Projeto de pesquisa de iniciação científica – PIBIC

Acelerando criptografia completamente homomórfica baseada em inteiros

Candidato: João Pedro Martins Leôncio Eusébio
Orientador: Hilder Vitor Lima Pereira

Instituto de Computação da Universidade Estadual de Campinas (IC-UNICAMP)

Março de 2024

Resumo: Criptografia completamente homomórfica (CCH) é uma poderosa primitiva criptográfica que permite que qualquer função seja calculada sobre dados cifrados. Construir CCH é altamente não trivial, pois é difícil achar um problema subjacente que ao mesmo tempo garanta a segurança da cifra e forneça propriedades algébricas suficientes para que a cifra seja homomórfica. Por isso, existem apenas três família de CCH: baseadas no problema *learning with errors* (LWE), no problema *NTRU*, e no problema *approximate greatest common divisor* (AGCD), sendo que esta última é conhecida como criptografia homomórfica baseada em inteiros. Primeiramente, as cifras homomórficas construídas com o LWE eram mais eficientes que as outras, assim, receberam maior atenção da comunidade acadêmica, que acabou propondo novos métodos para deixá-las ainda mais rápida, aumentando ainda mais a distância entre elas e as baseadas no NTRU e no AGCD. Logo, a cifra baseada no AGCD mais eficiente atualmente, chamada FHEZ, ainda é ordens de magnitude mais lenta que a mais eficiente baseada no LWE. Assim, esse projeto visa implementar FHEZ usando técnicas avançadas e modernas, que não estavam disponíveis quando FHEZ foi proposto, para acelerar FHEZ, aproximando assim CCH baseada em inteiros da baseada no LWE.

Research project – PIBIC

Accelerating fully homomorphic encryption over the integers

Student: João Pedro Martins Leôncio Eusébio
Advisor: Hilder Vitor Lima Pereira

Instituto de Computação da Universidade Estadual de Campinas (IC-UNICAMP)

March 2024

Resume: Fully homomorphic encryption (FHE) is a powerful criptografic primitive which allows any function to be calculated on encrypted data. Building a FHE is highly non-trivial, on the grounds that it is hard to find a subjacent problem that, simultaneously, ensures the security of the cipher and supplies sufficient algebric properties for the cipher to be homomorphic. Therefore, there are only three families of FHE: based on the *learning with erros* (LWE) problem, on NTRU problem, and on the *approximate greatest common divisor* (AGCD) problem, the latter being known as homomorphic encryption based on integers. Firstly, the homomorphic ciphers built with LWE were more efficient than the others, and thus, received more attention from the academic community, who proposed new methods to make it even faster, further increasing the distance between them and those based on NTRU and AGCD. Therefore, the current fastest cipher based on AGCD, namely FHEZ, is still orders of magnitude slower than the most efficient cipher based on LWE. That being said, this project aims to implement FHEZ using advanced and modern techniques, which were not available when FHEZ was proposed, to accelerate it, thus approaching integer-based FHE to that based on LWE.

1 Introdução

Criptografia moderna segue metodologias científicas formais, em que hipóteses são explicitamente feitas e então primitivas criptográficas têm suas propriedades provadas matematicamente baseadas nessas hipóteses. Por exemplo, usando como hipótese a intratabilidade computacional do problema conhecido como *learning with errors* (LWE) [Reg09], é possível provar que diversas cifras são seguras. Em particular, assumindo que o LWE é difícil, pode-se construir criptografia completamente homomórfica (CCH) [GSW13], que é uma primitiva criptográfica extremamente genérica e poderosa, com aplicações variando desde o campo prático, como na implementação de redes neurais artificiais privativas [LKL$^+$22], até o campo teórico, como na construção de novas primitivas, como obfuscação indistinguível [GP21].

Essas hipóteses que constituem as bases das primitivas criptográficas estão sempre sendo testadas, colocadas à prova, o que, normalmente, aumenta a confiança nelas. Ainda tomando o LWE como exemplo, cada vez que esse problema é analisado minuciosamente e nenhum algoritmo eficiente para resolvê-lo é encontrado, cresce a certeza de que esse é realmente um problema difícil. No entanto, já houve casos em que hipóteses de segurança que acreditavam-se ser verdadeiras foram demonstradas como falsas, o que invalidou todas as contruções baseadas em tais hipóteses. Um exemplo recente que ganhou muita atenção foi o problema computacional usado para construir o protocolo de troca de chaves baseados em isogenias em curvas elípticas supersingulares conhecido como SIKE. Após anos de análises, acreditava-se que se tratava realmente de um problema computacionalmente difícil, inclusive para computadores quânticos. No entanto, para a surpresa da comunidade de criptólogos, Castryck e Decru descobriram um algoritmo que resolve esse problema em tempo polinomial em um computador clássico [CD23]. Com isso, SIKE se tornou totalmente inseguro e novos protocolos devem ser estudados para substituí-lo.

Por conta desse constante estado de testes de hipóteses, desenvolveu-se em criptografia um conceito importante conhecido como variabilidade de hipóteses de segurança: em suma, é necessário ter várias construções diferentes de uma mesma primitiva criptográfica, cada construção baseada em um problema computacional diferente. Por exemplo, mesmo que já haja um protocolo de troca de chaves padronizado e amplamente utilizado, que seja construído sobre o problema da fatoração de inteiros, ainda é extremamente necessário construir outros protocolos de troca de chaves baseados em outros problemas, como o problema do logaritmo discreto, o LWE, etc. Assim, se eventualmente um algoritmo eficiente para o problema da fatoração de inteiros for descoberto, haverá protocolos de troca de chave para substituir o que é comumente usado. Essa variabilidade de hipóteses foi o que guiou o NIST[1] na criação de mais um processo de padronização de para assinaturas digitais pós-quânticas, já que o processo anterior acabou selecionando dois candidatos cujas hipóteses de segurança são relacionadas com reticulados estruturados (ou seja, problemas computacionais próximos do LWE). Na página da nova chamada[2], pode-se ler que o NIST estava interessado em outros tipos de problemas computacionais: "*NIST was primarily interested in additional general-purpose signature schemes that were not based on structured lattices*".

Note que o estudo de novas construções baseadas em diferentes problemas criptográficos vai além da possibilidade de existência de tais contruções, pois é preciso também que essas construções sejam eficientes e otimizadas, para que possam satisfatoriamente substituir umas as outras. Tendo isso em vista, em 2021, [Per21] propôs a primeira cifra completamente homomórfica baseada no problema *approximate greatest common divisor* (AGCD) que leva menos de 1 segundo para executar o *bootstrapping*, que é a operação principal em CCH. Na época, apenas CCH baseada no problema LWE tinha tal eficiência e não era claro que cifras homomórficas baseadas no AGCD poderiam ser tão

[1] *National Institute of Standards and Technology*, orgão estadunidense responsável por criar padrões, notadamente, para primitivas criptográficas.
[2] `https://csrc.nist.gov/news/2023/additional-pqc-digital-signature-candidates`

rápidas. No entanto, desde a publicação de [Per21], diversas técnicas para acelerar CCH foram propostas e muitas delas encorporadas às cifras homomórficas baseadas no LWE. Como resultado, existe atualmente uma grande diferença entre a eficiência de CCH baseada no LWE e de de CCH baseada no AGCD, ou como são comumente conhecidas, CCH baseada em inteiros, já que os tempos de execução da primeira família continuaram a melhorar, enquanto a segunda família de CCH ainda apresenta os mesmos tempos de execução obtidos em [Per21].

Portanto, motivado pelo princípio da variabilidade de hipóteses de segurança, esse projeto de pesquisa visa aplicar diversas técnicas de otimização de CCH sobre a cifra apresentada em [Per21], visando diminuir drasticamente a distância entre ela e as baseadas no LWE.

Será utilizada a linguagem Rust, pois permitirá o uso da nova biblioteca de transformada rápida de Fourier (FFT), que é escrita em Rust, e foi desenvolvida pela empresa Zama. A biblioteca se chama Concrete-FFT[3] e é a biblioteca padrão da reimplementação da cifra TFHE feita em Rust[4]. Note que CCH trabalha sobre polinômios, então, a FFT é uma peça fundamental em suas implementações e bibliotecas de FFT mais rápidas implicam CCH mais rápida.

Além disso, como os polinômios utilizados em [Per21] têm coeficientes com mais de 200 bits, a implementação de [Per21] disponível utiliza a biblioteca NTL[5] para inteiros de precisão arbitrária. No entanto, existem técnicas mais eficientes, como a representação conhecida como *double-CRT* [GPvL23], em que um polinômio com coeficientes grandes é transformado em um conjunto de polinômios com coeficientes de 32 ou 64 bits, assim, sendo possível representá-los com tipos de dados comuns em qualquer processador e linguagem de programação atuais. Isso elimina a necessidade de bibliotecas de inteiros grandes e acelera as operações polinômiais.

Finalmente, pretende-se utilizar novas técnicas de decomposição de criptogramas propostas em [BCG$^+$23], ou seja, depois da publicação de [Per21]. Isso deve trazer mais ganhos de desempenho.

2 Objetivos

Idealmente, as técnicas estudadas se mostrarão compatíveis com a cifra completamente homomórfica de [Per21], então, este projeto produzirá uma variante dessa cifra, que incluirá representações diferentes para os criptogramas, a saber, a representação conhecida como *double-CRT*. Além disso, essa variante também terá uma nova forma de decompor os criptogramas.

Após a descrição teórica da nova cifra homomórfica, ela será implementada em Rust. Logo, um resultado desse projeto será uma biblioteca moderna, disponibilizada em uma licença livre, como a GPL, em algum repositório aberto, em sites como o GitHub, para que qualquer pessoa interessada em criptografia homomórfica possa utilizá-la. Graças à combinação de melhorias teóricas com melhorias na implementação (como a utilização de uma biblioteca de FFT criada especificamente para cifras homomórficas), essa biblioteca terá tempos de execução muito menores do que a implementação atual de [Per21].

Adicionalmente, pretende-se escrever um artigo descrevendo a nova cifra obtida nesse projeto e a sua implementação, além dos tempos de execução e comparações com o estado da arte.

[3] `https://github.com/zama-ai/concrete-fft`

[4] `https://docs.zama.ai/tfhe-rs`

[5] `https://libntl.org/`

3 Métodos

A implementação de uma nova versão da cifra homomórfica FHEZ[6] tem como etapas principais o estudo de novas técnicas usadas em CCH e das bibliotecas existentes, a implementação e uma fase de testes, verificação e comparação.

Portanto, o projeto se iniciará com o estudo teórico de técnicas modernas utilizadas em implementações recentes de CCH (i.e., técnicas posteriores à publicação de FHEZ), visando identificar se são compatíveis com CCH sobre inteiros e se realmente podem trazer ganhos de eficiência à FHEZ. Para isso, serão estudados artigos relevantes da área, como [BCG$^+$23], que propõe uma nova técnica de decomposição de criptogramas e [GPvL23], onde a representação conhecida como *double-CRT* é apresentada.

Já se sabe que essa representação não é compatível diretamente com CCH sobre inteiros, pois ela assume que os criptogramas são definidos sobre um anel do tipo $R_Q = \mathbb{Z}_Q[X]/\langle X^N + 1\rangle$, ou seja, anel de polinômios em X de grau menor que N e coeficientes definidos módulo Q. Além disso, assume-se $Q = \prod_{i=0}^{L} q_i$, onde $q_0, q_1, ..., q_L$ são primos diferentes tais que o polinômio $X^N + 1$ pode ser fatorado completamente (i.e., em termos lineares) módulo cada q_i. No entanto, a cifra construída em [Per21], i.e., FHEZ, trabalha sobre o anel $R = \mathbb{Z}[X]/\langle X^N + 1\rangle$, ou seja, em vez de coeficientes módulo Q, ela usa coeficientes inteiros (o que justifica o nome "CCH sobre inteiros"). Então, um desafio desse projeto é verificar se podemos utilizar o *double-CRT* neste caso.

Uma direção possível para superar esse desafio, é primeiro encontrarmos um limitante superior para o tamanho dos coeficientes. Por exemplo, digamos que, uma vez escolhido o nível de segurança e todos os parâmetros da cifra de [Per21], os criptogramas sejam compostos por polinômios cujos coeficientes tenham 350 bits, então, o limitante superior é 2^{350}. Em seguida, podemos definir um inteiro Q tal que ele é um produto de primos, como antes, e também é maior que o limitante superior encontrado, neste exemplo, $Q > 2^{350}$ e $Q = \prod_{i=0}^{L} q_i$, permitindo assim que os coeficientes inteiros dos polinômios utilizados FHEZ possam ser representados com o *double-CRT*.

Uma vez identificadas as novas técnicas que serão utilizadas na nova versão do esquema FHEZ, começará a implementação, em Rust. Para isso, tanto a nova biblioteca de FFT específica para criptografia homomórfica, i.e., concrete-FFT quanto implementações existentes de cifras homomórficas similares serão estudadas. Em seguida, um *back end* aritmético será implementado, ou seja, as operações sobre o anel, como somas e multiplicações polinomiais utilizando a biblioteca FFT. Então, utilizando esse *back end*, os blocos fundamentais que constituem a cifra FHEZ serão implementados e testados (e.g., função que cifra, função que decifra e função que executa multiplicação homomórfica). Para cada operação do *back end* e para cada bloco, serão escritos testes unitários. Então, esses blocos serão combinados com o intuito de construir a cifra completamente. Também serão implementados testes para a cifra em si, cobrindo geração de chaves, cifração, decifração e o *bootstrapping*.

Uma vez verificada que a implementação está correta, serão feitos experimentos para coletar tempos de execução e consumo de memória da nova versão do FHEZ. Esses mesmos experimentos serão executados com a versão existente de FHEZ e com outras cifras homomórficas similares, como FHEW [DM15], TFHE [CGGI19] e FINAL [BIP$^+$22], para que todos esses resultados possam ser comparados entre si.

Com o intuito de garantir a reprodutibilidade, todo o código produzido neste projeto, incluindo a cifra, os testes e os experimentos, serão disponibilizados publicamente em um repositório como o GitHub.

Por fim, será escrito um artigo descrevendo como o estudo foi conduzido, as técnicas aplicadas, os resultados obtidos e como eles se comparam ao estado da arte.

[6] `https://github.com/hilder-vitor/FHEZ`

	1º	2º	3º	4º
Bagagem teórica				
Levantamento bibliográfico				
Levantamento ferramental				
Desenvolvimento da nova cifra				
Implementação				
Verificação				
Redação de artigos				

Tabela 1: Cronograma previsto para o projeto de iniciação científica. Cada coluna representa um trimestre. Cores mais escuras representam uma carga maior de trabalho.

4 Cronograma

Este projeto tem duração prevista de um ano e sua execução será organizada da seguinte forma: No primeiro trimestre, o aluno estudará o artigo [Per21], para formar uma boa base teórica para o desenvolvimento do projeto. No segundo trimestre, estudará especificamente a técnica conhecida como *double-CRT* e já começará a implementar algumas operações usando essa técnica, para se familiarizar com ela e já ir planejando como integrá-la à cifra [Per21]. Principalmente, se iniciará a implementação da representação *double-CRT* com a biblioteca concrete-FFT. O terceiro semestre se concentrará na implementação completa da cifra de [Per21] em Rust, usando *double-CRT* com a biblioteca concrete-FFT. O último semestre abordará as novas formas de decompor os criptogramas descritos em [BCG$^+$23]. As atividades são apresentadas de forma detalhada na Tabela 1 e descritas na seguinte lista:

1. Bagagem teórica geral: aprender mais sobre criptografia homomórfica. Visto que o aluno já tem um bom conhecimento sobre este tópico, essa parte demandará pouco trabalho.

2. Levantamento bibliográfico: estudar os artigos principais para esse projeto, a saber, [Per21], que constrói CCH sobre inteiros; [BCG$^+$23], que propõe uma nova técnica de decomposição de criptogramas, e [GPvL23], que explica em detalhes o *double-CRT*.

3. Levantamento ferramental: pesquisar, analisar e testar implementações de esquemas de criptografia homomórfica, como a implementação original de [Per21] e a reimplementação de TFHE, chamada ZAMA TFHE-rs. E também estudar as bibliotecas que serão usadas, principalmente, a concrete-FFT.

4. Reimplementação: implementar em Rust a cifra de [Per21] com as melhorias descritas anteriormente.

5. Verificação: execução de testes para verificar a corretude, tempos de execução e consumo de memória, além de obter comparações com o estado da arte.

6. Redação de artigos: após obtenção dos resultados, o aluno de IC participará da redação de artigos científicos.

5 Disseminação e avaliação

A principal forma de disseminação do conhecimento obtido neste projeto será a publicação, em alguma licença de código livre, de uma biblioteca implementando a nova cifra, mais eficiente. Além disso, pretende-se escrever um artigo para ser submetido a alguma conferência de criptografia, como a SBSeg, ACNS, Crypto, Eurocrypt, Asiacrypt, e PKC.

Referências

[BCG+23] Mariya Georgieva Belorgey, Sergiu Carpov, Nicolas Gama, Sandra Guasch, and Dimitar Jetchev. Revisiting key decomposition techniques for fhe: Simpler, faster and more generic. *Cryptology ePrint Archive*, 2023.

[BIP+22] Charlotte Bonte, Ilia Iliashenko, Jeongeun Park, Hilder V. L. Pereira, and Nigel P. Smart. FINAL: Faster FHE instantiated with NTRU and LWE. Asiacrypt – Taipei, Taiwan, 2022. `https://ia.cr/2022/074`.

[CD23] Wouter Castryck and Thomas Decru. An efficient key recovery attack on sidh. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 423–447, Cham, 2023. Springer Nature Switzerland.

[CGGI19] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast Fully Homomorphic Encryption Over the Torus. *Journal of Cryptology*, Apr 2019.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: Bootstrapping Homomorphic Encryption in Less Than a Second. In *Advances in Cryptology – EUROCRYPT 2015*, pages 617–640, Berlin, Heidelberg, 2015. Springer.

[GP21] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 736–749, 2021.

[GPvL23] Antonio Guimarães, Hilder V. L. Pereira, and Barry van Leeuwen. Amortized bootstrapping revisited: Simpler, asymptotically-faster, implemented. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023*, pages 3–35, Singapore, 2023. Springer Nature Singapore.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. In *Advances in Cryptology – CRYPTO 2013*, pages 75–92, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[LKL+22] Joon-Woo Lee, HyungChul Kang, Yongwoo Lee, Woosuk Choi, Jieun Eom, Maxim Deryabin, Eunsang Lee, Junghyun Lee, Donghoon Yoo, Young-Sik Kim, et al. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. *IEEE Access*, 10:30039–30054, 2022.

[Per21] Hilder Vitor Lima Pereira. Bootstrapping fully homomorphic encryption over the integers in less than one second. In *IACR International Conference on Public-Key Cryptography*, pages 331–359. Springer, 2021.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), sep 2009.