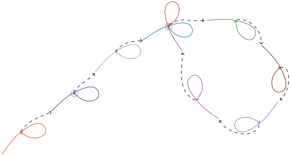
Memory Efficient Attacks on Small LWE Keys

Andre Esser [1] Rahul Girme [2] Arindam Mukherjee [2] Santanu Sarkar [2]

[1]Technology Innovation Institute, UAE

[2]Indian Institute of Technology Madras, India

Asiacrypt 2023, 7 December 2023



Motivation

- Used in NTRU, Kyber, Dilithium.
- Hybrid Attack (Nick Howgrave-Graham, 2007).
- May's combinatorial attack (Crypto 2021).
- Enormous memory required.

This work: New memoryless algorithm and time-memory trade-off.

Preliminaries

Definition (Ternary LWE problem)

Given: $q = \mathrm{poly}(n)$, $w \in \mathbb{N}$, $\mathbf{A} \in \mathbb{Z}_q^{n \times n}$, $\mathbf{b} \in \mathbb{Z}_q^n$, satisfying $\mathbf{A}\mathbf{s} = \mathbf{b} + \mathbf{e} \mod q$, where $\mathbf{s}, \mathbf{e} \in \{-1, 0, 1\}^n$ and $\mathrm{wt}(\mathbf{s}) = w$.
Find: $\mathbf{s} \in \{-1, 0, 1\}^n$.

- Simple linear algebra if $\mathbf{e}$ is known.
- If $\mathbf{A}\mathbf{s} - \mathbf{b} \in \{-1, 0, 1\}^n$, then output $\mathbf{s}$. Brute-force in 3^n with polynomial memory.
- For simplicity assume $\mathbf{s}$ has equal number of 1 and -1.
- For the rest of the talk $\mathrm{wt}(\mathbf{s}) = \frac{n}{2}$.

Basics of Collision Search

For a random function $f : S \to S$, $(y_1, y_2) \in S^2$ with $f(y_1) = f(y_2)$ can be found using $\mathcal{O}(\sqrt{|S|})$ evaluations of f and polynomial memory.

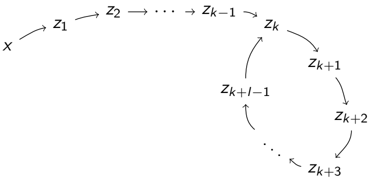
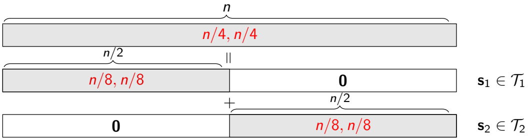


Figure: $f^i(x)$ is denoted by z_i

Procedure can be extended to two random functions f_1 and f_2, namely $\textsc{Rho}(f_1, f_2, x)$, where x is random starting point. Collision is unique (depends on the starting point).

Solving LWE via Collision Search (van Vredendaal 2016)

Recall $\mathbf{A}\mathbf{s} = \mathbf{b} + \mathbf{e} \mod q$. Split $\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2$.



$$\mathbf{A}\mathbf{s}_1 = \mathbf{b} + \mathbf{e} - \mathbf{A}\mathbf{s}_2$$

Guess lower ℓ coordinates of $\mathbf{e}$ (May 2021). Small, Get for free, Not enough for linear algebra.

$$\underbrace{\pi_\ell(\mathbf{A}\mathbf{s}_1)}_{f_1(\mathbf{s}_1)} = \underbrace{\pi_\ell(\mathbf{b} + \mathbf{e} - \mathbf{A}\mathbf{s}_2)}_{f_2(\mathbf{s}_2)}$$

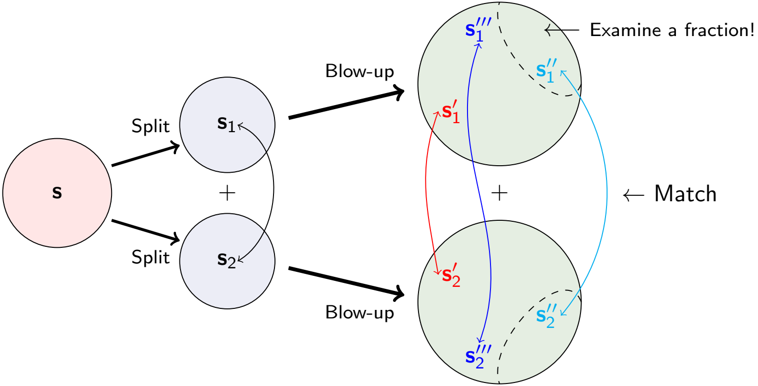
Search for collisions between f_1 and f_2.

$$T = \tilde{\mathcal{O}}\left(\#\mathrm{Collisions} \cdot T_{\mathrm{Rho}}\right) = \boxed{2^{1.125n}}.$$

Representation Technique (Howgrave-Graham, Joux 2010)

$(1, 0, -1, 0, -1, 1) =$

$$\begin{array}{cccc}
(1,0,-1,0,\ \ 0,0) & (1,0,\ \ 0,0,-1,0) & (0,0,-1,0,\ \ 0,1) & (0,0,\ \ 0,0,-1,1) \\
+ & + & + & \\
(0,0,\ \ 0,0,-1,1) & (0,0,-1,0,\ \ 0,1) & (1,0,\ \ 0,0,-1,0) & (1,0,-1,0,\ \ 0,0)
\end{array}$$



Impact of representations

Search space goes up but representations take care of it.

$$T = \tilde{\mathcal{O}}\left(\frac{\#\text{Collisions}}{\#\text{Representations}} \cdot T_{\text{Rho}}\right).$$

Basic Collision Search Instantiations

1. van Vredendaal:

n

$n/4, n/4$

$n/2$ $=$

$n/8, n/8$ | **0**

$+$ $n/2$

0 | $n/8, n/8$

$2^{1.125n}$

2. May's Rep-0:

1 0 $n/8, n/8$ 0 -1

$+$

0 1 $n/8, n/8$ -1 0

$2^{1.0919n}$

3. May's Rep-1:

1 -1 $n/8 + d, n/8 + d$

$+$

-1 1 $n/8 + d, n/8 + d$

$2^{0.9635n}$

4. May's Rep-2:

2 -2 2 -1 -2 1

$+$

-2 2 -1 2 1 -2

$2^{0.962n}$

Our result: $\boxed{2^{0.8425n}}$

Nested-Collision-Search (DDKS 2016)

$$\mathbf{s} = \mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3 + \mathbf{s}_4.$$

$$\mathbf{A}(\mathbf{s}_1 + \mathbf{s}_2) = \mathbf{b} - \mathbf{A}(\mathbf{s}_3 + \mathbf{s}_4) + \mathbf{e} \mod q.$$

Guess the first 2ℓ coordinates of $\mathbf{e}$. (Small, Get for free and Sufficient to identify $\mathbf{s}$ uniquely!)

$$\pi_{2\ell}\big(\mathbf{A}(\mathbf{s}_1 + \mathbf{s}_2)\big) = \pi_{2\ell}(\mathbf{b} + \mathbf{e}) - \pi_{2\ell}\big(\mathbf{A}(\mathbf{s}_3 + \mathbf{s}_4)\big) \mod q,$$

Randomly choose $\mathbf{r} := \pi_\ell\big(\mathbf{A}(\mathbf{s}_1 + \mathbf{s}_2)\big)$.

$$\underbrace{\pi_\ell(\mathbf{A}\mathbf{s}_1)}_{f_1(\mathbf{s}_1)} = \underbrace{\mathbf{r} - \pi_\ell(\mathbf{A}\mathbf{s}_2)}_{f_2(\mathbf{s}_2)} \mod q$$

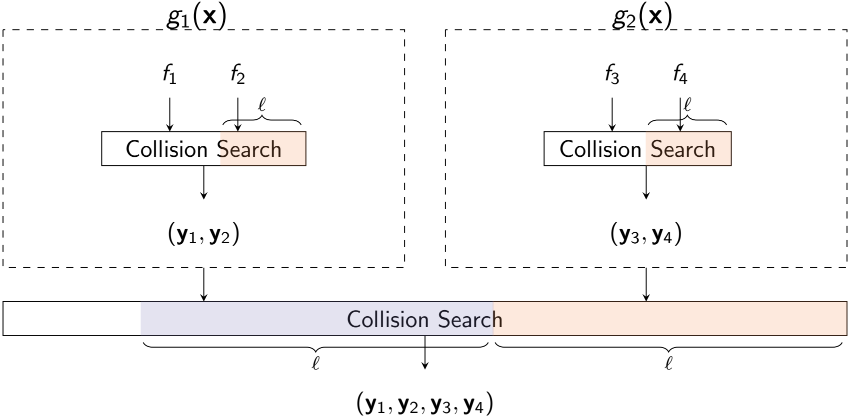
$$\underbrace{\pi_\ell(\mathbf{A}\mathbf{s}_3)}_{f_3(\mathbf{s}_3)} = \underbrace{\pi_\ell(\mathbf{b} + \mathbf{e}) - \mathbf{r} - \pi_\ell(\mathbf{A}\mathbf{s}_4)}_{f_4(\mathbf{s}_4)} \mod q.$$

Nested-Collision-Search

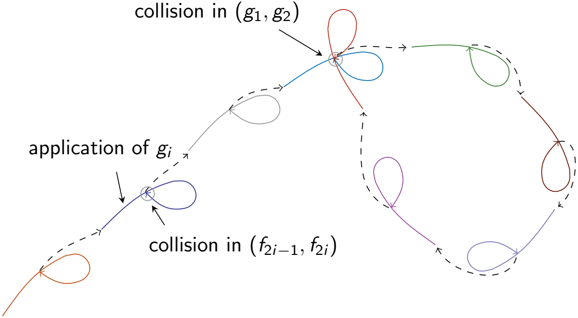
Let $\vartheta_\ell(\mathbf{x}) := (x_{\ell+1}, \ldots, x_{2\ell})$.

$$g_1 \colon \mathbf{x} \mapsto \vartheta_\ell(\mathbf{A}(\mathbf{y}_1 + \mathbf{y}_2)) \quad , \text{ where } (\mathbf{y}_1, \mathbf{y}_2) = \textsc{Rho}(f_1, f_2, \mathbf{x}) \quad \text{and}$$
$$g_2 \colon \mathbf{x} \mapsto \vartheta_\ell(\mathbf{b}') - \vartheta_\ell(\mathbf{A}(\mathbf{y}_3 + \mathbf{y}_4)), \text{ where } (\mathbf{y}_3, \mathbf{y}_4) = \textsc{Rho}(f_3, f_4, \mathbf{x}).$$

Now just search for collisions between g_1, g_2.

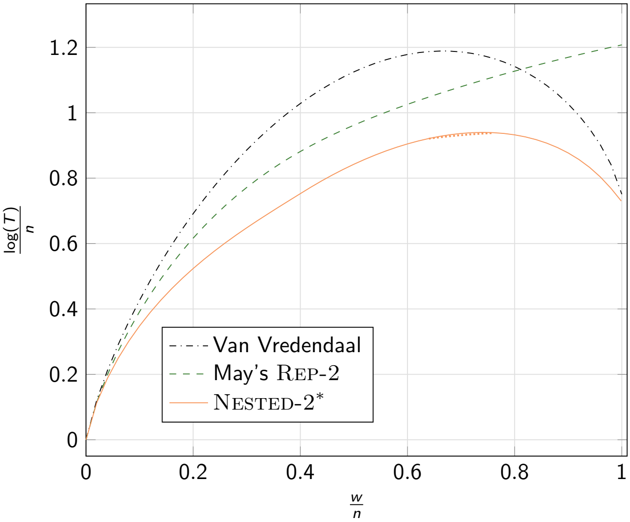


Time-Complexity of Nested-Collision-Search



$$T = \underbrace{\frac{\#\text{Collisions}}{\#\text{Representations}}}_{\text{Less}} \cdot \underbrace{T_{\text{Nested-Rho}}}_{\text{More}} = \boxed{\tilde{\mathcal{O}}\left(2^{0.8425n}\right)}$$

Comparing our Memoryless Algorithm

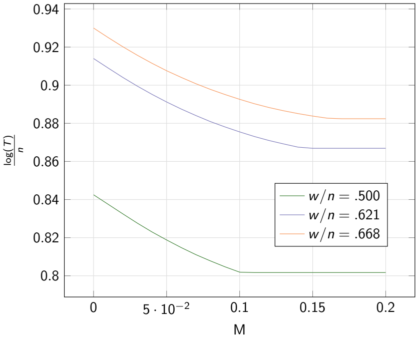


Time-Memory Trade-off using PCS

Theorem (Parallel Collision Search, Oorschot, Wiener 1999)

Finds M collisions between f_1 and f_2 using on expectation $\tilde{\mathcal{O}}\left(\sqrt{M \cdot |S|}\right)$ function evaluations and $\tilde{\mathcal{O}}(M)$ units of memory.

Maximum memory spent $\leq$ # needed collisions (small for optimal instantiation)
Idea: Incorporate the PCS speedup in Time complexity and re-optimize.



Conclusion

- Improved memoryless algorithm for ternary LWE keys.
- Time-memory Trade-off for a small exponential memory.
- More in the paper:
  - Adapting our technique for small max-norms 2 and 3.
  - Using techniques from information set decoding to improve the uniform-secrets case.

`https://eprint.iacr.org/2023/243`
`https://github.com/arindamIITM/Small-LWE-Keys`

Thank You.