FINAL: Faster FHE instantiated with NTRU and LWE.

Charlotte Bonte*, Ilia Iliashenko⋆, Jeongeun Park⋆,
Hilder V. L. Pereira⋆, and Nigel P. Smart⁂

⋆ imec-COSIC, KU Leuven
∗ Intel Corporation, Emerging Security Lab
⁂ imec-COSIC, KU Leuven and Zama Inc.

06 Dec 2022

RLWE vs NTRU

Hardness assumptions for FHE

Let $R := \mathbb{Z}[X]/\langle X^N + 1\rangle$

RLWE problem

- Secret: $s \in R$
- $a_i \leftarrow \mathcal{U}(R_q)$
- $e_i \leftarrow \chi$
- $b_i := a_i \cdot s + e_i \bmod q$

Then $(a_i, b_i) \approx_C \mathcal{U}(R_q^2)$.

Hardness assumptions for FHE

Let $R := \mathbb{Z}[X]/\langle X^N + 1\rangle$

RLWE problem

- Secret: $s \in R$
- $a_i \leftarrow \mathcal{U}(R_q)$
- $e_i \leftarrow \chi$
- $b_i := a_i \cdot s + e_i \bmod q$

Then $(a_i, b_i) \approx_C \mathcal{U}(R_q^2)$.

NTRU problem

- Secret: $f \in R$ with small coefficients.
- Noise: $g_i \leftarrow \chi$
- $b_i := g_i \cdot f^{-1} \bmod q$

Then $b_i \approx_C \mathcal{U}(R_q)$.

Hardness assumptions for FHE

Let $R := \mathbb{Z}[X]/\langle X^N + 1 \rangle$

RLWE problem

- Secret: $s \in R$
- $a_i \leftarrow \mathcal{U}(R_q)$
- $e_i \leftarrow \chi$
- $b_i := a_i \cdot s + e_i \bmod q$

Then $(a_i, b_i) \approx_C \mathcal{U}(R_q^2)$.

NTRU problem

- Secret: $f \in R$ with small coefficients.
- Noise: $g_i \leftarrow \chi$
- $b_i := g_i \cdot f^{-1} \bmod q$

Then $b_i \approx_C \mathcal{U}(R_q)$.

Ideally, we should halve the memory consumption and running time.

Past NTRU constructions

- Indeed, past NTRU-based FHE schemes, like YASHE, were more efficient than the corresponding RLWE-based schemes [LN14].

[LN14] Tancrède Lepoint and Michael Naehrig. *A Comparison of the Homomorphic Encryption Schemes FV and YASHE*. AFRICACRYPT 2014.

Past NTRU constructions

- Indeed, past NTRU-based FHE schemes, like YASHE, were more efficient than the corresponding RLWE-based schemes [LN14].
- However, they used $R_q := \mathbb{Z}_q[X]/\langle X^N + 1\rangle$ with $q \in \Omega(2^N)$.

[LN14] Tancrède Lepoint and Michael Naehrig. *A Comparison of the Homomorphic Encryption Schemes FV and YASHE*. AFRICACRYPT 2014.

Past NTRU constructions

- Indeed, past NTRU-based FHE schemes, like YASHE, were more efficient than the corresponding RLWE-based schemes [LN14].
- However, they used $R_q := \mathbb{Z}_q[X]/\langle X^N + 1\rangle$ with $q \in \Omega(2^N)$.
- While first ciphers based on NTRU used very small q, like $q \in O(N)$.

[LN14] Tancrède Lepoint and Michael Naehrig. *A Comparison of the Homomorphic Encryption Schemes FV and YASHE*. AFRICACRYPT 2014.

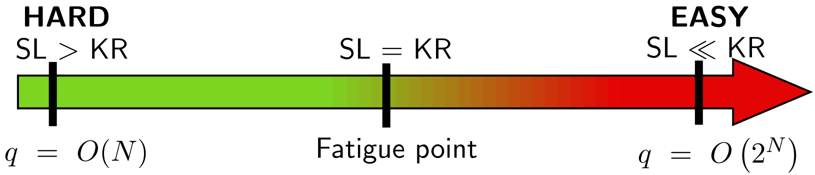
Past NTRU constructions

- Indeed, past NTRU-based FHE schemes, like YASHE, were more efficient than the corresponding RLWE-based schemes [LN14].
- However, they used $R_q := \mathbb{Z}_q[X]/\langle X^N + 1\rangle$ with $q \in \Omega(2^N)$.
- While first ciphers based on NTRU used very small q, like $q \in O(N)$.
- It turns out that the NTRU problem is insecure when q is too big in comparison with N.

[LN14] Tancrède Lepoint and Michael Naehrig. *A Comparison of the Homomorphic Encryption Schemes FV and YASHE*. AFRICACRYPT 2014.

Overstretched regime

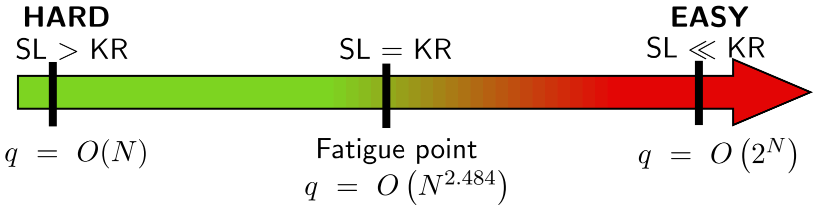
- Key recovery attacks (KR): exponential time in N.
- New sublattice attacks (SL): hardness varies depending on q.



[DW21] L. Ducas and W. van Woerden. *NTRU fatigue: How stretched is overstretched?*. ASIACRYPT 2021.

Overstretched regime

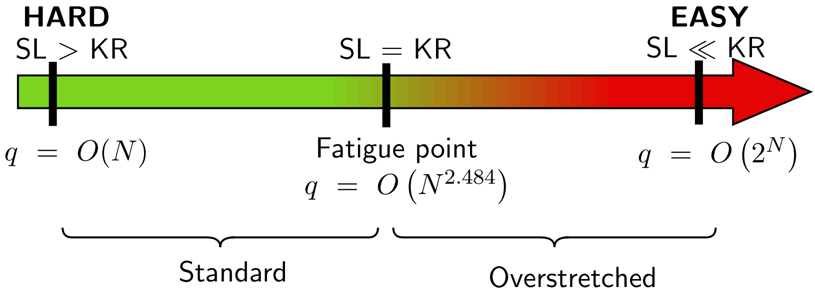
- Key recovery attacks (KR): exponential time in N.
- New sublattice attacks (SL): hardness varies depending on q.



[DW21] L. Ducas and W. van Woerden. *NTRU fatigue: How stretched is overstretched?*. ASIACRYPT 2021.

Overstretched regime

- Key recovery attacks (KR): exponential time in N.
- New sublattice attacks (SL): hardness varies depending on q.



[DW21] L. Ducas and W. van Woerden. *NTRU fatigue: How stretched is overstretched?*. ASIACRYPT 2021.

Contributions

Contributions

- NGS: a GSW-like scheme based on NTRU (faster external product).

Contributions

- NGS: a GSW-like scheme based on NTRU (faster external product).
- We use NGS to bootstrap an NTRU-based scheme.

Contributions

- NGS: a GSW-like scheme based on NTRU (faster external product).
- We use NGS to bootstrap an NTRU-based scheme.
- Thus, we obtain FHE based on NTRU with parameters below the stretched regime: $q = \tilde{O}(N)$ instead of $O\left(N^{2.484}\right)$.

Contributions

- NGS: a GSW-like scheme based on NTRU (faster external product).
- We use NGS to bootstrap an NTRU-based scheme.
- Thus, we obtain FHE based on NTRU with parameters below the stretched regime: $q = \tilde{O}(N)$ instead of $O\left(N^{2.484}\right)$.
- We use NGS to bootstrap an LWE-based scheme.

Contributions

- NGS: a GSW-like scheme based on NTRU (faster external product).
- We use NGS to bootstrap an NTRU-based scheme.
- Thus, we obtain FHE based on NTRU with parameters below the stretched regime: $q = \tilde{O}(N)$ instead of $O\left(N^{2.484}\right)$.
- We use NGS to bootstrap an LWE-based scheme.
- Our LWE-NTRU scheme is 28% faster than TFHE.

Contributions

- NGS: a GSW-like scheme based on NTRU (faster external product).
- We use NGS to bootstrap an NTRU-based scheme.
- Thus, we obtain FHE based on NTRU with parameters below the stretched regime: $q = \tilde{O}(N)$ instead of $O\left(N^{2.484}\right)$.
- We use NGS to bootstrap an LWE-based scheme.
- Our LWE-NTRU scheme is 28% faster than TFHE.
- Bootstrapping keys 45% smaller than in TFHE.

NTRU-based GSW-like Scheme

NGS: NTRU-based GSW-like Scheme

- Secret: sk is a small $f \in R$.
- Scalar ciphertext: $c = g/f + \Delta m \in R_q$, for a random g.
- Vector ciphertext:

$$\mathbf{c} = \mathbf{g}/f + (B^0, B^1, ..., B^{\ell-1}) \cdot m \in R_q^\ell$$

where $\ell = \lceil \log_B(q) \rceil$.

NGS: NTRU-based GSW-like Scheme

- Secret: sk is a small $f \in R$.
- Scalar ciphertext: $c = g/f + \Delta m \in R_q$, for a random g.
- Vector ciphertext:

$$\mathbf{c} = \mathbf{g}/f + (B^0, B^1, ..., B^{\ell-1}) \cdot m \in R_q^\ell$$

where $\ell = \lceil \log_B(q) \rceil$.

$$\mathbf{G} = \begin{bmatrix} B^0 & 0 \\ B^1 & 0 \\ \vdots & \vdots \\ B^{\ell-1} & 0 \\ 0 & B^0 \\ 0 & B^1 \\ \vdots & \vdots \\ 0 & B^{\ell-1} \end{bmatrix}$$

NGS: NTRU-based GSW-like Scheme

- Secret: sk is a small $f \in R$.
- Scalar ciphertext: $c = g/f + \Delta m \in R_q$, for a random g.
- Vector ciphertext:

$$\mathbf{c} = \mathbf{g}/f + (B^0, B^1, ..., B^{\ell-1}) \cdot m \in R_q^\ell$$

  where $\ell = \lceil \log_B(q) \rceil$.

- External product:
  - Let $\mathbf{y} = \text{Decomp}_B(c) \in R_q^\ell$
  - Compute $c_{mult} = \mathbf{y} \cdot \mathbf{c} = \sum_{i=0}^{\ell-1} y_i \cdot c_i \in R_q$
  - Cost: ℓ multiplications on R_q

NGS vs ring-GSW

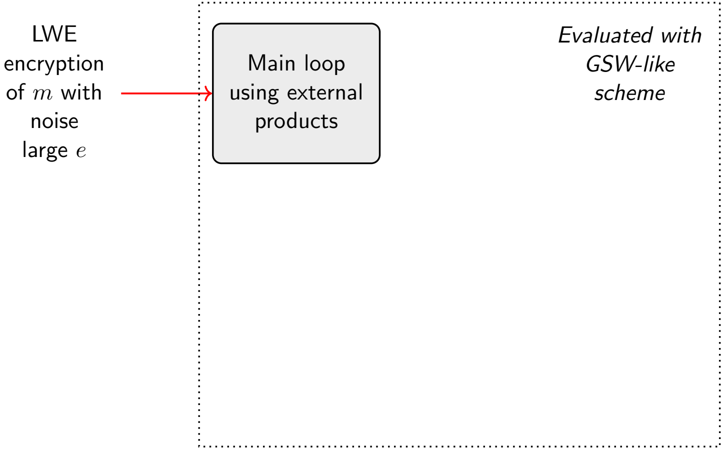
	NGS	GSW
Scalar ciphertext	1 poly	2 polys
"Full" ciphertext	ℓ polys	$4 \cdot \ell$ polys
External prod.	ℓ mults	$4 \cdot \ell$ mults

FHE with fast bootstrapping

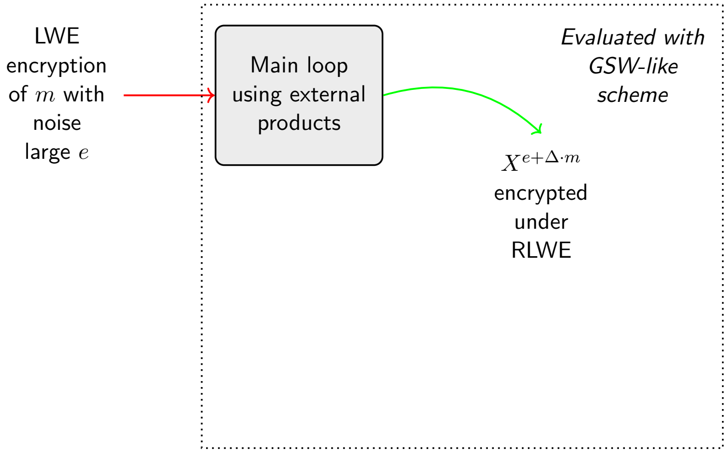
Overview of bootstrapping in FHEW and TFHE

LWE encryption of m with noise large e

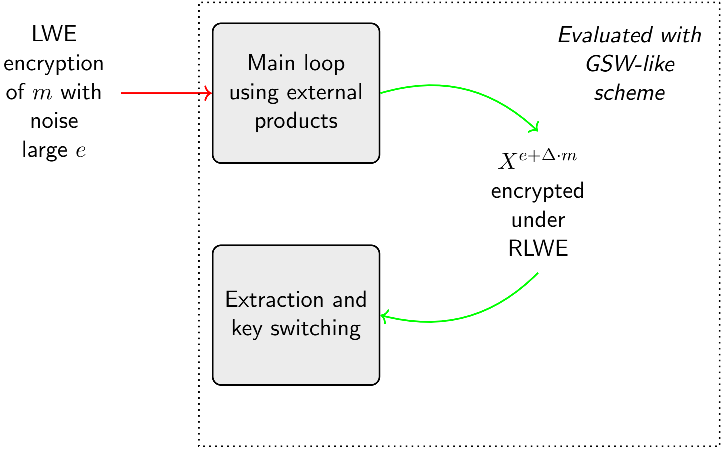
Overview of bootstrapping in FHEW and TFHE



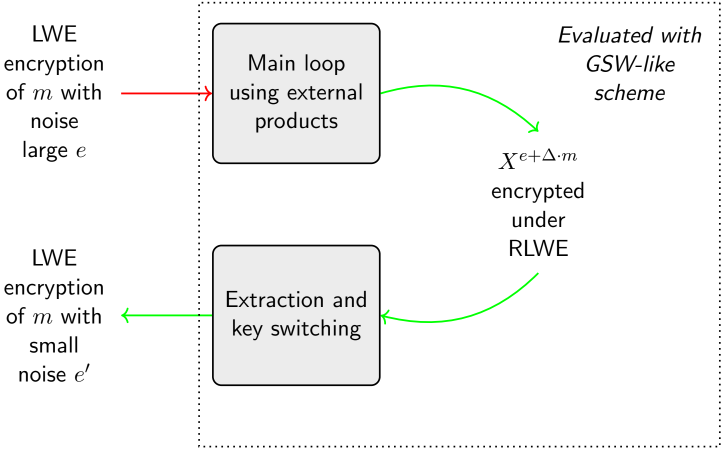
Overview of bootstrapping in FHEW and TFHE



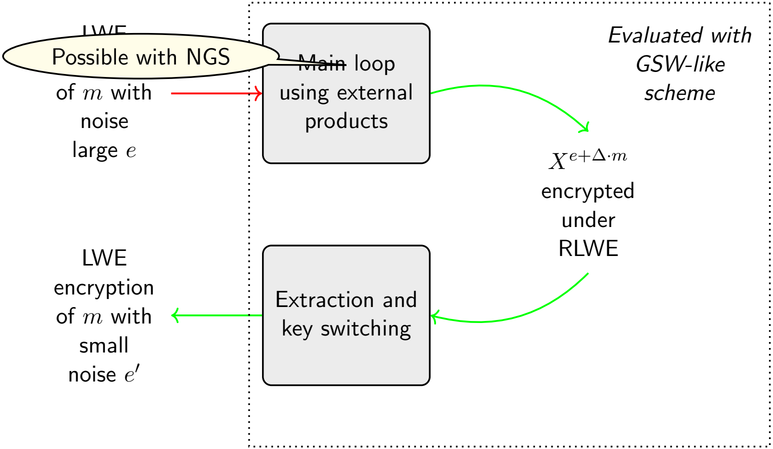
Overview of bootstrapping in FHEW and TFHE



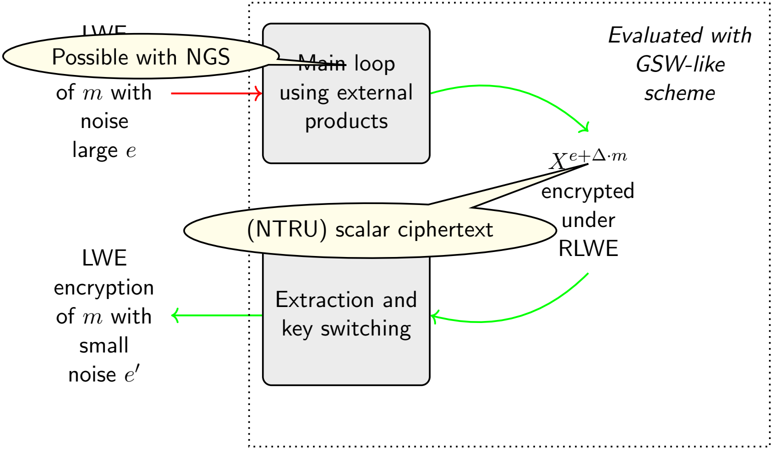
Overview of bootstrapping in FHEW and TFHE



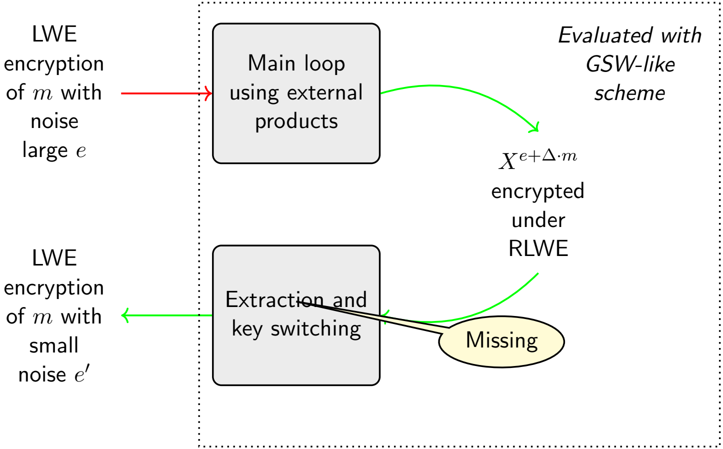
Overview of bootstrapping in FHEW and TFHE



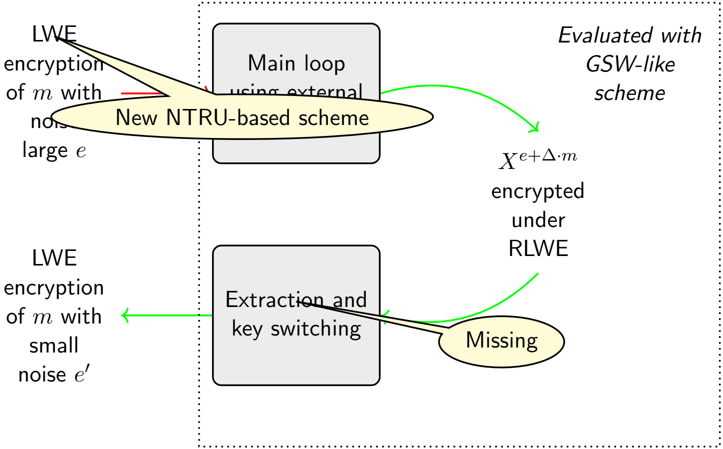
Overview of bootstrapping in FHEW and TFHE



Overview of bootstrapping in FHEW and TFHE



Overview of bootstrapping in FHEW and TFHE



Matrix-NTRU (MNTRU) scheme

- We propose a base scheme whose security relies on the MNTRU problem.
- $\mathbf{C} := \mathbf{G} \cdot \mathbf{F}^{-1} \bmod q \ \approx_C \ \mathcal{U}(\mathbb{Z}_q^{n\times n})$.

Matrix-NTRU (MNTRU) scheme

- We propose a base scheme whose security relies on the MNTRU problem.
- $\mathbf{C} := \mathbf{G} \cdot \mathbf{F}^{-1} \bmod q \ \approx_C \ \mathcal{U}(\mathbb{Z}_q^{n\times n})$.
- But then to decrypt we have to multiply by the matrix $\mathbf{F}$...

Matrix-NTRU (MNTRU) scheme

- We propose a base scheme whose security relies on the MNTRU problem.
- $\mathbf{C} := \mathbf{G} \cdot \mathbf{F}^{-1} \bmod q \ \approx_C \ \mathcal{U}(\mathbb{Z}_q^{n \times n})$.
- But then to decrypt we have to multiply by the matrix $\mathbf{F}$...
- So, we define a ciphertext by using just the first row of $\mathbf{C}$

$$\mathsf{row}_1(\mathbf{C}) = \mathbf{G} \cdot \mathsf{col}_1(\mathbf{F}^{-1})$$

- Then, ciphertext is a vector $\mathbf{c} \in \mathbb{Z}_q^n$.

Matrix-NTRU (MNTRU) scheme

- We propose a base scheme whose security relies on the MNTRU problem.
- $\mathbf{C} := \mathbf{G} \cdot \mathbf{F}^{-1} \bmod q \ \approx_C \ \mathcal{U}(\mathbb{Z}_q^{n \times n})$.
- But then to decrypt we have to multiply by the matrix $\mathbf{F}$...
- So, we define a ciphertext by using just the first row of $\mathbf{C}$

$$\mathsf{row}_1(\mathbf{C}) = \mathbf{G} \cdot \mathsf{col}_1(\mathbf{F}^{-1})$$

- Then, ciphertext is a vector $\mathbf{c} \in \mathbb{Z}_q^n$.
- Let $\mathbf{f}$ be the first column of $\mathbf{F}$.
- To decrypt, we compute $\mathbf{c} \cdot \mathbf{f}$.
- Thus, we can also use external products to compute $X^{\mathbf{c} \cdot \mathbf{f}}$.

Extraction and key switching

- We start with $\mathsf{Enc}_{\mathbf{f}}(m) \in \mathbb{Z}_q^n$.
- After n external products: $\mathsf{Enc}_f(X^{e+\Delta m}) \in R_q$.

Extraction and key switching

- We start with $\mathsf{Enc}_{\mathbf{f}}(m) \in \mathbb{Z}_q^n$.
- After n external products: $\mathsf{Enc}_f(X^{e+\Delta m}) \in R_q$.
- Extraction is roughly the same as in TFHE: $\mathsf{Enc}_f(m) \in \mathbb{Z}_q^N$.

Extraction and key switching

- We start with $\mathsf{Enc}_{\mathbf{f}}(m) \in \mathbb{Z}_q^n$.
- After n external products: $\mathsf{Enc}_f(X^{e+\Delta m}) \in R_q$.
- Extraction is roughly the same as in TFHE: $\mathsf{Enc}_f(m) \in \mathbb{Z}_q^N$.

- Then, we propose a key switching for NTRU ciphertexts...
- After key switching: $\mathsf{Enc}_{\mathbf{f}}(m) \in \mathbb{Z}_q^n$.

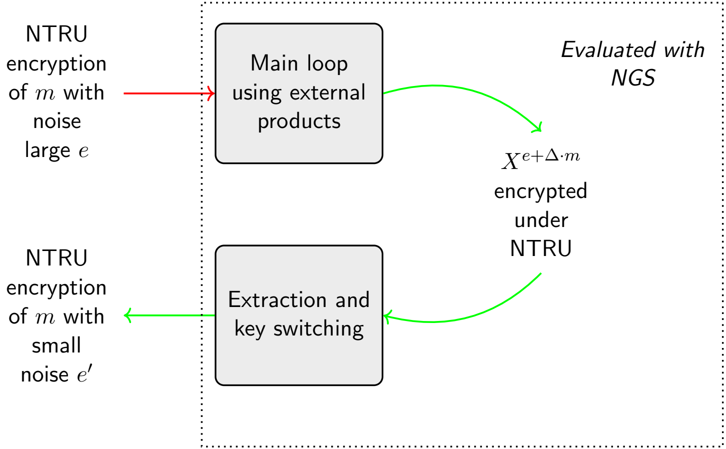
Extraction and key switching

- We start with $\mathsf{Enc}_{\mathbf{f}}(m) \in \mathbb{Z}_q^n$.
- After n external products: $\mathsf{Enc}_f(X^{e+\Delta m}) \in R_q$.
- Extraction is roughly the same as in TFHE: $\mathsf{Enc}_f(m) \in \mathbb{Z}_q^N$.

- Then, we propose a key switching for NTRU ciphertexts...
- After key switching: $\mathsf{Enc}_{\mathbf{f}}(m) \in \mathbb{Z}_q^n$.

We can set $q \in \tilde{O}(N)$, thus, below the fatigue point $O(N^{2.48})$.

FHE based on NTRU



Using NGS to bootstrap LWE-based scheme

Considering a base scheme over the LWE problem, as in FHEW and TFHE, we have

- $\text{Enc}_{\mathbf{s}}(m) = (\mathbf{a}, b = \mathbf{a} \cdot \mathbf{s} + e + \Delta m) \in \mathbb{Z}_q^{n+1}$

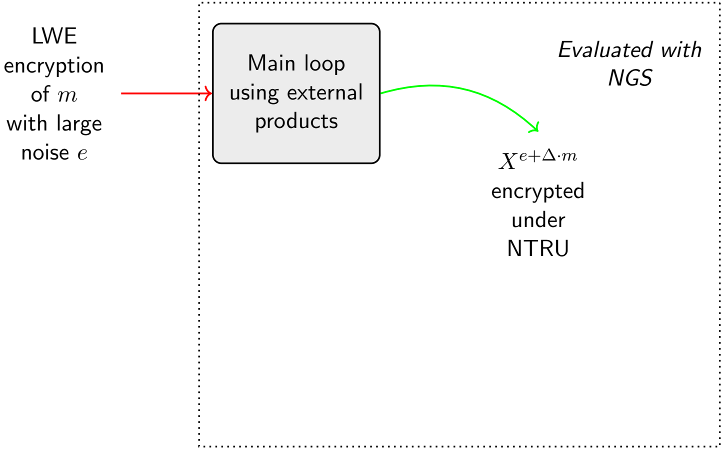
Using NGS to bootstrap LWE-based scheme

Considering a base scheme over the LWE problem, as in FHEW and TFHE, we have

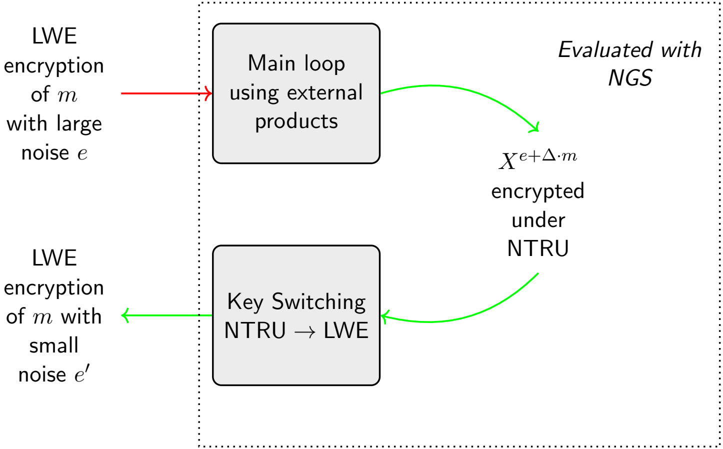
- $\mathsf{Enc}_{\mathbf{s}}(m) = (\mathbf{a}, b = \mathbf{a} \cdot \mathbf{s} + e + \Delta m) \in \mathbb{Z}_q^{n+1}$
- We can already use NGS external products to compute

$$\mathsf{Enc}_f(X^{b-\mathbf{a}\cdot\mathbf{s}}) = \mathsf{Enc}_f(X^{e+\Delta m}) \in R_q$$

FHE based on LWE and NTRU



FHE based on LWE and NTRU



Practical results and conclusion

Proof-of-concept implementation

C++ implementation available in
`https://github.com/KULeuven-COSIC/FINAL`.

For a fair comparison, we used the same FFT library as TFHE and compiled our code with the same compilation flags.

Parameter selection

For each basis B_i we have a different dimension $\ell_i := \lceil \log_{B_i}(Q) \rceil$ for n_i bootstrapping keys. For the first n_1 external products, we use the decomposition base B_1, then we use B_2 for the remaining n_2 external products.

Base scheme	n	q	N	Q	(B_1, n_1)	(B_2, n_2)	ℓ_1	ℓ_2
MNTRU	800	$\approx 2^{17}$	2^{10}	$\approx 2^{19.8}$	$(8, 750)$	$(16, 50)$	7	5
LWE	610	$\approx 2^{16.5}$	2^{10}	$\approx 2^{19.8}$	$(8, 140)$	$(16, 470)$	7	5

Parameter selection

For each basis B_i we have a different dimension $\ell_i := \lceil \log_{B_i}(Q) \rceil$ for n_i bootstrapping keys. For the first n_1 external products, we use the decomposition base B_1, then we use B_2 for the remaining n_2 external products.

Base scheme	n	q	N	Q	(B_1, n_1)	(B_2, n_2)	ℓ_1	ℓ_2
MNTRU	800	$\approx 2^{17}$	2^{10}	$\approx 2^{19.8}$	$(8, 750)$	$(16, 50)$	7	5
LWE	610	$\approx 2^{16.5}$	2^{10}	$\approx 2^{19.8}$	$(8, 140)$	$(16, 470)$	7	5

Bootstrapping executes n external products...

Practical results

	Key switching key	Bootstrapping key	Mult. on R_Q	FFTs	Run. time
TFHE	40 MB	31 MB	7560	6300	66 ms
MNTRU	34.4 MB	43 MB	11000	6300	92 ms
LWE	26.3 MB	13 MB	3330	3940	48 ms

Practical results

	Key switching key	Bootstrapping key	Mult. on R_Q	FFTs	Run. time
TFHE	40 MB	31 MB	7560	6300	66 ms
MNTRU	34.4 MB	43 MB	11000	6300	92 ms
LWE	26.3 MB	13 MB	3330	3940	48 ms

LWE-NTRU FHE:

45% less key material and bootstrapping 28% faster than TFHE.

Conclusion

- We proposed an FHE scheme based on NTRU with parameters lying outside of the stretched regime.
  - We hope that this result will bring more attention to NTRU-based FHE and that further researcher will improve its efficiency.

Conclusion

- We proposed an FHE scheme based on NTRU with parameters lying outside of the stretched regime.
  - We hope that this result will bring more attention to NTRU-based FHE and that further researcher will improve its efficiency.
- We proposed an NTRU-based GSW-like scheme (NGS) with smaller ciphertexts and faster homomorphic multiplications in comparison to RLWE-based GSW.
  - We hope that NGS will be used to replace GSW in other applications apart from the bootstrapping presented here.

Conclusion

- We proposed an FHE scheme based on NTRU with parameters lying outside of the stretched regime.
  - We hope that this result will bring more attention to NTRU-based FHE and that further researcher will improve its efficiency.
- We proposed an NTRU-based GSW-like scheme (NGS) with smaller ciphertexts and faster homomorphic multiplications in comparison to RLWE-based GSW.
  - We hope that NGS will be used to replace GSW in other applications apart from the bootstrapping presented here.
- We proposed an NTRU-to-LWE key switching and showed how to use it in combination with NGS to bootstrap LWE schemes.
- Therewith, we ran bootstrapping faster than in TFHE.

Thanks!

Any question or comment?

Please, feel free to contact!

`https://hilder-vitor.github.io`