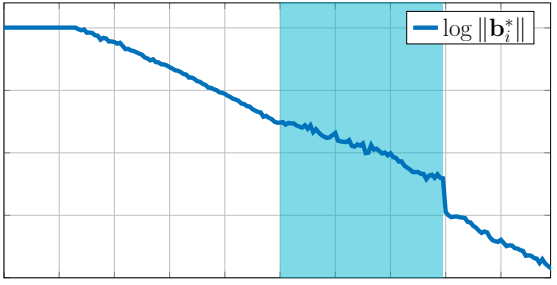
BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 51$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v}' \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v}'$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



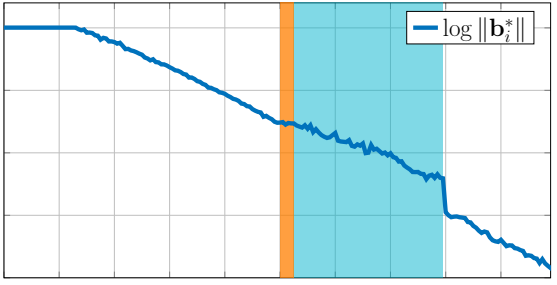
The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 51$

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies
- **return** *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 61$

$j \leftarrow \min(i + \beta - 1, d)$

$\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

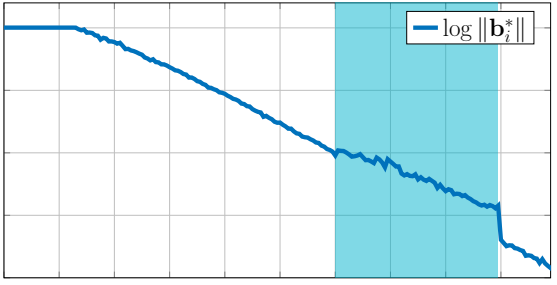
$\boldsymbol{v}' \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$

extend $\boldsymbol{B}$ by inserting $\boldsymbol{v}'$ into $\boldsymbol{B}$ at index i

LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // $i = 61$

$j \leftarrow \min(i + \beta - 1, d)$

$\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

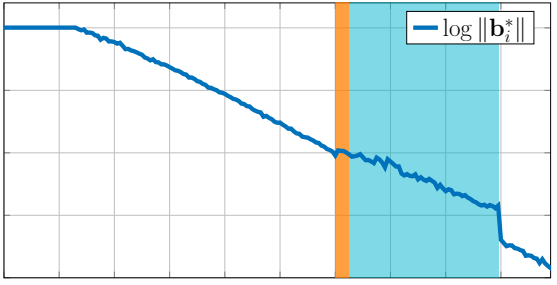
$\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$

extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i

LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile



The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // 71

$j \leftarrow \min(i + \beta - 1, d)$

$\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathsf{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$

if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**

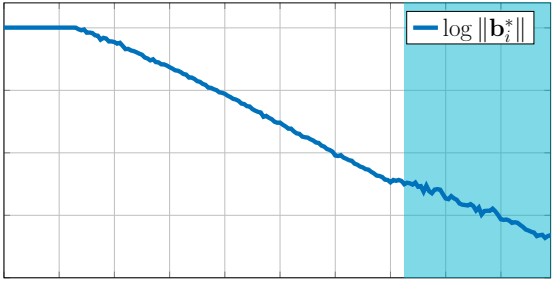
$\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$

extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i

LLL on $\boldsymbol{B}$ to remove linear dependencies

return *whether an insertion was made*

Lattice basis profile

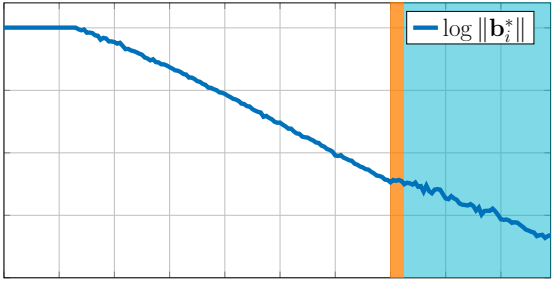


The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // 71
  $j \leftarrow \min(i + \beta - 1, d)$
  $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
    $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
    LLL on $\boldsymbol{B}$ to remove linear dependencies
  return *whether an insertion was made*

Lattice basis profile

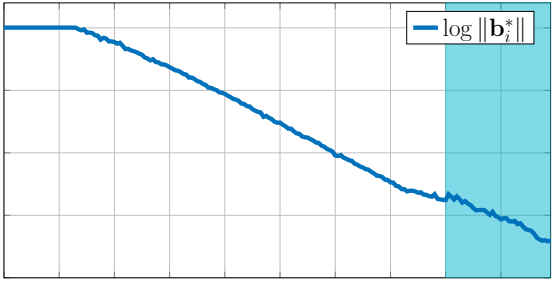


The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // 81
  $j \leftarrow \min(i + \beta - 1, d)$
  $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathsf{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
    $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
    LLL on $\boldsymbol{B}$ to remove linear dependencies
  return *whether an insertion was made*

Lattice basis profile

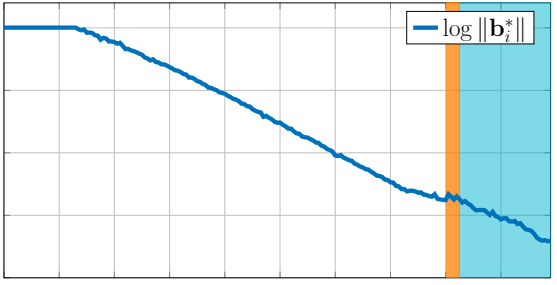


The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // 81
  $j \leftarrow \min(i + \beta - 1, d)$
  $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
    $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
    LLL on $\boldsymbol{B}$ to remove linear dependencies
  return *whether an insertion was made*

Lattice basis profile



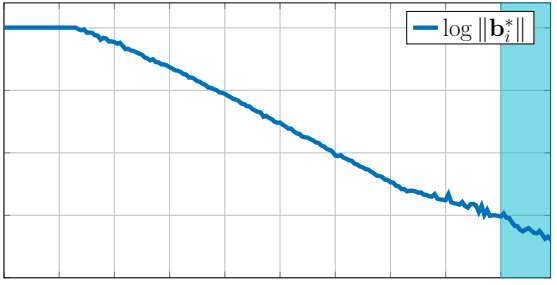
The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // 91

- $j \leftarrow \min(i + \beta - 1, d)$
- $\boldsymbol{v} = x_i\pi_i(\boldsymbol{b}_i) + \cdots + x_j\pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathsf{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
- **if** $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
  - $\boldsymbol{v'} \leftarrow x_i\boldsymbol{b}_i + \cdots + x_j\boldsymbol{b}_j$
  - extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
  - LLL on $\boldsymbol{B}$ to remove linear dependencies
- **return** *whether an insertion was made*

Lattice basis profile

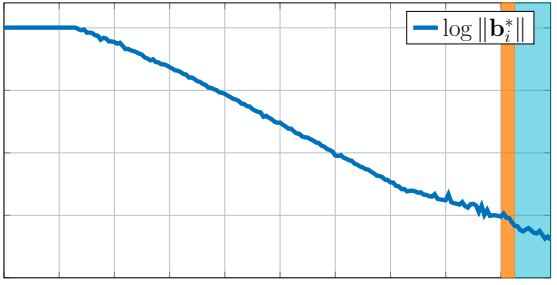


The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do** // 91
  $j \leftarrow \min(i + \beta - 1, d)$
  $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
    $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
    LLL on $\boldsymbol{B}$ to remove linear dependencies
  return *whether an insertion was made*

Lattice basis profile

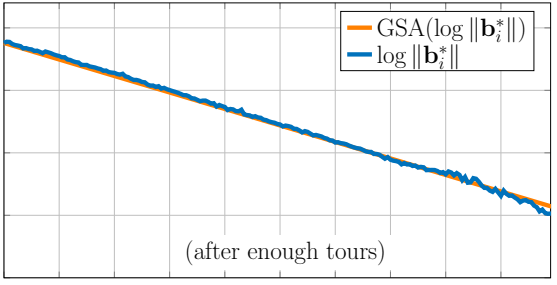


The BKZ-β tour

BKZ-β tour

for $i \leftarrow 1$ **to** d **do**
  $j \leftarrow \min(i + \beta - 1, d)$
  $\boldsymbol{v} = x_i \pi_i(\boldsymbol{b}_i) + \cdots + x_j \pi_i(\boldsymbol{b}_j) \leftarrow O_{\mathrm{SVP}}(\pi_i(\boldsymbol{B}[i:j]))$
  if $\|\boldsymbol{v}\| < \|\boldsymbol{b}_i^*\|$ **then**
    $\boldsymbol{v'} \leftarrow x_i \boldsymbol{b}_i + \cdots + x_j \boldsymbol{b}_j$
    extend $\boldsymbol{B}$ by inserting $\boldsymbol{v'}$ into $\boldsymbol{B}$ at index i
    LLL on $\boldsymbol{B}$ to remove linear dependencies
  return *whether an insertion was made*

Lattice basis profile



Block-wise lattice reduction

From the BKZ-β tour, we can define two algorithms.

Block-wise lattice reduction

From the BKZ-β tour, we can define two algorithms.

- BKZ-β [SE91] runs τ tours with block size β.

BKZ-β

Input: $\tau \in \mathbb{Z}_+$
repeat τ *times*
  run a BKZ-β tour
  if *no changes made to the basis* **then**
    return

Block-wise lattice reduction

From the BKZ-β tour, we can define two algorithms.

- BKZ-β [SE91] runs τ tours with block size β.
- Progressive BKZ (PBKZ) [AWHT16] runs BKZ tours with increasing block size, with τ tours for each block size.

BKZ-β

Input: $\tau \in \mathbb{Z}_+$
repeat τ *times*
  run a BKZ-β tour
  if *no changes made to the basis* **then**
    return

Progressive BKZ (PBKZ)

Input: $\tau \in \mathbb{Z}_+$
for $\beta = 3 \dots d$ **do**
  run τ tours of BKZ-β

Block-wise lattice reduction

From the BKZ-β tour, we can define two algorithms.

- BKZ-β [SE91] runs τ tours with block size β.
- Progressive BKZ (PBKZ) [AWHT16] runs BKZ tours with increasing block size, with τ tours for each block size.

BKZ-β

Input: $\tau \in \mathbb{Z}_+$
repeat τ *times*
  run a BKZ-β tour
  if *no changes made to the basis* **then**
    return

Progressive BKZ (PBKZ)

Input: $\tau \in \mathbb{Z}_+$
for $\beta = 3 \dots d$ **do**
  run τ tours of BKZ-β

At the end of each tour we can check if a solution to LWE was found.

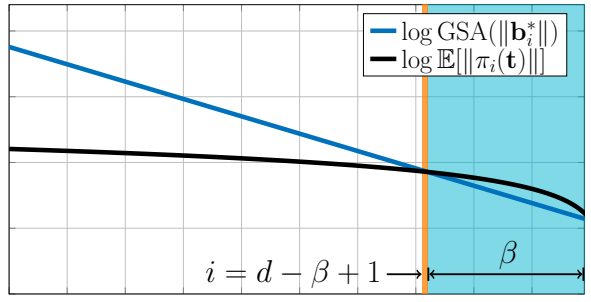
The cost of LWE

- Objective: recover $\boldsymbol{t} = (\boldsymbol{s}, \boldsymbol{e}, 1) \in \Lambda$.
- In practice, recovery of $\boldsymbol{t}$ follows from recovery of $\pi_{d-\beta+1}(\boldsymbol{t})$.

The cost of LWE

- Objective: recover $\boldsymbol{t} = (\boldsymbol{s}, \boldsymbol{e}, 1) \in \Lambda$.
- In practice, recovery of $\boldsymbol{t}$ follows from recovery of $\pi_{d-\beta+1}(\boldsymbol{t})$.
- [ADPS16]: choose smallest β^* s.t. $\|\pi_{d-\beta^*+1}(\boldsymbol{t})\| \leq \mathrm{GSA}(\|\boldsymbol{b}^*_{d-\beta^*+1}\|)$.

[ADPS16] success condition



- [ADPS16]'s approach was originally experimentally verified in [AGVW17].
- Success probabilities near 1 using BKZ-β^*, but stay high also for smaller block sizes $\beta^* - h$.
- This could be an issue if h grows with n.

n	65	80	100
β^*	93.3%	94.2%	88.8%
β^*-5	52.8%	60.6%	39.6%
β^*-10	4.8%	8.9%	5.8%
β^*-15	—	0.2%	0.2%

Table: Measured success probability for solving Search-LWE with secret dimension n, using block size β, as reported in [AGVW17].

- [ADPS16]'s approach was originally experimentally verified in [AGVW17].
- Success probabilities near 1 using BKZ-β^*, but stay high also for smaller block sizes $\beta^* - h$.
- This could be an issue if h grows with n.

n	65	80	100
β^*	93.3%	94.2%	88.8%
β^*-5	52.8%	60.6%	39.6%
β^*-10	4.8%	8.9%	5.8%
β^*-15	—	0.2%	0.2%

Table: Measured success probability for solving Search-LWE with secret dimension n, using block size β, as reported in [AGVW17].

Our main contribution

We extend [ADPS16, DDGR20]'s approach to predict successes for $\beta < \beta^*$.

We extend [DDGR20] to simulate the probability of recovering $\pi_{d-\beta+1}(\boldsymbol{t})$.

Estimating succcess probability of BKZ-β

Input: (n, q, χ, m), β, τ
$p_{\text{tot}} \leftarrow 0, \sigma^2 \leftarrow \mathbb{V}(\chi)$
$d \leftarrow n + m + 1$
for *tour* $\leftarrow 1$ **to** τ **do**
  $\texttt{prof} \leftarrow \text{BKZSim}((n, q, \chi, m), \beta, \text{tour})$
  $p_{\text{new}} \leftarrow P[x \leftarrow \sigma^2 \chi_\beta^2 : x \leq \texttt{prof}[d-\beta+1]]$
  $p_{\text{tot}} \leftarrow p_{\text{tot}} + (1 - p_{\text{tot}}) \cdot p_{\text{new}}$
return p_{tot}

We extend [DDGR20] to simulate the probability of recovering $\pi_{d-\beta+1}(\boldsymbol{t})$.

- Use a basis profile simulator 'BKZSim' (such as [CN11]) to keep track of the basis profile as BKZ tours happen.

Estimating succcess probability of BKZ-β

Input: (n, q, χ, m), β, τ
$p_{\text{tot}} \leftarrow 0, \sigma^2 \leftarrow \mathbb{V}(\chi)$
$d \leftarrow n + m + 1$
for *tour* $\leftarrow 1$ **to** τ **do**
  $\texttt{prof} \leftarrow \text{BKZSim}((n, q, \chi, m), \beta, \text{tour})$
  $p_{\text{new}} \leftarrow P[x \leftarrow \sigma^2 \chi_\beta^2 : x \leq \texttt{prof}[d-\beta+1]]$
  $p_{\text{tot}} \leftarrow p_{\text{tot}} + (1 - p_{\text{tot}}) \cdot p_{\text{new}}$
return p_{tot}

We extend [DDGR20] to simulate the probability of recovering $\pi_{d-\beta+1}(\boldsymbol{t})$.

- Use a basis profile simulator 'BKZSim' (such as [CN11]) to keep track of the basis profile as BKZ tours happen.
- In each "tour", the probability of recovering $\boldsymbol{t}$ is $P[\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2 \leq \|\boldsymbol{b}^*_{d-\beta+1}\|^2]$, where we model $\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2 \sim \sigma^2 \cdot \chi^2_\beta$.

Estimating succcess probability of BKZ-β

Input: (n, q, χ, m), β, τ
$p_{\text{tot}} \leftarrow 0, \sigma^2 \leftarrow \mathbb{V}(\chi)$
$d \leftarrow n + m + 1$
for *tour* $\leftarrow 1$ **to** τ **do**
  $\texttt{prof} \leftarrow \text{BKZSim}((n, q, \chi, m), \beta, \text{tour})$
  $p_{\text{new}} \leftarrow P[x \leftarrow \sigma^2 \chi^2_\beta : x \leq \texttt{prof}[d-\beta+1]]$
  $p_{\text{tot}} \leftarrow p_{\text{tot}} + (1 - p_{\text{tot}}) \cdot p_{\text{new}}$
return p_{tot}

We extend [DDGR20] to simulate the probability of recovering $\pi_{d-\beta+1}(\boldsymbol{t})$.

- Use a basis profile simulator 'BKZSim' (such as [CN11]) to keep track of the basis profile as BKZ tours happen.
- In each "tour", the probability of recovering $\boldsymbol{t}$ is $P[\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2 \leq \|\boldsymbol{b}^*_{d-\beta+1}\|^2]$, where we model $\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2 \sim \sigma^2 \cdot \chi^2_\beta$.
- Accumulate success probabilities as tours progress.[1]

Estimating succcess probability of BKZ-β

Input: (n, q, χ, m), β, τ
$p_{\text{tot}} \leftarrow 0, \sigma^2 \leftarrow \mathbb{V}(\chi)$
$d \leftarrow n + m + 1$
for *tour* $\leftarrow 1$ **to** τ **do**
  $\texttt{prof} \leftarrow \text{BKZSim}((n, q, \chi, m), \beta, \text{tour})$
  $p_{\text{new}} \leftarrow P[x \leftarrow \sigma^2 \chi^2_\beta : x \leq \texttt{prof}[d-\beta+1]]$
  $p_{\text{tot}} \leftarrow p_{\text{tot}} + (1 - p_{\text{tot}}) \cdot p_{\text{new}}$
return p_{tot}

[1] We model the win events as independent, since tours re-randomise the basis (until it is reduced).

We extend [DDGR20] to simulate the probability of recovering $\pi_{d-\beta+1}(\boldsymbol{t})$.

- Use a basis profile simulator 'BKZSim' (such as [CN11]) to keep track of the basis profile as BKZ tours happen.
- In each "tour", the probability of recovering $\boldsymbol{t}$ is $P[\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2 \leq \|\boldsymbol{b}^*_{d-\beta+1}\|^2]$, where we model $\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2 \sim \sigma^2 \cdot \chi^2_\beta$.
- Accumulate success probabilities as tours progress.[1]

Estimating succcess probability of BKZ-β

Input: (n, q, χ, m), β, τ
$p_{\text{tot}} \leftarrow 0, \sigma^2 \leftarrow \mathbb{V}(\chi)$
$d \leftarrow n + m + 1$
for *tour* $\leftarrow 1$ **to** τ **do**
  $\texttt{prof} \leftarrow \text{BKZSim}((n, q, \chi, m), \beta, \text{tour})$
  $p_{\text{new}} \leftarrow P[x \leftarrow \sigma^2\chi^2_\beta : x \leq \texttt{prof}[d-\beta+1]]$
  $p_{\text{tot}} \leftarrow p_{\text{tot}} + (1 - p_{\text{tot}}) \cdot p_{\text{new}}$
return p_{tot}

For PBKZ: same idea, but increase β after τ tours until success probability ≈ 1.

[1] We model the win events as independent, since tours re-randomise the basis (until it is reduced).

Experimental verification

To test our success probability estimations we chose three sets of parameters estimated to need $\beta^* \approx 60$.

n	q	σ	β^*
72	97	1	61
93	257	1	61
100	257	$\sqrt{2/3}$	60

Table: LWE parameters, σ is the standard deviation for a discr. Gaussian distribution.

Experimental verification

To test our success probability estimations we chose three sets of parameters estimated to need $\beta^* \approx 60$.

Two batches

- Discrete Gaussian error and secret.
- Binary (resp. ternary) error and secret for $n < 100$ (resp. $n = 100$).

The simulator *does not* receive the distribution as input, just its variance.

n	q	σ	β^*
72	97	1	61
93	257	1	61
100	257	$\sqrt{2/3}$	60

Table: LWE parameters, σ is the standard deviation for a discr. Gaussian distribution.

Experimental verification

To test our success probability estimations we chose three sets of parameters estimated to need $\beta^* \approx 60$.

Two batches

- Discrete Gaussian error and secret.
- Binary (resp. ternary) error and secret for $n < 100$ (resp. $n = 100$).

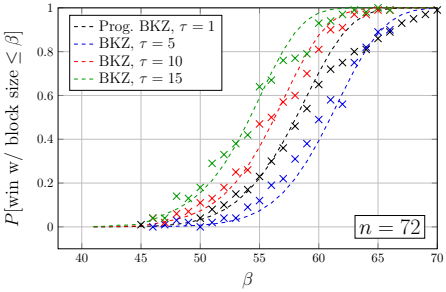
n	q	σ	β^*
72	97	1	61
93	257	1	61
100	257	$\sqrt{2/3}$	60

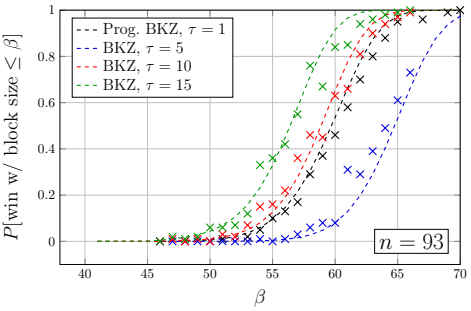
Table: LWE parameters, σ is the standard deviation for a discr. Gaussian distribution.

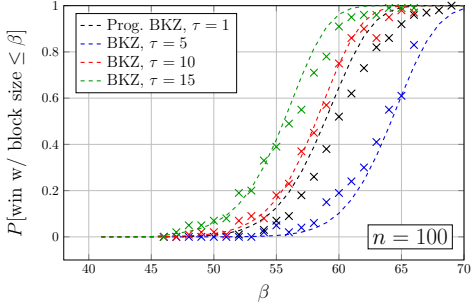
The simulator *does not* receive the distribution as input, just its variance.

We run many variants of these experiments. We now look at some results, more can be found in the paper.

Gaussian $\boldsymbol{e}$ and $\boldsymbol{s}$, BKZ and PBKZ



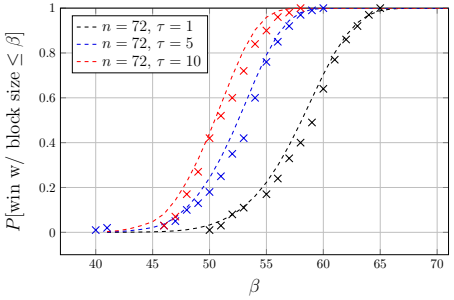


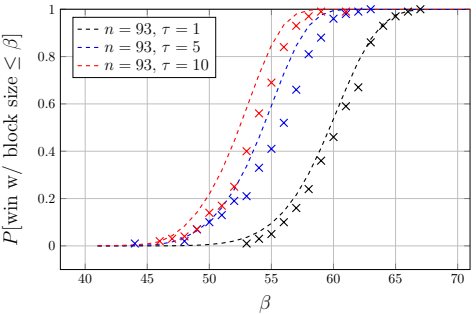


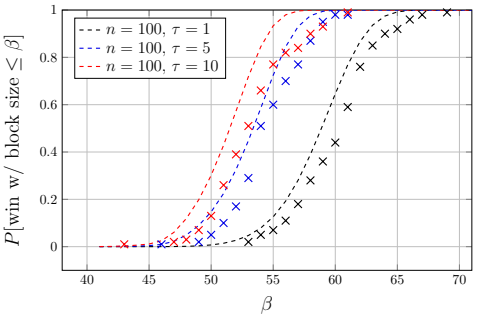
Discrete Gaussian experiments, PBKZ and various tours of BKZ. All discrete Gaussian errors.

Dashed lines are simulations, crosses experiments.

Binary/Ternary $\boldsymbol{e}$ and $\boldsymbol{s}$, PBKZ







Progressive BKZ experiments and simulations for binary and ternary secrets.

No changes are made the simulator.

The effect of sample variance

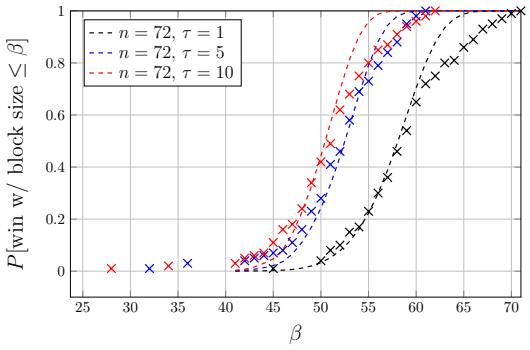


Figure: Gaussian $\boldsymbol{s}$ and $\boldsymbol{e}$ instance, reduced using PBKZ.

The effect of sample variance

- Let $\boldsymbol{t} = (t_1, \ldots, t_d)$ with each $t_i \xleftarrow{iid} D$.
- Let $\bar{t} = \frac{1}{d}\sum_{i=1}^{d} t_i$ be the sample mean.
- Then $s^2 = \frac{1}{d}\sum_{i=1}^{d} (t_i - \bar{t})^2$ is the sample variance.

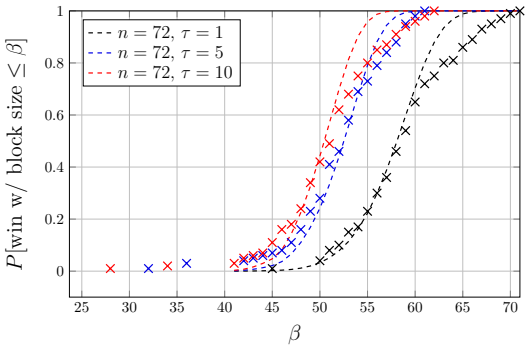


Figure: Gaussian $\boldsymbol{s}$ and $\boldsymbol{e}$ instance, reduced using PBKZ.

The effect of sample variance

- Let $\boldsymbol{t} = (t_1, \ldots, t_d)$ with each $t_i \xleftarrow{iid} D$.
- Let $\bar{t} = \frac{1}{d}\sum_{i=1}^{d} t_i$ be the sample mean.
- Then $s^2 = \frac{1}{d}\sum_{i=1}^{d} (t_i - \bar{t})^2$ is the sample variance.

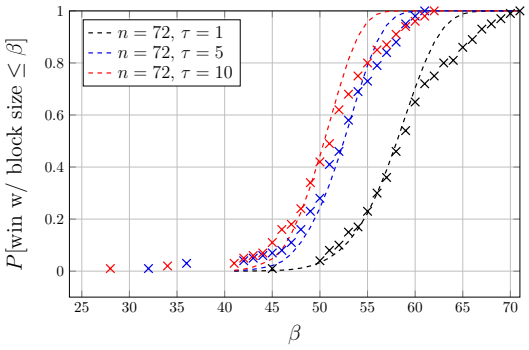
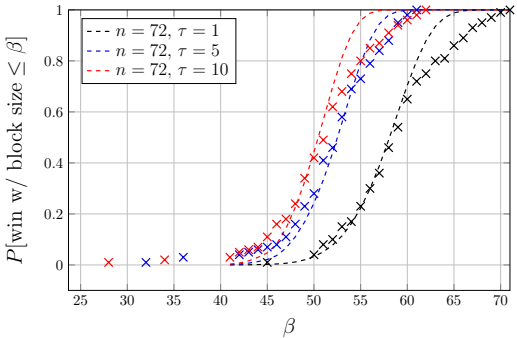
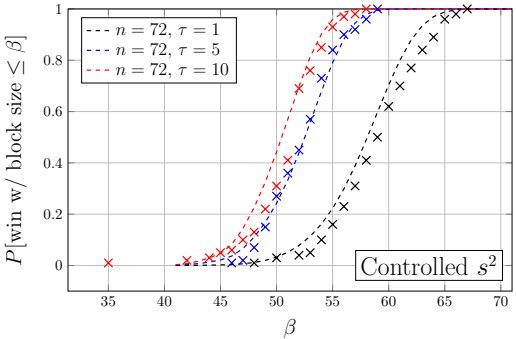


Figure: Gaussian $\boldsymbol{s}$ and $\boldsymbol{e}$ instance, reduced using PBKZ.

Given $\boldsymbol{t}$, s^2 affects the norms $\|\pi_{d-\beta+1}(\boldsymbol{t})\|^2$, not σ^2. Simulations are off when $s^2 \not\approx \sigma^2$.

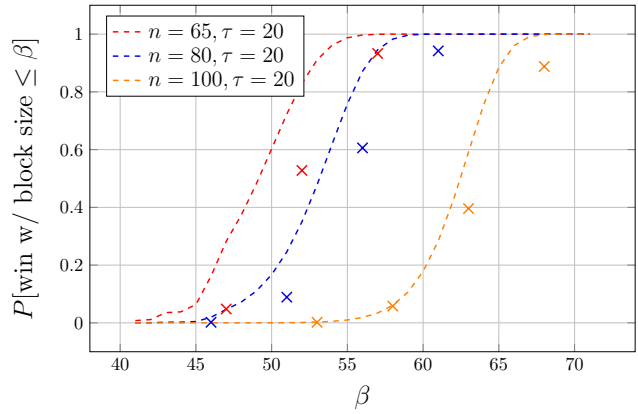
Controlling s^2, Gaussian $\boldsymbol{e}$ and $\boldsymbol{s}$, PBKZ





- On the right we control the sample variance to be within 2% of σ^2.
- Should not affect crypto instances: $\mathbb{V}(s^2) \in O(1/d)$.

Finally, we use the uSVP simulator to explain the [AGVW17] success probabilities.



Our predictions for [AGVW17]'s parameters seem to match their reported probabilities.

- The uSVP simulators allow us to predict success probabilities for $\beta < \beta^*$.
- What is the impact on cryptographic parameters? Consider β a random variable.

	BKZ 2.0, $\tau = 15$			PBKZ, $\tau = 1$		PBKZ, $\tau = 5$	
scheme	β^*	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$
Kyber 512	381	386.06	2.56	389.53	2.88	385.70	2.32
Kyber 768	623	634.41	2.96	638.23	3.30	634.00	2.66
Kyber 1024	873	891.13	3.31	895.24	3.66	890.63	2.96
LightSaber	404	408.81	2.65	412.24	2.96	408.35	2.39
Saber	648	659.36	3.00	663.10	3.32	658.85	2.68
FireSaber	890	907.76	3.34	911.78	3.68	907.16	2.97
ntruhps2048509	374	375.93	2.58	379.56	2.92	375.71	2.36
ntruhps2048677	521	522.78	2.82	526.77	3.18	522.67	2.57
ntruhps4096821	621	628.78	2.83	632.54	3.17	628.43	2.55
ntruhrss701	471	477.20	2.48	480.51	2.77	476.72	2.23

Table: Cryptographic parameters for NIST PQC finalists, using 2nd round parameters.

- The uSVP simulators allow us to predict success probabilities for $\beta < \beta^*$.
- What is the impact on cryptographic parameters? Consider β a random variable.

		BKZ 2.0, $\tau = 15$		PBKZ, $\tau = 1$		PBKZ, $\tau = 5$	
scheme	β^*	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$
Kyber 512	381	386.06	2.56	389.53	2.88	385.70	2.32
Kyber 768	623	634.41	2.96	638.23	3.30	634.00	2.66
Kyber 1024	873	891.13	3.31	895.24	3.66	890.63	2.96
LightSaber	404	408.81	2.65	412.24	2.96	408.35	2.39
Saber	648	659.36	3.00	663.10	3.32	658.85	2.68
FireSaber	890	907.76	3.34	911.78	3.68	907.16	2.97
ntruhps2048509	374	375.93	2.58	379.56	2.92	375.71	2.36
ntruhps2048677	521	522.78	2.82	526.77	3.18	522.67	2.57
ntruhps4096821	621	628.78	2.83	632.54	3.17	628.43	2.55
ntruhrss701	471	477.20	2.48	480.51	2.77	476.72	2.23

Table: Cryptographic parameters for NIST PQC finalists, using 2nd round parameters.

- The uSVP simulators allow us to predict success probabilities for $\beta < \beta^*$.
- What is the impact on cryptographic parameters? Consider β a random variable.

	BKZ 2.0, $\tau = 15$			PBKZ, $\tau = 1$		PBKZ, $\tau = 5$	
scheme	β^*	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$
Kyber 512	381	386.06	2.56	389.53	2.88	385.70	2.32
Kyber 768	623	634.41	2.96	638.23	3.30	634.00	2.66
Kyber 1024	873	891.13	3.31	895.24	3.66	890.63	2.96
LightSaber	404	408.81	2.65	412.24	2.96	408.35	2.39
Saber	648	659.36	3.00	663.10	3.32	658.85	2.68
FireSaber	890	907.76	3.34	911.78	3.68	907.16	2.97
ntruhps2048509	374	375.93	2.58	379.56	2.92	375.71	2.36
ntruhps2048677	521	522.78	2.82	526.77	3.18	522.67	2.57
ntruhps4096821	621	628.78	2.83	632.54	3.17	628.43	2.55
ntruhrss701	471	477.20	2.48	480.51	2.77	476.72	2.23

Table: Cryptographic parameters for NIST PQC finalists, using 2nd round parameters.

- The uSVP simulators allow us to predict success probabilities for $\beta < \beta^*$.
- What is the impact on cryptographic parameters? Consider β a random variable.

	BKZ 2.0, $\tau = 15$			PBKZ, $\tau = 1$		PBKZ, $\tau = 5$	
scheme	β^*	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$
Kyber 512	381	386.06	2.56	389.53	2.88	385.70	2.32
Kyber 768	623	634.41	2.96	638.23	3.30	634.00	2.66
Kyber 1024	873	891.13	3.31	895.24	3.66	890.63	2.96
LightSaber	404	408.81	2.65	412.24	2.96	408.35	2.39
Saber	648	659.36	3.00	663.10	3.32	658.85	2.68
FireSaber	890	907.76	3.34	911.78	3.68	907.16	2.97
ntruhps2048509	374	375.93	2.58	379.56	2.92	375.71	2.36
ntruhps2048677	521	522.78	2.82	526.77	3.18	522.67	2.57
ntruhps4096821	621	628.78	2.83	632.54	3.17	628.43	2.55
ntruhrss701	471	477.20	2.48	480.51	2.77	476.72	2.23

Table: Cryptographic parameters for NIST PQC finalists, using 2nd round parameters.

- The uSVP simulators allow us to predict success probabilities for $\beta < \beta^*$.
- What is the impact on cryptographic parameters? Consider β a random variable.

		BKZ 2.0, $\tau = 15$		PBKZ, $\tau = 1$		PBKZ, $\tau = 5$	
scheme	β^*	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$	$\mathbb{E}(\beta)$	$\sqrt{\mathbb{V}}(\beta)$
Kyber 512	381	386.06	2.56	389.53	2.88	385.70	2.32
Kyber 768	623	634.41	2.96	638.23	3.30	634.00	2.66
Kyber 1024	873	891.13	3.31	895.24	3.66	890.63	2.96
LightSaber	404	408.81	2.65	412.24	2.96	408.35	2.39
Saber	648	659.36	3.00	663.10	3.32	658.85	2.68
FireSaber	890	907.76	3.34	911.78	3.68	907.16	2.97
ntruhps2048509	374	375.93	2.58	379.56	2.92	375.71	2.36
ntruhps2048677	521	522.78	2.82	526.77	3.18	522.67	2.57
ntruhps4096821	621	628.78	2.83	632.54	3.17	628.43	2.55
ntruhrss701	471	477.20	2.48	480.51	2.77	476.72	2.23

Table: Cryptographic parameters for NIST PQC finalists, using 2nd round parameters.

Observations: $\mathbb{E}(\beta) > \beta^*$, $\mathbb{V}(\beta)$ stays small.

Both observations should be good news:

Both observations should be good news:

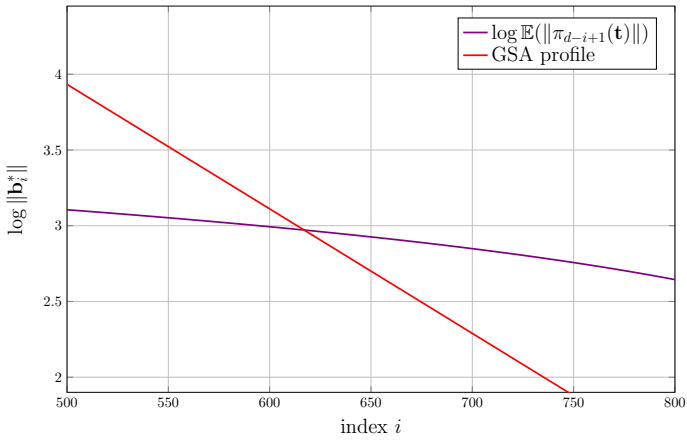
- $\mathbb{V}(\beta)$ stays small: successful β can not be much smaller than $\mathbb{E}(\beta)$. Low probability attacks by just picking smaller β should not be significantly cheaper.

Both observations should be good news:

- $\mathbb{V}(\beta)$ stays small: successful β can not be much smaller than $\mathbb{E}(\beta)$. Low probability attacks by just picking smaller β should not be significantly cheaper.
- $\mathbb{E}(\beta) > \beta^*$: it seems [ADPS16] underestimates the hardness of LWE. This is surprising, by accounting for $\beta < \beta^*$ we could expect the opposite effect.

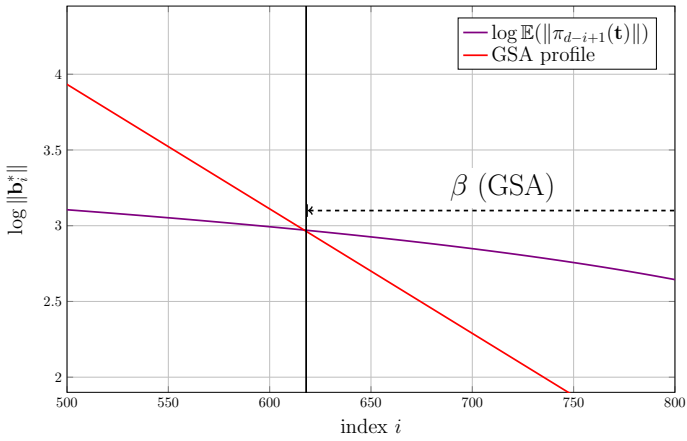
$\mathbb{E}(\beta) > \beta^*$ is caused by internally using the [CN11] simulator in place of the GSA.

- We modified the LWE estimator to use [CN11] in place of the GSA.
- Let's look at Kyber 512.



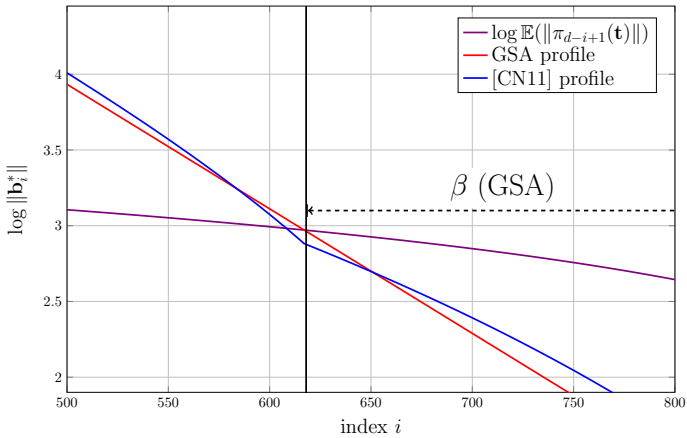
$\mathbb{E}(\beta) > \beta^*$ is caused by internally using the [CN11] simulator in place of the GSA.

- We modified the LWE estimator to use [CN11] in place of the GSA.
- Let's look at Kyber 512.



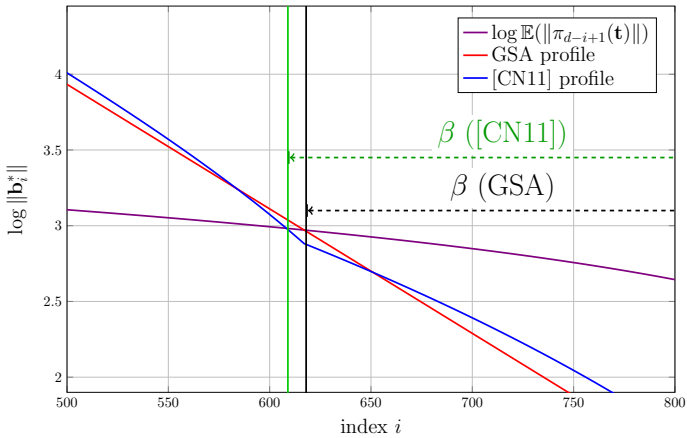
$\mathbb{E}(\beta) > \beta^*$ is caused by internally using the [CN11] simulator in place of the GSA.

- We modified the LWE estimator to use [CN11] in place of the GSA.
- Let's look at Kyber 512.



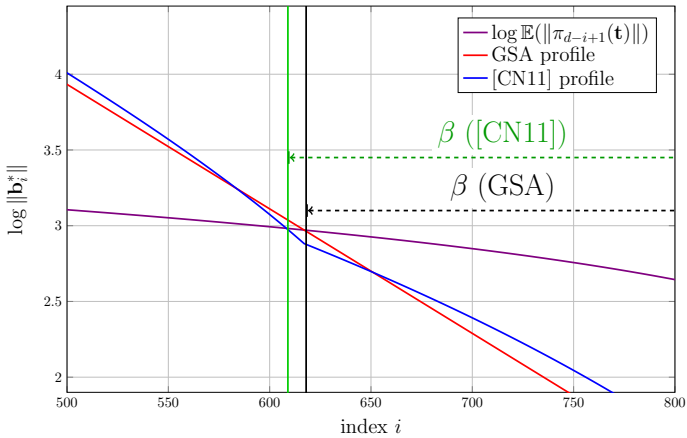
$\mathbb{E}(\beta) > \beta^*$ is caused by internally using the [CN11] simulator in place of the GSA.

- We modified the LWE estimator to use [CN11] in place of the GSA.
- Let's look at Kyber 512.



$\mathbb{E}(\beta) > \beta^*$ is caused by internally using the [CN11] simulator in place of the GSA.

- We modified the LWE estimator to use [CN11] in place of the GSA.
- Let's look at Kyber 512.
- This effect carries to our uSVP simulators, and similarly to [DDGR20]'s code.



Conclusions

- We capture the success probabilities of smaller-than-expected-successful block sizes.
- Effect seem consistent across secret and error distributions.
- Hardness of LWE does not seem significantly impacted.

Conclusions

- We capture the success probabilities of smaller-than-expected-successful block sizes.
- Effect seem consistent across secret and error distributions.
- Hardness of LWE does not seem significantly impacted.

- More details on eprint @ ia.cr/2020/1308
- Code and data @ github.com/fvirdia/usvp-simulation/

Conclusions

- We capture the success probabilities of smaller-than-expected-successful block sizes.
- Effect seem consistent across secret and error distributions.
- Hardness of LWE does not seem significantly impacted.

- More details on eprint @ ia.cr/2020/1308
- Code and data @ github.com/fvirdia/usvp-simulation/

Thank you


Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe.
Post-quantum key exchange - A new hope.
In Thorsten Holz and Stefan Savage, editors, *USENIX Security 2016*, pages 327–343. USENIX Association, August 2016.


Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer.
Revisiting the expected cost of solving uSVP and applications to LWE.
In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 297–322. Springer, Heidelberg, December 2017.


Yoshinori Aono, Yuntao Wang, Takuya Hayashi, and Tsuyoshi Takagi.
Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator.
In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 789–819. Springer, Heidelberg, May 2016.


Yuanmi Chen and Phong Q. Nguyen.
BKZ 2.0: Better lattice security estimates.
In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2011.


Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi.
LWE with side information: Attacks and concrete security estimation.
In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 329–358. Springer, Heidelberg, August 2020.


Claus-Peter Schnorr.

Lattice reduction by random sampling and birthday methods.
In Helmut Alt and Michel Habib, editors, *STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, February 27 - March 1, 2003, Proceedings*, volume 2607 of *Lecture Notes in Computer Science*, pages 145–156. Springer, 2003.

Claus-Peter Schnorr and M Euchner.
Lattice basis reduction: Improved practical algorithms and solving subset sum problems.
In *FCT*, 1991.