Security Guidelines for Implementing Homomorphic Encryption

J.-P. Bossuat, R. Cammarota, J. H. Cheon, I. Chillotti, B. R. Curtis, W. Dai, H. Gong, E. Hales, D. Kim, B. Kumara, C. Lee, X. Lu, C. Maple, A. Pedrouzo-Ulloa, R. Player, L. A. Ruiz Lopez, Y. Song, D. Yhee, B. Yildiz.

---

https://eprint.iacr.org/2024/463

FHE:IDEAs (Interesting Directions and Emerging Applications) Workshop
25 May 2024

Motivation

Context

- FHE applications and commercialisation has been advancing rapidly
- `homomorphicencryption.org` community standardisation (2017)
- Various documents have been published by this effort
  - 'Homomorphic Encryption Standard' [ACC+19]
  - Endorsed by community at 2nd and 3rd HE.org meetings (2018)
- Formal standardisation through ISO/IEC JTC1 SC27 WG2 (2021)

M. R. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter, S. Lokam, D. Micciancio, D. Moody, T. Morrison, A. Sahai, V. Vaikuntanathan. Homomorphic encryption standard. `eprint.iacr.org/2019/939`

ISO/IEC standardisation

- Standards (for BFV/BGV/CKKS/CGGI) still under development

- May need to explicitly offer parameter sets that are standardised
- May be useful to offer users guidance on choosing parameters

- For security, relevant to consider concrete hardness of LWE
- Many other parameters relevant when deploying FHE!

Secure LWE parameter sets for FHE in [ACC+19]

distribution	n	security level	logq
(-1, 1)	1024	128	27
		192	19
		256	14
	2048	128	54
		192	37
		256	29
	4096	128	109
		192	75
		256	58
	8192	128	218
		192	152
		256	118
	16384	128	438
		192	305
		256	237
	32768	128	881
		192	611
		256	476

This collaboration

- Established Oct 2021: security working group established
  - Supporting New Work Item ISO/IEC 18033-8 FHE (Aug 2021)

- Led by J. H. Cheon and H. Gong
- 19 collaborators in total

- Initial goal: develop Annex to Working Draft on parameter selection
- Later goal: produce a separate white paper
  - This work is that white paper!

- Participants in this working group are not all members of ISO/IEC
  - Including myself, until February 2024

Contributions in support of ISO/IEC effort

- Present LWE parameter sets that target particular levels of security
  - Security estimated using Lattice Estimator
  - Code: github.com/gong-cr/FHE-Security-Guidelines/
- Present 'functional' parameter for particular FHE schemes
  - Parameters relevant for security, correctness, and performance
  - Necessarily exemplar!
  - May not suit all implementations in all application contexts
- Survey parameter selection support in open source FHE libraries

M. R. Albrecht, R. P. and S. Scott. On the concrete hardness of Learning with Errors. In *Journal of Mathematical Cryptology*, 2015.
https://github.com/malb/lattice-estimator

Comparison to [ACC+19]

[ACC+19]	This work
Dimensions $n \in \{1024, ..., 32768\}$	Dimensions $n \in \{1024, ..., 131072\}$
Uniform, ternary, Gaussian secrets No sparse secrets	Binary, ternary, Gaussian secrets No sparse secrets
Max $\log q$ for fixed σ	Max $\log q$ for fixed n, σ Min $\log \sigma$ for fixed n, q
Not easily reproducible Difficult to update	Code to reproduce all tables Users can re-run code as needed
Only LWE parameters	Examples of full parameter sets
Describes various FHE schemes	Pointers to schemes and libraries
Describes various LWE algorithms	Pointers to cryptanalysis literature (but a few details today!)

What we don't cover (at the moment)

- NTRU-based FHE
- Hardness of LWE via reductions
- Weak instances of RLWE
- Machine learning attacks
- Side channel attacks
- IND-CPA^D security

Outline of rest of talk

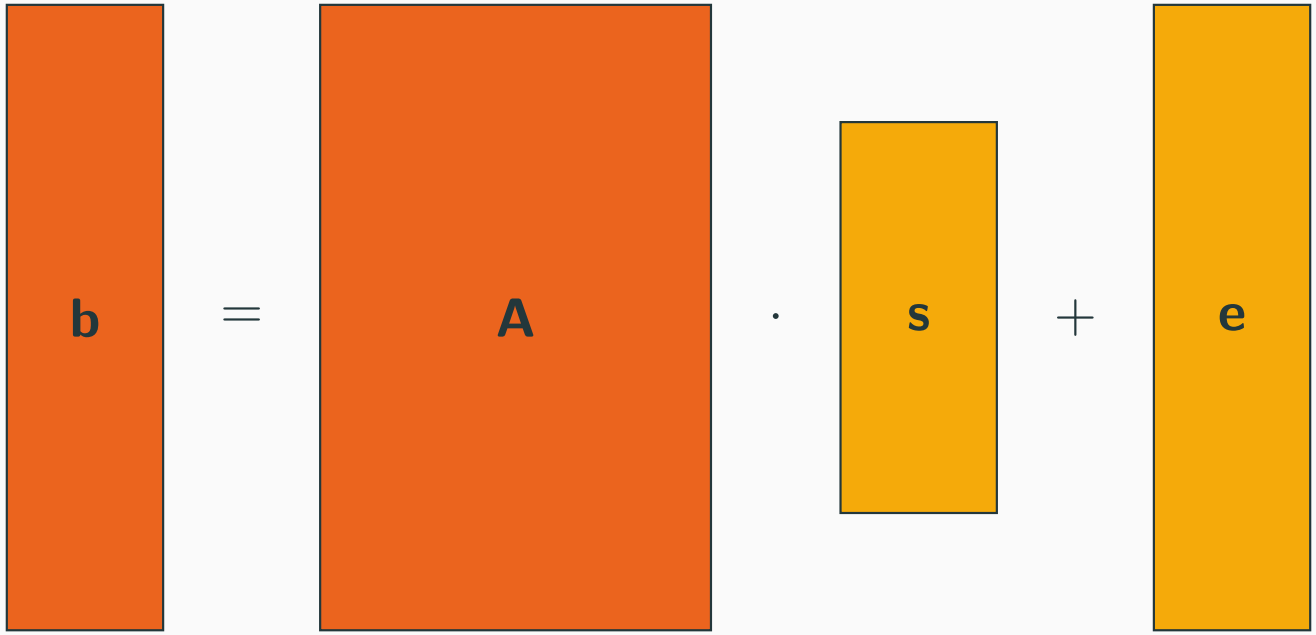
- Estimating security of FHE parameter sets
- Algorithms for solving LWE
- Example parameter sets given in our work
- Automated parameter selection in libraries and next steps

Estimating security of FHE parameter sets

Focus of security analysis

- Typical statement: FHE scheme is IND-CPA secure if LWE is hard
- Variants: LWE, Ring-LWE (RLWE), General-LWE (GLWE)
  - GLWE is also known as Module-LWE
- Do not know how to exploit algebraic structure of RLWE and GLWE
  - Hence, interpret as LWE instances
- Focus on concrete hardness of LWE

The Learning with Errors problem (LWE)

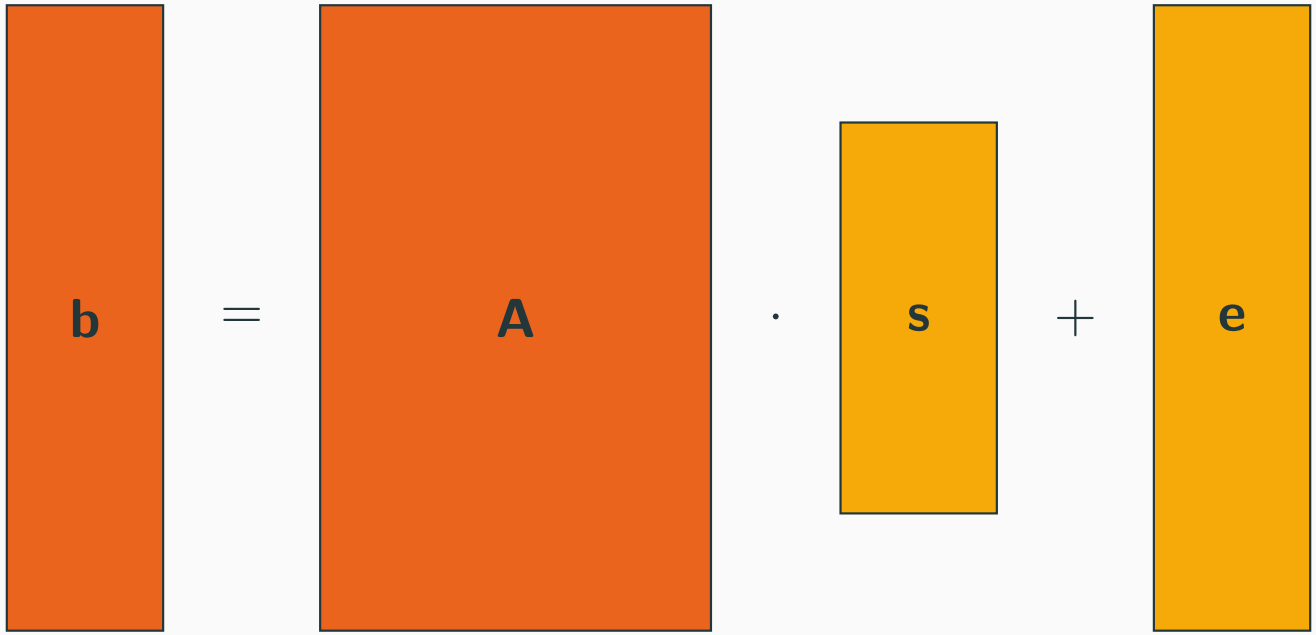


Search: Given $(\mathbf{A}, \mathbf{b})$, recover $\mathbf{s}$

Decision: Decide if pairs $(\mathbf{A}, \mathbf{b})$ are formed as above or uniformly at random

O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, 2005.

LWE parameters



n: dimension

q: modulus

σ: standard deviation of error distribution

m: number of samples

S: secret distribution

e.g. uniform in small range, sparse with h nonzero components

Target security levels

- Classical and quantum security levels according to FIPS 203 (draft)
- Category 128, 192, 256:
  - Any algorithm that solves the underlying LWE instance must require (classical) computational resources comparable to or greater than those required for key search on a block cipher with a 128-bit, respectively 192-bit, respectively 256-bit key.
- Category 128Q, 192Q, 256Q:
  - Any algorithm that solves the underlying LWE instance must require quantum computational resources comparable to or greater than those required for key search on a block cipher with a 128-bit, respectively 192-bit, respectively 256-bit key.

National Institute of Standards and Technology. Module-lattice-based key-encapsulation mechanism standard. Technical Report, Federal Information Processing Standards Publications 203 (draft), 2023.

The Lattice Estimator

README

Security Estimates for Lattice Problems

launch binder

docs passing

This Sage module provides functions for estimating the concrete security of Learning with Errors instances.

The main purpose of this estimator is to give designers an easy way to choose parameters resisting known attacks and to enable cryptanalysts to compare their results and ideas with other techniques known in the literature.

Quick Start

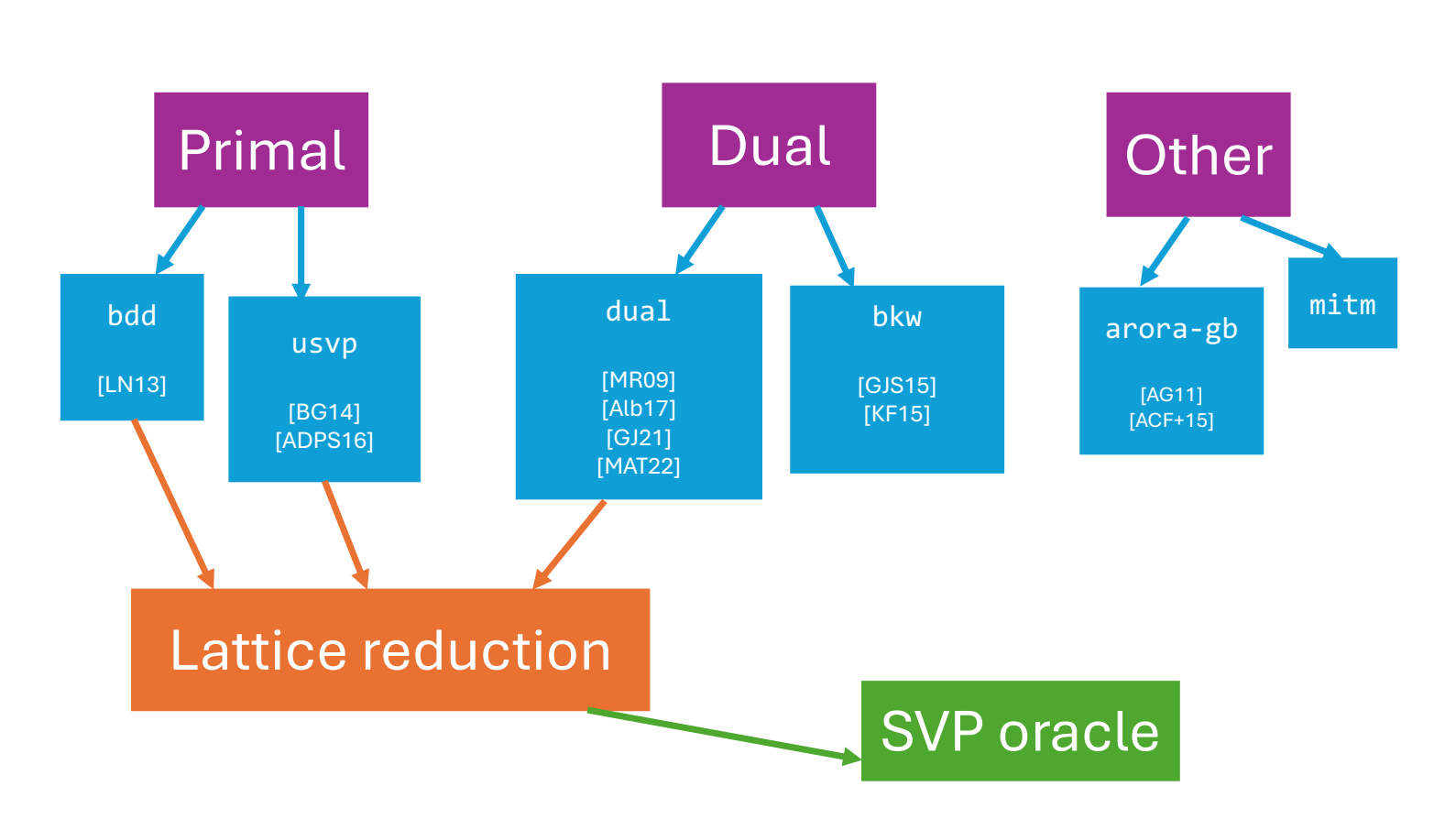
- Usage

```
>>> from estimator import *
>>> schemes.Kyber512
LWEParameters(n=512, q=3329, Xs=D(σ=1.22), Xe=D(σ=1.22), m=512, tag='Kyber 512')

>>> LWE.primal_usvp(schemes.Kyber512)
rop: ≈2^143.8, red: ≈2^143.8, δ: 1.003941, β: 406, d: 998, tag: usvp

>>> r = LWE.estimate.rough(schemes.Kyber512)
usvp                 :: rop: ≈2^118.6, red: ≈2^118.6, δ: 1.003941, β: 406, d: 998, tag:
dual_hybrid          :: rop: ≈2^115.4, red: ≈2^115.3, guess: ≈2^110.0, β: 395, p: 6, ζ:

>>> r = LWE.estimate(schemes.Kyber512)
```

M. R. Albrecht, R. P. and S. Scott. On the concrete hardness of Learning with Errors. In *Journal of Mathematical Cryptology*, 2015.

`https://github.com/malb/lattice-estimator`

The Lattice Estimator: supported algorithms

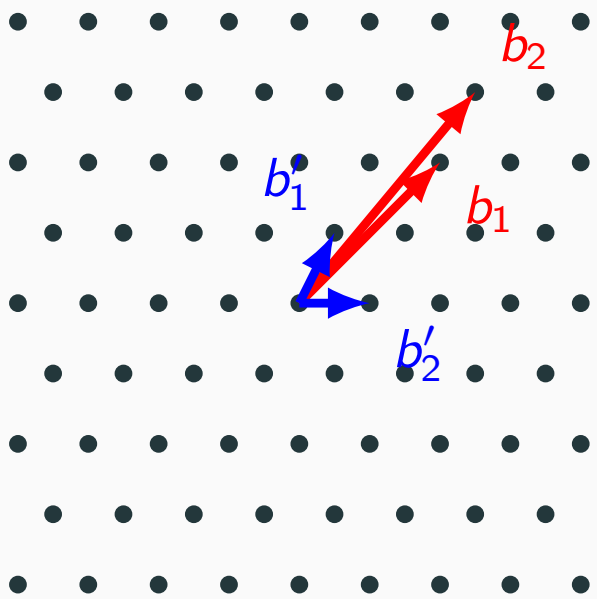


- Plus some hybrid combinatorial algorithms
  - `dual_hybrid`, `bdd_hybrid`, `bdd_mitm_hybrid`

[How07],[ACW19],[CHHS19],[EJK20]

Lattice Reduction

- Lattice basis reduction algorithms are central to solving LWE
- Examples of lattice reduction algorithms: LLL, BKZ, BKZ 2.0
- Goal is to take a 'bad' basis into a 'good' basis



BKZ (with block size β)

- The main lattice reduction algorithm considered in practice
- Involves solving exact SVP in dim β
- Various cost models have been proposed

Cost model:

The combination of the cost of solving SVP in dimension β and the number of required SVP oracle calls

Our cost model for BKZ-β in a lattice of dimension d:

- Classical: $T_{BKZ}(\beta, d) = 8d \cdot 2^{0.292\beta+16.4}$ `RC.BDGL16`
- Quantum: $T_{BKZ}(\beta, d) = 8d \cdot 2^{0.265\beta+16.4}$ `RC.LaaMosPol14`
- We also set `red_shape_model="gsa"`

Computational cost metric

- To meet a target security level, we need to define a metric for the "computational resources"
  - Multiple such metrics exist
  - Refinement is the subject of ongoing research

- Unit of computation used in the Estimator: ring operations (rops)
  - A parameter set meets Category 128 if algorithms are estimated to cost greater than 2^{128} rops when using the classical cost model

- We assess concrete security using `bdd`, `bdd_hybrid`, `usvp`, and `dual_hybrid` estimates from commit 00ec72c of the Estimator

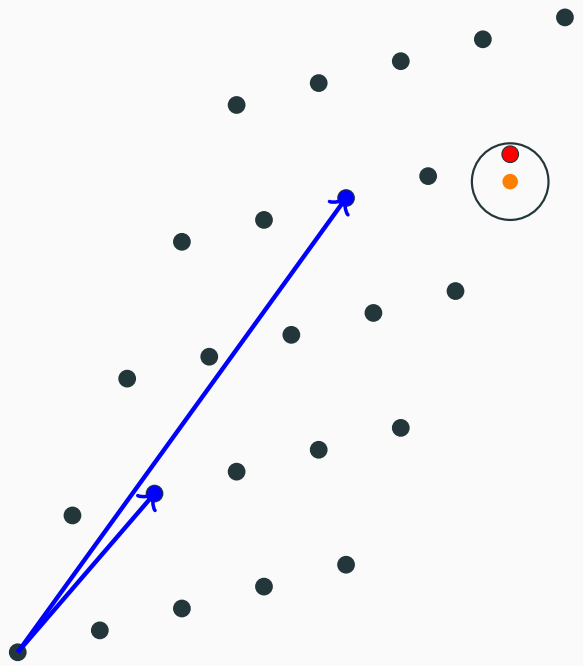
The Lattice Estimator: unsupported algorithms

- Some cryptanalytic algorithms applicable to typical FHE parameters are not currently supported in the Lattice Estimator
    - Examples include [May21],[HKLS22],[BLLW22],[EGMS23]

- Success prob. of the dual attack may be overestimated [DP23a]
    - May impact the utility of the dual attack estimates
    - Refinement of dual attacks is ongoing research [DP23b, PS24]

Algorithms for solving LWE

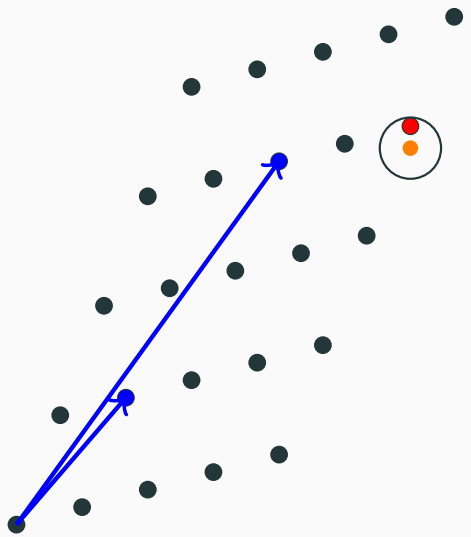
The Bounded Distance Decoding (BDD) problem



Bounded Distance Decoding:

Given $L(\mathbf{B})$, a target vector $\mathbf{t}$, and a guarantee that $\mathbf{t}$ is close to the lattice, find a $\mathbf{w} \in L$ such that $d(\mathbf{w}, \mathbf{t}) = d(L, \mathbf{t})$.

Solving LWE via BDD (bdd)

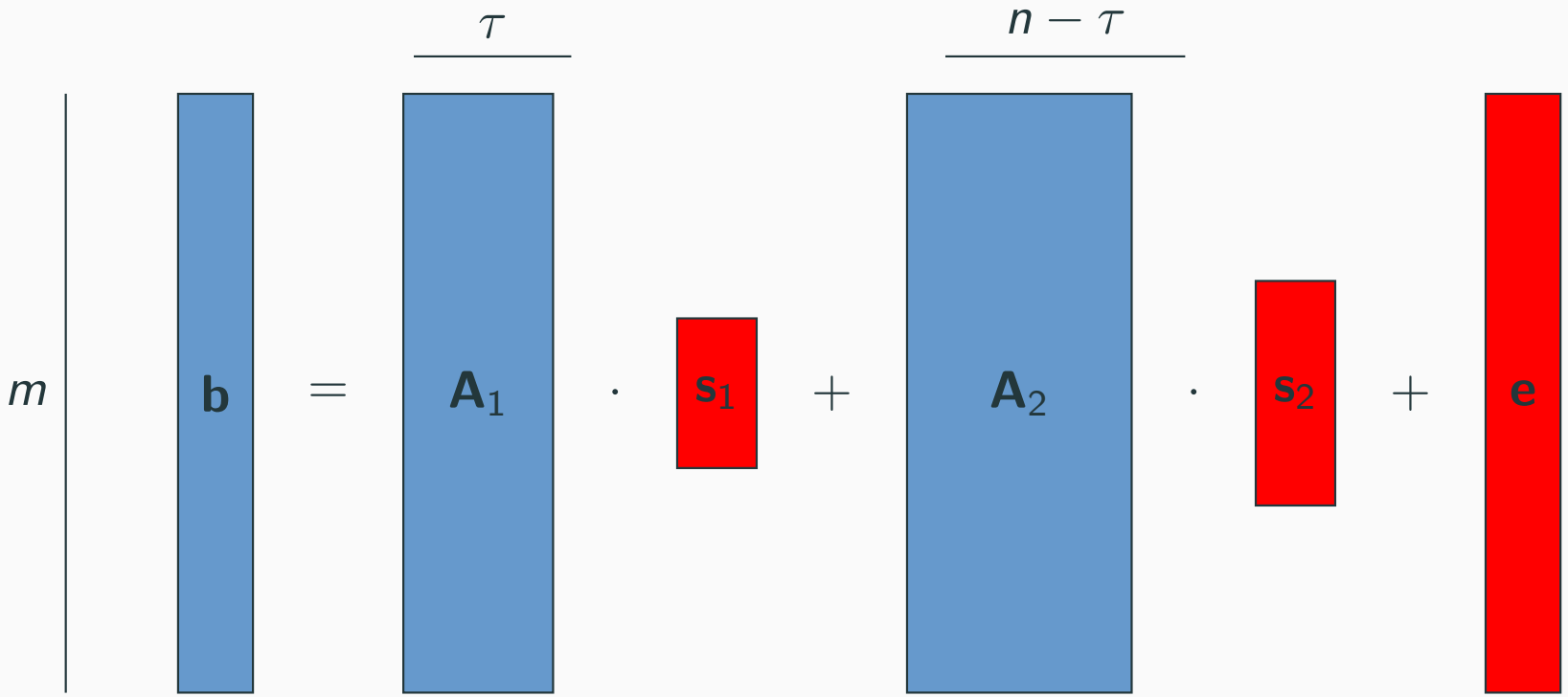


- Observe that $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$ is a perturbed point in $L(\mathbf{A})$
- Preprocess $\mathbf{A}$ with lattice reduction for better decoding
- Decode $\mathbf{b}$ to recover the lattice point $\mathbf{w} := \mathbf{A}\mathbf{s}$ and hence $\mathbf{s}$
  - Using e.g. Babai's Nearest Plane

M. Liu, P.Q. Nguyen. Solving BDD by Enumeration: An Update. In *CT-RSA*, 2013.

Solving small secret LWE by hybrid decoding (bdd_hybrid)

Split $\mathbf{A} = [\mathbf{A}_1|\mathbf{A}_2]$ into a guessing part and a part to be solved by BDD:



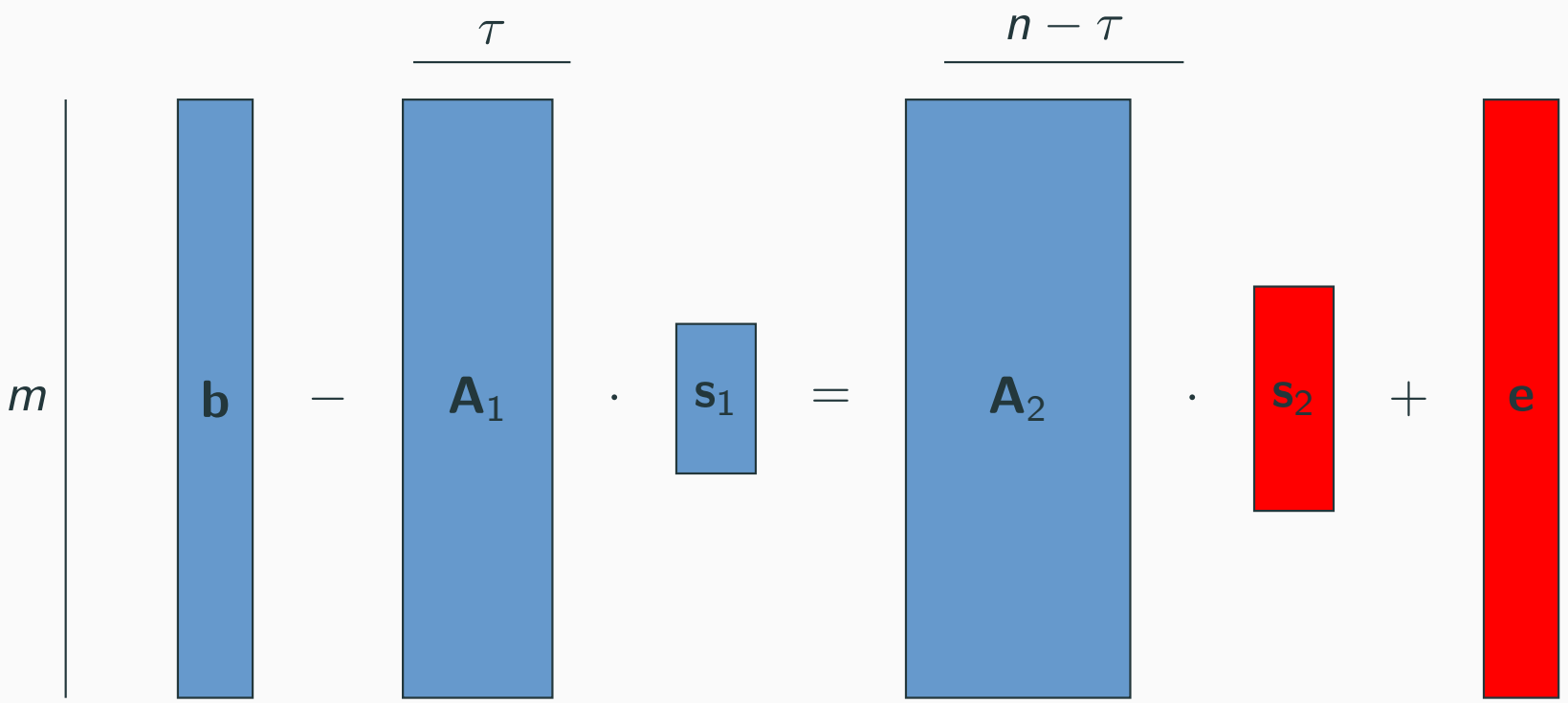
We guess $\mathbf{s}_1 \in \{-1, 0, 1\}^\tau$ and solve BDD to find $\mathbf{s}_2 \in \{-1, 0, 1\}^{n-\tau}$

Slide credit: Ben Curtis

M. R. Albrecht, B. R. Curtis, T. Wunderer. Exploring Trade-offs in Batch Bounded Distance Decoding. In *SAC*, 2019.

Solving small secret LWE by hybrid decoding (bdd_hybrid)

If we correctly guess the first τ components of $\mathbf{s}$, then we have:



- Using a BDD solver on input $\mathbf{b} - \mathbf{A}_1\mathbf{s}_1$ will recover $\mathbf{A}_2\mathbf{s}_2$

Slide credit: Ben Curtis

M. R. Albrecht, B. R. Curtis, T. Wunderer. Exploring Trade-offs in Batch Bounded Distance Decoding. In *SAC*, 2019.

The uSVP problem

The i^{th} successive minimum $\lambda_i(L)$ of a lattice L

The radius of the smallest ball centered at the origin containing at least i linearly independent lattice vectors

γ-unique Shortest Vector Problem

Given a basis $\mathbf{B}$ of a lattice L such that $\lambda_2(L) > \gamma\lambda_1(L)$, find a vector $\mathbf{v}$ in L such that $\|\mathbf{v}\| = \lambda_1(L)$

Primal uSVP attack on LWE (usvp)

- From the LWE instance $\mathbf{A}$ construct a lattice with uSVP structure
  - The short vector is $(\mathbf{e}, 1)$ for general LWE
  - The short vector is $(\mathbf{s}, \mathbf{e}, 1)$ for small secret LWE

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I} & -\mathbf{A} & \mathbf{b} \\ 0 & \mathbf{I} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- We find this short vector using lattice reduction
  - Quality of lattice reduction needed is given by [ADPS16]

E. Alkim, L. Ducas, T. Pöppelmann, P. Schwabe. Postquantum key exchange - A new hope. In *USENIX*, 2016.
S. Bai, S. D. Galbraith. Lattice Decoding Attacks on Binary LWE. In *ACISP*, 2014.

Dual attack on LWE instance $(\mathbf{A}, \mathbf{b})$

- Find a short $\mathbf{y}$ such that $\mathbf{yA} \equiv \mathbf{0} \pmod{q}$
  - i.e. $\mathbf{y}$ solves Short Integer Solution (SIS) problem for $\mathbf{A}$
- Compute $\langle \mathbf{y}, \mathbf{b} \rangle$
  - If $\mathbf{b} = \mathbf{As} + \mathbf{e}$, then $\langle \mathbf{y}, \mathbf{b} \rangle = \langle \mathbf{yA}, \mathbf{s} \rangle + \langle \mathbf{y}, \mathbf{e} \rangle \equiv \langle \mathbf{y}, \mathbf{e} \rangle \pmod{q}$
  - If $\mathbf{b}$ is uniformly random, so is $\langle \mathbf{y}, \mathbf{b} \rangle$
- If $\mathbf{y}$ short then $\langle \mathbf{y}, \mathbf{e} \rangle$ also short: can be distinguished from uniform
- The short $\mathbf{y}$ is found via lattice reduction

D. Micciancio, O. Regev. In *Lattice-based Cryptography, pages 147–191*. Springer Berlin Heidelberg, 2009.

Hybrid dual (`dual_hybrid`) on LWE instance $(\mathbf{A}, \mathbf{b})$

- Can have a hybrid dual variant by guessing part of $\mathbf{s}$:

$$\mathbf{A}\mathbf{s} + \mathbf{e} = \mathbf{A}_F\mathbf{s}_F + \mathbf{A}_L\mathbf{s}_L + \mathbf{e}$$

  for $\mathbf{A} = [\mathbf{A}_F|\mathbf{A}_L]$, $\mathbf{s} = [\mathbf{s}_F, \mathbf{s}_L]$

  - Guess $\mathbf{s}_F$ and find a short $\mathbf{y}$ solving SIS for $\mathbf{A}_L$
  - If guess correct, $\langle \mathbf{y}, \mathbf{b} - \mathbf{A}_F\mathbf{s}_F \rangle = \langle \mathbf{y}, \mathbf{e} \rangle$

- Check many $\mathbf{s}_F$ at once using FFT as distinguisher

- Further improvements: modulus switching, quantum, coding theory

Q. Guo, T. Johansson. Faster dual lattice attacks for solving LWE with applications to CRYSTALS. In *Asiacrypt*, 2021.
MATZOV. Report on the Security of LWE: Improved Dual Lattice Attack. `https://doi.org/10.5281/zenodo.6412487`, 2022.
M. R. Albrecht, Y. Shen. Quantum Augmented Dual Attack. *Eprint*, 2022.
K. Carrier, Y. Shen, J.P. Tillich. Faster Dual Lattice Attacks by Using Coding Theory. *Eprint*, 2022.
A. Pouly, Y. Shen. Provable Dual Attacks on Learning with Errors. *Eurocrypt*, 2024.

Example parameter sets

Max log q for target security 128 or 128Q for fixed n, $\sigma = 3.2$

n	$\log_2(q)$ (Classical)		$\log_2(q)$ (Quantum)	
	Ternary	Gaussian	Ternary	Gaussian
$\lambda = 128$				
1024	26	29	25	27
2048	54	56	50	52
4096	108	110	101	103
8192	217	219	203	205
16384	438	439	409	411
32768	881	883	825	827
65536	1776	1778	1663	1665
131072	3576	3578	3348	3351

Min log σ for target security 128 or 128Q for fixed n, q

n	$\log_2(q)$	$\log_2(\sigma)$ (Classical)			$\log_2(\sigma)$ (Quantum)		
		Binary	Ternary	Gaussian	Binary	Ternary	Gaussian
$\lambda = 128$							
630	32	17.9	16.6	14.2	18.9	17.7	15.4
1024		7.6	6.3	4.5	9.2	8.0	6.3
≥ 2048		2.0	2.0	2.0	2.0	2.0	2.0
630	64	49.9	48.6	46.2	50.9	49.7	47.4
750		46.8	45.5	43.0	48.0	46.7	44.4
870		43.7	42.4	39.9	45.0	43.8	41.4
1024		39.6	38.3	36.1	41.2	40.0	37.9
2048		12.6	11.4	9.4	16.0	14.8	12.7
≥ 4096		2.0	2.0	2.0	2.0	2.0	2.0

Example functional parameters for CGGI

λ	128	128	128	128
n	742	777	630	512
$\log_2(N)$	11	9	10	10
k	1	3	1	1
q	2^{64}	2^{64}	2^{32}	2^{27} / 2^{14}
t	2^4	2	2	2
χ_{LWE}	Binary	Binary	Binary	Ternary
χ_{GLWE}	Binary	Binary	Binary	Ternary
β_{ks}	2^{23}	2^{18}	2^7	128
ℓ_{ks}	5	3	3	-
β_{pbs}	2^{23}	2^{18}	2^2	2^7
ℓ_{pbs}	1	1	8	-
σ_{LWE}	$2^{-17.11}$	$2^{-18.03}$	2^{-15}	3.2
σ_{GLWE}	$2^{-51.60}$	$2^{-38.08}$	2^{-25}	3.2
p_{error}	2^{-40}	2^{-40}	2^{-165}	2^{-52}

- From TFHE-rs (cols 1, 2), TFHElib (col 3), OpenFHE (col 4)

Example functional parameters for CKKS with bootstrapping

	Set I	Set II	Set III
λ	128	128	192
$\log_2(N)$	16	16	17
Number of Slots[20]	32768	32768	65536
$\chi_{\mathbf{s}}$	Ternary	Ternary	Ternary
σ $(\chi_{\mathbf{e}})$	3.19	3.19	3.19
Base Prime Size	45	60	60
L (after bootstrapping)	11	5	14
$\log_2$(Scaling Factor)	35[21]	55	55
$\log_2(PQ)$	1769	1750	2425
$\log_2(Q)$	1464	1270	1765
$\log_2(P)$	305	480	660
Level cost of SlotsToCoeffs	4	2	2
Level cost of EvalMod	12	13	13
$\log_2(\Pr[\|\|I(X)\|\| > K])$[22]	-37.65	-37.65	-11.66
K	512	512	512
Level cost of CoeffsToSlots	3	2	2
Iterations[23]	1	1	1
Precision Bit	15.9	9.4	7.5

- From Lattigo (set I), OpenFHE (sets II, III)

Next steps

Param selection in major libraries

- OpenFHE (BFV, BGV, CKKS, FHEW)
  - Process to select parameters, depending on desired security level, depth support, batch size, key-switching mechanism, etc
  - For BFV, BGV, CKKS uses the tables in [ACC+19]
- SEAL (BFV, BGV, CKKS)
  - Provides maximal $\log q$ based on the tables in [ACC+19]
  - EVA compiler
- Lattigo (BFV, BGV, CKKS, FHEW)
  - User can set their own params, with some validation
- TFHE-rs (CGGI)
  - Offers parameter sets for different configurations
  - Concrete compiler with parameter optimisation tool
  - Uses Lattice Estimator to estimate security of the parameters

Future directions

- Expand the scope of the document
  - More schemes (e.g. NTRU-based)
  - More secret and error distributions
  - Broader attack scenarios (e.g. IND-CPA^D)

- Parameter selection more broadly
  - Automated frameworks for systematic parameter selection

Save the dates!

The 7th HomomorphicEncryption.org Standards Workshop

Sunday 13 October 2024

and

WAHC 2024: 12th Workshop on Encrypted Computing & Applied Homomorphic Cryptography (Submission deadline: 12 July!)

Monday 14 October 2024

Salt Lake City, Utah, USA

WAHC 2024 is an affiliated event of ACM CCS 2024 (14-18 Oct)

`https://homomorphicencryption.org/wahc-2024/`

Thank you!

Thank you!

Questions?

eprint.iacr.org/2024/463

Feedback on the paper is very welcome!

Code to reproduce all tables:

github.com/gong-cr/FHE-Security-Guidelines

github.com/WeiDaiWD/SEAL-Depth-Estimator