RUB

RUHR-UNIVERSITÄT BOCHUM

Making Fully Homomorphic Encryption practical

Construction and Cryptanalysis of lattice-based schemes

@ School on Correct and Secure implementation, Crete, 13.10.2017

Matthias Minihold
ECRYPT-NET Early Stage Researcher
Cryptology and IT-Security, Ruhr-Universität Bochum

hgi
Horst Görtz Institut
für IT-Sicherheit

Outline

1. Fully Homomorphic Encryption
   - Practical FHE
   - Privacy-Preserving Image Classification
   - Torus Fully Homomorphic Encryption (TFHE)
     - Introduction of acronyms: TFHE, TLWE, and TGSW.
     - Evaluating the multisum
     - Bootstrapping the multisum
     - 2D Torus
2. Learning with Errors (LWE)

Horst Görtz Institut für IT-Sicherheit

Practical FHE.

- Set out in search for the holy grail: Practical FHE!

Practical FHE.

- Set out in search for the holy grail: Practical FHE!
- Instead of looking for speed-ups of theoretic, asymptotic bounds of the best algorithms, we consider one example where a new FHE scheme can be applied in the cloud setting.

Practical FHE.

- Set out in search for the holy grail: Practical FHE!
- Instead of looking for speed-ups of theoretic, asymptotic bounds of the best algorithms, we consider one example where a new FHE scheme can be applied in the cloud setting.
- Secondment at CryptoExperts (CRX).

Practical FHE.

- Set out in search for the holy grail: Practical FHE!
- Instead of looking for speed-ups of theoretic, asymptotic bounds of the best algorithms, we consider one example where a new FHE scheme can be applied in the cloud setting.
- Secondment at CryptoExperts (CRX).
- Joint work (currently in submission) by:

Florian.Bourse@ens.fr

Michele.Minelli@ens.fr

Matthias.Minihold@rub.de

Pascal.Paillier@cryptoexperts.com

Outline

MNIST

- MNIST database: 60 000 training and 10 000 testing images,
- 28×28 pixels in 8 [bit] gray-scale.

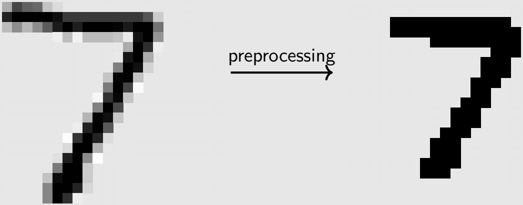

Figure: Preprocessing one MNIST's test set images.

Discretized Neural Networks are suited for FHE.

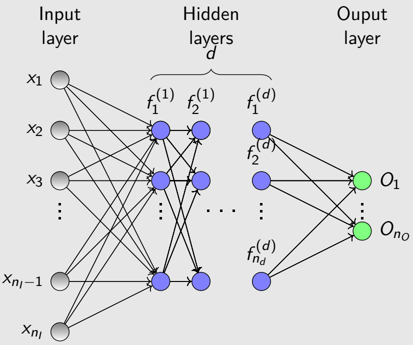

Figure: A Deep DiNN.

Close-up on a single neuron.

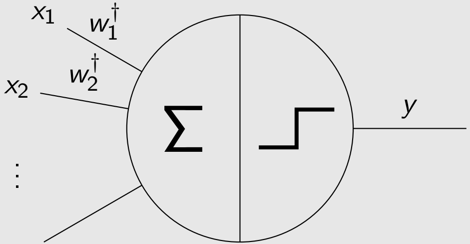

Figure: Evaluation of a single neuron. The output value is $y = \text{sign}(\langle \vec{w}^\dagger, \vec{x} \rangle)$, where $w_i^\dagger$ are the preprocessed (clear or encrypted) weights associated to the incoming wires and x_i are the corresponding (clear or encrypted) input values.

Neural Network activation functions.

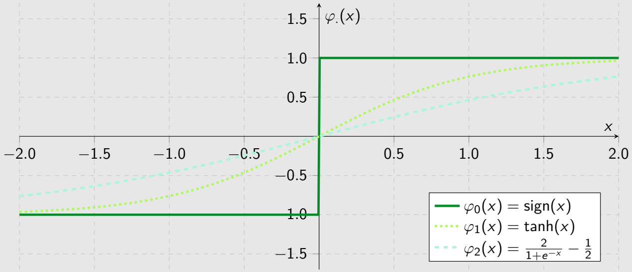

Figure: Several neural network activation functions and our choice φ_0.

Neural Network activation functions.

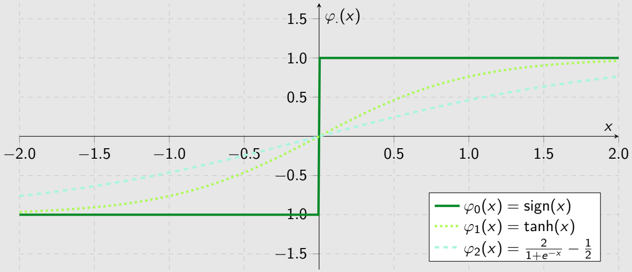

Figure: Several neural network activation functions and our choice φ_0.

- FHE encrypted inputs and weights trained on clear data,

Neural Network activation functions.

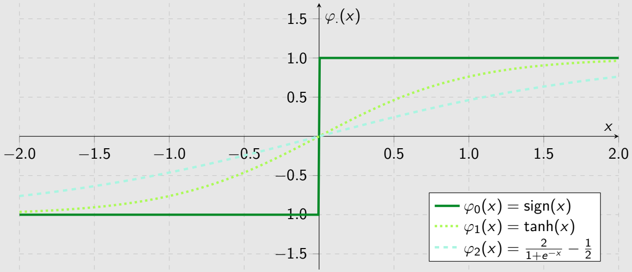

Figure: Several neural network activation functions and our choice φ_0.

- FHE encrypted inputs and weights trained on clear data,
- Our DiNN has a single hidden layer of 30 neurons,

Neural Network activation functions.

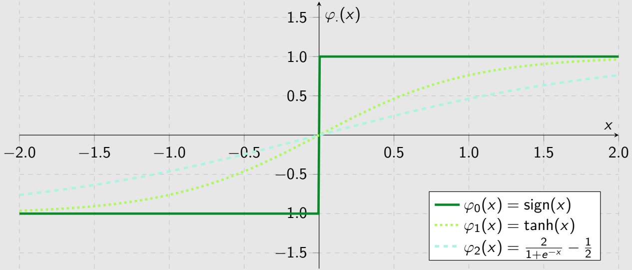

Figure: Several neural network activation functions and our choice φ_0.

- FHE encrypted inputs and weights trained on clear data,
- Our DiNN has a single hidden layer of 30 neurons,
- Experiments with clear vs. encrypted inputs and clear weights.

Homomorphic Evaluation of Deep Discretized NNs

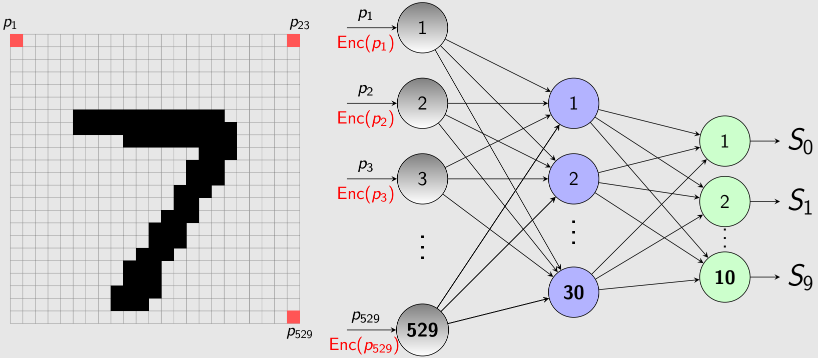

Figure: Running an experiment on our neural network with 529:30:10–topology. Classifies the depicted shape (without leaking privacy of the input data), and outputs the (encrypted) scores S_i assigned to each digit. The highest score is compared to the known label evaluating our success.

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].
- Homomorphic evaluation of our DiNN takes 0.88 [sec/classification].

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].
- Homomorphic evaluation of our DiNN takes 0.88 [sec/classification].
- DiNN achieves 90.03% accuracy (vs. 90.82% on clear inputs).

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].
- Homomorphic evaluation of our DiNN takes 0.88 [sec/classification].
- DiNN achieves 90.03% accuracy (vs. 90.82% on clear inputs).
- Our implementation requires about 28 [ms/bootstrapping] on a regular CPU (single core of Intel Core i7-4720HQ CPU @ 2.60GHz.)

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].
- Homomorphic evaluation of our DiNN takes 0.88 [sec/classification].
- DiNN achieves 90.03% accuracy (vs. 90.82% on clear inputs).
- Our implementation requires about 28 [ms/bootstrapping] on a regular CPU (single core of Intel Core i7-4720HQ CPU @ 2.60GHz.)

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].
- Homomorphic evaluation of our DiNN takes 0.88 [sec/classification].
- DiNN achieves 90.03% accuracy (vs. 90.82% on clear inputs).
- Our implementation requires about 28 [ms/bootstrapping] on a regular CPU (single core of Intel Core i7-4720HQ CPU @ 2.60GHz.)

Bootstrapping after hidden layer ensures low noise level

Encryption $(\langle \mathbf{w}, \mathbf{x} \rangle) \rightarrow$ Encryption $(\text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle))$ with "fresh" noise.

Results: It's a seven!

- With LWE dimension $n = 700$ and Gaussian noise parameter $\sigma = 2^{-30}$, we aim for a security level of roughly 80 [bit].
- Homomorphic evaluation of our DiNN takes 0.88 [sec/classification].
- DiNN achieves 90.03% accuracy (vs. 90.82% on clear inputs).
- Our implementation requires about 28 [ms/bootstrapping] on a regular CPU (single core of Intel Core i7-4720HQ CPU @ 2.60GHz.)

Bootstrapping after hidden layer ensures low noise level

$\text{Encryption}\left(\langle \mathbf{w}, \mathbf{x} \rangle\right) \rightarrow \text{Encryption}\left(\text{sign}\left(\langle \mathbf{w}, \mathbf{x} \rangle\right)\right)$ with “fresh” noise.

scale-invariance allows computing on encrypted data over many layers.

Outline

Horst Görtz Institut für IT-Sicherheit

TLWE – Unified treatment of (Ring-)LWE

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

Extend the TFHE scheme of Chilotti *et al.* [CGGI16]

- Trained network weights are available in clear,

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

Extend the TFHE scheme of Chilotti *et al.* [CGGI16]

- Trained network weights are available in clear,
- Evaluate the multisum using homomorphic additions,

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

Extend the TFHE scheme of Chilotti *et al.* [CGGI16]

- Trained network weights are available in clear,
- Evaluate the multisum using homomorphic additions,
- Tailored Bootstrapping mechanism,

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

Extend the TFHE scheme of Chilotti *et al.* [CGGI16]

- Trained network weights are available in clear,
- Evaluate the multisum using homomorphic additions,
- Tailored Bootstrapping mechanism,
- Careful choice of:

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

Extend the TFHE scheme of Chilotti *et al.* [CGGI16]

- Trained network weights are available in clear,
- Evaluate the multisum using homomorphic additions,
- Tailored Bootstrapping mechanism,
- Careful choice of:
  1. Message space (accommodates encryption scheme's largest results),

TLWE – Unified treatment of (Ring-)LWE

LWE assumption (over the Torus)

Given a secret $\mathbf{s} \xleftarrow{\$} \{0,1\}^n$, it is hard to distinguish between $(\mathbf{a}, b)$, where $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$ and $b = \langle \mathbf{s}, \mathbf{a} \rangle + e \in \mathbb{T}$, with $e \leftarrow \chi$, and $(\mathbf{u}, v) \xleftarrow{\$} \mathbb{T}^{n+1}$.

Extend the TFHE scheme of Chilotti *et al.* [CGGI16]

- Trained network weights are available in clear,
- Evaluate the multisum using homomorphic additions,
- Tailored Bootstrapping mechanism,
- Careful choice of:
  1. Message space (accommodates encryption scheme's largest results),
  2. Noise level (control growth to ensure correctness).

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Our Extended LWE-based encryption scheme

- $\mathbf{s} = \mathrm{Setup}(\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{T}^n$, $n = n(\lambda)$;

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Our Extended LWE-based encryption scheme

- $\mathbf{s} = \mathrm{Setup}(\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{T}^n$, $n = n(\lambda)$;
- $(\mathbf{a}, b) = \mathrm{Enc}(\mathbf{s}, \mu)$ with $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$, $b = \langle \mathbf{s}, \mathbf{a} \rangle + e + \frac{\mu}{2B+2}$, $e \leftarrow \chi$;

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Our Extended LWE-based encryption scheme

- $\mathbf{s} = \text{Setup}(\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{T}^n$, $n = n(\lambda)$;
- $(\mathbf{a}, b) = \text{Enc}(\mathbf{s}, \mu)$ with $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$, $b = \langle \mathbf{s}, \mathbf{a} \rangle + e + \frac{\mu}{2B+2}$, $e \leftarrow \chi$;
- $\text{Dec}(\mathbf{s}, (\mathbf{a}, b))$ returns $\lfloor (b - \langle \mathbf{s}, \mathbf{a} \rangle) \cdot (2B + 2) \rceil$ – correct w.o.p.

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Our Extended LWE-based encryption scheme

- $\mathbf{s} = \mathrm{Setup}(\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{T}^n$, $n = n(\lambda)$;
- $(\mathbf{a}, b) = \mathrm{Enc}(\mathbf{s}, \mu)$ with $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$, $b = \langle \mathbf{s}, \mathbf{a} \rangle + e + \frac{\mu}{2B+2}$, $e \leftarrow \chi$;
- $\mathrm{Dec}(\mathbf{s}, (\mathbf{a}, b))$ returns $\lfloor (b - \langle \mathbf{s}, \mathbf{a} \rangle) \cdot (2B + 2) \rceil$ – correct w.o.p.

Evaluating the multisum

- Given a task, throw neural network, choosing $B = \max_{\mathbf{w}} \|\mathbf{w}\|_1$,
- Given a message $\mu \in [-B, B] \subseteq \mathbb{N}$,
- Split the torus into $2B + 2$ "slices".

Our Extended LWE-based encryption scheme

- $\mathbf{s} = \mathrm{Setup}(\lambda)$ samples $\mathbf{s} \xleftarrow{\$} \mathbb{T}^n$, $n = n(\lambda)$;
- $(\mathbf{a}, b) = \mathrm{Enc}(\mathbf{s}, \mu)$ with $\mathbf{a} \xleftarrow{\$} \mathbb{T}^n$, $b = \langle \mathbf{s}, \mathbf{a} \rangle + e + \frac{\mu}{2B+2}$, $e \leftarrow \chi$;
- $\mathrm{Dec}(\mathbf{s}, (\mathbf{a}, b))$ returns $\lfloor (b - \langle \mathbf{s}, \mathbf{a} \rangle) \cdot (2B + 2) \rceil$ – correct w.o.p.

Our Homomorphism (Fixing secret key $\mathbf{s}$)

For $c_1 = (\mathbf{a}_1, b_1) \leftarrow \mathrm{Enc}(\mathbf{s}, \mu_1)$, $c_2 = (\mathbf{a}_2, b_2) \leftarrow \mathrm{Enc}(\mathbf{s}, \mu_2)$, $w \in \mathbb{Z}$:

$$\mathrm{Dec}(\mathbf{s}, (\mathbf{a}_1 + w \cdot \mathbf{a}_2, b_1 + w \cdot b_2)) = \mu_1 + w \cdot \mu_2.$$

Bootstrapping the multisum

Consider the torus $\mathbb{R}/\mathbb{Z} =: \mathbb{T} = (\mathbb{T}, +, *)$:

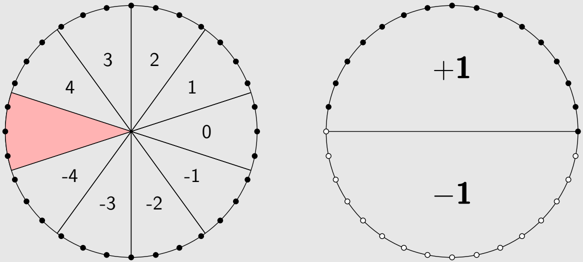

Figure: On the left, discretize torus elements onto the wheel (the $2N$ dots on it) by rounding to the closest dot. Each slice corresponds to one of the possible results of the multisum operation (the colored slice represents the forbidden zone). On the right, final result of the bootstrapping: each dot of the top (resp. bottom) part of the wheel is mapped to $+1$ and -1, respectively.

Open Questions

- Paralellism across neural network should be straight-forward,

Open Questions

- Paralellism across neural network should be straight-forward,
- better neural network (look-up hacky tricks in the literature),

Open Questions

- Paralellism across neural network should be straight-forward,
- better neural network (look-up hacky tricks in the literature),
- Look at speed-ups due to Lagrange representation & FFT-techniques,

Open Questions

- Paralellism across neural network should be straight-forward,
- better neural network (look-up hacky tricks in the literature),
- Look at speed-ups due to Lagrange representation & FFT-techniques,
- Optimization of cryptographic algorithms: Batched bootstrapping

Open Questions

- Paralellism across neural network should be straight-forward,
- better neural network (look-up hacky tricks in the literature),
- Look at speed-ups due to Lagrange representation & FFT-techniques,
- Optimization of cryptographic algorithms: Batched bootstrapping
- generalization to 2-D torus $\mathbb{R}^2/\mathbb{Z}^2 =: \mathbb{T}^2 = (\mathbb{T}^2, +, *)$?

Open Questions

- Paralellism across neural network should be straight-forward,
- better neural network (look-up hacky tricks in the literature),
- Look at speed-ups due to Lagrange representation & FFT-techniques,
- Optimization of cryptographic algorithms: Batched bootstrapping
- generalization to 2-D torus $\mathbb{R}^2/\mathbb{Z}^2 =: \mathbb{T}^2 = (\mathbb{T}^2, +, *)$?

Open Questions

- Paralellism across neural network should be straight-forward,
- better neural network (look-up hacky tricks in the literature),
- Look at speed-ups due to Lagrange representation & FFT-techniques,
- Optimization of cryptographic algorithms: Batched bootstrapping
- generalization to 2-D torus $\mathbb{R}^2/\mathbb{Z}^2 =: \mathbb{T}^2 = (\mathbb{T}^2, +, *)$?

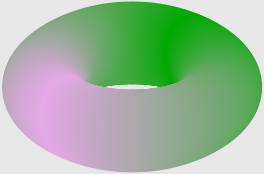

Figure: 2D Torus.

Learning with Errors

Cryptanalysis of computationally hard, underlying problems, i.e. assess algorithmic approaches to solve average- and worst-case instances.

Learning with Errors

Cryptanalysis of computationally hard, underlying problems, i.e. assess algorithmic approaches to solve average- and worst-case instances.

Promising to use (side-channel) information, parallelization and fplll, then shift and balance workload to enumeration in a clever way to break lattice challenges or post-quantum candidates.

Learning with Errors

Cryptanalysis of computationally hard, underlying problems, i.e. assess algorithmic approaches to solve average- and worst-case instances.

Promising to use (side-channel) information, parallelization and fplll, then shift and balance workload to enumeration in a clever way to break lattice challenges or post-quantum candidates.

Best current (primal) attack: BDD

First LLL/BKZ-reduction of the basis matrix, then enumerate points.

Learning with Errors (LWE) Problem

Given 3-parameters and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}, \mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \mod q$, find: $\mathbf{s}$.

Learning with Errors (LWE) Problem

Given 3-parameters and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}, \mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \mod q$, find: $\mathbf{s}$.

Dimension n, modulus q, and error-bound $\|\mathbf{e}\|$ depend on sec-level λ.

Learning with Errors (LWE) Problem

Given 3-parameters and $\mathbf{A} \in \mathbb{Z}_q^{m \times n}, \mathbf{t} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \mod q$, find: $\mathbf{s}$.

Dimension n, modulus q, and error-bound $\|\mathbf{e}\|$ depend on sec-level λ.

Current Best Asymptotic Complexity of Attacking LWE.

Let $q = n^{\alpha}, \|\mathbf{e}\| = n^{\beta} \in \mathcal{O}(\mathrm{poly}(n))$:

$$T_{LWE} = 2^{\mathrm{c}_{\mathrm{LWE}} \cdot n \cdot \frac{\log n}{\log(q/\|\mathbf{e}\|)}},$$

with $\mathrm{c}_{\mathrm{LWE}}$ a function of $\mathrm{c}_{\mathrm{BKZ}}$ and poly(n)- or 2^n-space requirements.

LWE in Theory / Practice

Attacking LWE In Practice Step 1

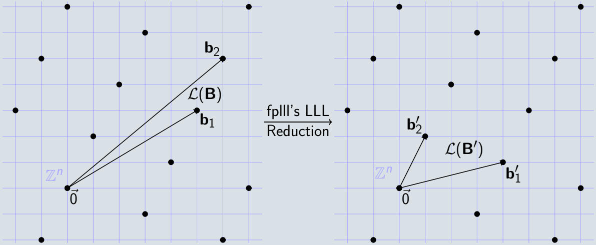

Figure: Step 1: Find a 'good' basis for lattice $\Lambda_q(\mathbf{A})$, i.e. using fplll.

LWE in Theory / Practice

Attacking LWE In Practice Step 1

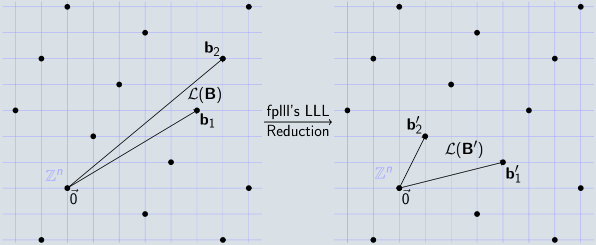

Figure: Step 1: Find a 'good' basis for lattice $\Lambda_q(\mathbf{A})$, i.e. using fplll.

Attacking LWE In Practice Step 2

Enumerate all points within radius $||\mathbf{e}||$ relative to $\mathbf{t}$.

RUHR-UNIVERSITÄT BOCHUM

QUESTIONS?

Thank you for your attention!

This research has received funding from the European Union's Horizon 2020 research and innovation programme Marie Skłodowska-Curie ITN ECRYPT-NET (Project Reference 643161).

Horst Görtz Institut für IT-Sicherheit