Capítulo 4

Encriptação homomórfica

Autores: Eduardo Morais e Ricardo Dahab
Apresentador: Eduardo Morais
Instituição: Universidade Estadual de Campinas (UNICAMP)

Abstract

In 1978, Rivest, Adleman and Dertouzos [RAD78] suggested the construction of privacy homomorphisms as a mechanism to protect computation on secret data. The problem remained open until 2009 [Gen09a], when Craig Gentry proposed the use of ideal lattices to construct a fully homomorphic cryptosystem.

Privacy homomorphisms are an interesting solution in a cloud computing scenario, but unfortunately it is not efficient enough to be used in practice. However many concrete improvements have been studied recently, encouraging people to look for better algorithms. In this course Craig Gentry's breakthrough will be carefully studied, presenting the state of the art and pointing out the problems that remain to be solved.

Resumo

Em 1978, Rivest, Adleman e Dertouzos [RAD78] sugeriram a construção de ***homomorfismos secretos*** - privacy homomorphisms - *como forma de prover um mecanismo de proteção para computação sobre dados sigilosos. O problema permaneceu em aberto até recentemente, quando em 2009 Craig Gentry [Gen09a] o resolveu sugerindo a utilização de reticulados ideais na construção de um criptossistema completamente homomórfico.*

Infelizmente a proposta de Craig Gentry não é suficientemente eficiente para ser usada na prática, mas inúmeros trabalhos têm contribuído para que a eficiência dos algoritmos se torne cada vez maior. Neste minicurso serão estudados os esquemas recentemente propostos por Craig Gentry, apresentando o estado da arte e analisando os problemas que ainda precisam ser resolvidos.

4.1. Introdução

É grande a quantidade de aplicações em nuvem que estão surgindo recentemente e a segurança deste novo modelo de computação ainda é uma questão em aberto. O NIST [MG09] define computação em nuvem, descrevendo três categorias distintas: (i) **software como um serviço** (SaaS - *Software as a Service*) (ii) **plataforma como um serviço** (PaaS - *Platform as a Service*) e (iii) **infraestrutura como um serviço** (IaaS - *Infrastructure as a Service*). Nestes três modelos, a segurança pode ser obtida por meio do uso de criptografia. Além disso, a utilização de uma base de computação confiada (TCB - *Trusted Computing Base*) para gerenciamento de distribuição de chaves, permite a criação de canais seguros entre um cliente e um provedor de serviço em nuvem. Assim, a informação sigilosa pode ser protegida contra um adversário que intercepta as mensagens pela rede. Porém, esta informação ainda pode ser acessada pelo provedor do serviço, o que é uma ameaça em diversos cenários, como por exemplo no caso de informações médicas confidenciais, dados bancários, ou qualquer informação que fira o direito de privacidade de uma pessoa. Portanto, é clara a necessidade da **criptografia como um serviço** (CaaS), podendo ser utilizada nos três modelos de computação em nuvem para prover requisitos como sigilo, autenticidade, integridade e irretratabilidade.

A construção de algoritmos eficientes para obtenção de **homomorfismos secretos** - *privacy homomorphisms*, como apresentado por Rivest, Adleman e Dertouzos em 1978 [RAD78], é um objetivo muito cobiçado da criptografia moderna, porque permite a construção de *killer applications*. Estas aplicações podem ser usadas para prover CaaS; podemos citar, por exemplo, as seguintes:

- **banco de dados encriptados**, a nuvem manipula os dados encriptados e retorna para o usuário decriptar;
- **disco rígido encriptado**, semelhante ao caso anterior, a nuvem não consegue obter informação confidencial do disco, mas consegue manipulá-lo;
- **mecanismo de buscas encriptado**, permite buscas na internet sem revelar informação sobre o que está sendo buscado;
- **computação sobre dados encriptados**, permite delegar o processamento de um programa à nuvem, que computa sobre os dados encriptados e portanto não tem acesso à informação sigilosa;
- **ofuscação**, embora seja impossível obter ofuscação em um modelo de segurança rígido, veremos como é possível usar homomorfismo secreto para construir um esquema de ofuscação sob a hipótese de uso de um hardware resistente a ataques laterais.

4.1.1. Organização deste documento

O minicurso está organizado da seguinte maneira: no restante desta primeira seção serão apresentados alguns fundamentos matemáticos e definições preliminares, também serão mostradas algumas propostas anteriores a construção de Craig Gentry; na seção 2 serão definidos os conceitos básicos, assim como o criptossistema sobre números inteiros de

Craig Gentry; na seção 3 será descrito o esquema sobre reticulados ideais; na seção 4 são apresentadas as otimizações, em especial o esquema BGV; na seção 5 são discutidos esquemas práticos que podem ser implementados a partir de criptossistemas parcialmente homomórficos; na seção 6 serão feitas as considerações finais e alguns exercícios são propostos na seção 7.

4.1.2. Fundamentos matemáticos

Nesta seção serão discutidos brevemente os fundamentos matemáticos necessários para compreender as construções que serão realizadas nas seções futuras.

4.1.2.1. Circuitos

Um ***circuito booleano*** é um conjunto de portas lógicas conectadas por fios. Formalmente, pode ser modelado por um grafo orientado acíclico. O circuito recebe de entrada um conjunto de variáveis booleanas, que são processadas pelas portas lógicas, gerando um conjunto de variáveis booleanas como saída. A profundidade do circuito é a distância entre as entradas e as saídas do circuito. O tamanho do circuito é a quantidade de arestas do grafo.

Informalmente, o modelo de computação baseado em circuitos booleanos é equivalente ao modelo da máquina de Turing se fixarmos um limite para o tamanho dos circuitos. Podemos, assim, na prática, considerá-lo um modelo completo, capaz de computar um algoritmo arbitrário. Da mesma forma, ***circuitos algébricos*** também correspondem a um modelo computacional completo. Logo, a obtenção de uma função que, simultaneamente, seja um homomorfismo e também possa ser usado para criptografia, significa que é possível somar e multiplicar textos encriptados, e portanto é possível computar circuitos algébricos de maneira homomórfica. Como veremos adiante, algumas condições são necessárias para que esta construção seja segura e eficiente.

4.1.2.2. Álgebra abstrata

As construções que serão descritas adiante, farão uso de conceitos matemáticos da álgebra abstrata. Nesta seção serão apresentados alguns destes conceitos de maneira bastante breve. Existem diversos livros que podem ser usados para obter uma compreensão mais aprofundada do assunto, como por exemplo o de Dummit e Foot [DF04].

Um ***anel*** é uma estrutura matemática composta de um conjunto R e duas operações (geralmente $+$ e $\times$), tal que $(R,+)$ é um grupo abeliano, e as operações $+$ e $\times$ estão relacionadas pela propriedade distributiva. Em geral, temos um elemento neutro multiplicativo, mas nem todo elemento de R possui inverso multiplicativo.

Um ***ideal*** I sobre R é um subconjunto fechado em R, de modo que a multiplicação de elementos $i \in I$, por elementos $r \in R$, permanece no subconjunto I, ou seja, $i \times r \in I$. Um exemplo de ideal sobre o anel de inteiros $\mathbb{Z}$ é o subconjunto $p\mathbb{Z}$, para qualquer inteiro p, já que a multiplicação de qualquer múltiplo de p por um elemento arbitrário $r \in R$ continua sendo um múltiplo de p.

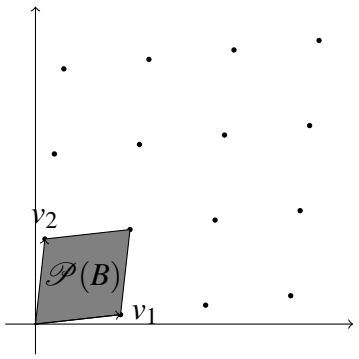

Figura 4.1.

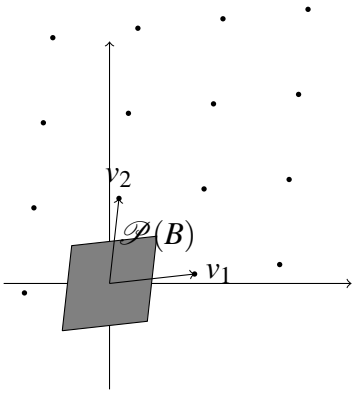

Figura 4.2.

Dados dois anéis R e S, um ***homomorfismo*** h é uma função entre os anéis, que preserva as operações de soma e multiplicação. Isto é, $h(r_1 + r_2) = h(r_1) + h(r_2)$ e $h(r_1 \times r_2) = h(r_1) \times h(r_2)$. Além disso, os elementos neutros aditivo e multiplicativo em R são mapeados nos elementos neutros respectivos em S.

4.1.2.3. Reticulados

Reticulados são combinações lineares *inteiras* de n elementos $b_1, ..., b_n \in \mathbb{R}^n$, linearmente independentes, denominados ***base do reticulado***.

$$\mathscr{L}(b_1, ..., b_n) = \left\{ \sum_{i=1}^{n} x_i b_i : x_i \in \mathbb{Z} \right\}.$$

Em outras palavras, um reticulado é um espaço vetorial discretizado, ou seja, existe uma analogia que nos permite utilizar conceitos como norma, dimensão, ortogonalidade, transformação linear, entre outros. Uma maneira alternativa de abordar o assunto é por meio de notação matricial, onde a base pode ser representada por uma matriz $B = [b_1, ..., b_n]$, pertencente a $\mathbb{R}^{n \times n}$. O reticulado gerado pela matriz B é definido por $\mathscr{L} = \{Bx \mid x \in \mathbb{Z}^n\}$, de forma que o determinante $\det(B)$ é independente da escolha da base e corresponde geometricamente ao inverso da densidade de pontos do reticulado em $\mathbb{Z}^n$.

Dado um reticulado $\mathscr{L}(B)$, os vetores que constituem a base do reticulado são arestas de um paralelepípedo de dimensão n. Assim, podemos definir $\mathscr{P}(B) = \{Bx \mid x \in [0,1)^n\}$, denominado ***paralelepípedo fundamental*** de B. Podemos alternativamente definir $\mathscr{P}_{1/2}(B)$ de forma a obter uma região simétrica. Para isso, seja $\mathscr{P}_{1/2}(B) = \{Bx \mid x \in [-1/2, 1/2)^n\}$, denominado ***paralelepípedo fundamental centralizado*** de B. A figura 4.1 ilustra um exemplo de paralelepípedo fundamental em dimensão 2, enquanto a figura 4.2 representa um paralelepípedo fundamental centralizado.

Seja $\mathscr{L} \in \mathbb{R}^n$ um reticulado de dimensão n e seja $\mathscr{F}$ o paralelepípedo fundamental de $\mathscr{L}$, então dado um elemento $w \in \mathbb{R}^n$, podemos escrever w na forma $w = v + t$, para

$v \in \mathscr{L}$ e $t \in \mathscr{F}$ únicos. Esta equação equivale a uma redução modular, onde o vetor t é interpretado como $w \pmod{\mathscr{F}}$.

O volume do paralelepípedo fundamental é dado por $\mathrm{Vol}(\mathscr{F}) = |\det(B)|$. Dadas duas bases $B = \{b_1, ..., b_n\}$ e $B' = \{b'_1, ..., b'_n\}$ de um mesmo reticulado $\mathscr{L}$, temos que $\det(B) = \pm\det(B')$.

O problema de encontrar o vetor de norma mínima (*shortest vector problem* - SVP) é uma das questões fundamentais em reticulados. Rigorosamente, dado o reticulado $\mathscr{L}(B)$, deseja-se encontrar o vetor não nulo com norma mínima. Na prática, é utilizado um fator de aproximação $\gamma(n)$ para o problema SVP, isto é, deseja-se encontrar um vetor cuja norma seja inferior ao vetor de norma mínima, multiplicado por $\gamma(n)$.

Outros problemas em reticulados, importantes do ponto de vista da criptografia, são:

- o ***problema do vetor de distância mínima*** (*closest vector problem* - CVP). Dados um reticulado $\mathscr{L}(B)$ e um vetor $t \in \mathbb{R}^m$, o objetivo é encontrar o vetor $v \in \mathscr{L}(B)$ que seja mais próximo de t;

- e o ***problema dos vetores independentes mínimos*** (*shortest independent vector problem* - SIVP). Dada uma base $B \in \mathbb{Z}^{n \times n}$, o problema consiste em encontrar n vetores linearmente independentes $(v_1, ..., v_n)$, pertencentes ao reticulado, tais que a norma máxima entre os vetores v_i seja mínima.

O criptossistema GGH [GGH97] utiliza o conceito de ortonormalidade da base na definição do par de chaves. A chave privada é definida como uma base B_{priv} do reticulado, formada por vetores quase ortogonais e com norma próxima a 1. Desta forma, antes de prosseguir, é preciso uma forma de medir o grau de ortonormalidade de uma determinada base.

Dado um reticulado $\mathscr{L}$ e uma base $B = (v_1, ...v_n)$, a ***razão de Hadamard***, denotada por $\mathscr{H}(B)$, é definida da seguinte maneira:

$$\mathscr{H}(B) = \left(\frac{\det \mathscr{L}}{||v_1||...||v_n||} \right)^{1/n}.$$

É fácil mostrar que para qualquer base B, temos que $0 \leq \mathscr{H}(B) \leq 1$. Além disso, quanto mais próximo de 1 mais ortonormal é a base. De modo geral, o criptossistema GGH funciona da seguinte maneira:

- o algoritmo de encriptação acrescenta o ruído $r \in \mathbb{R}^n$ ao texto claro $m \in \mathscr{L}$, gerando o texto encriptado $c = m + r$;

- o algoritmo de decriptação precisa ser capaz de retirar o ruído inserido. Alternativamente, é preciso resolver uma instância do problema CVP.

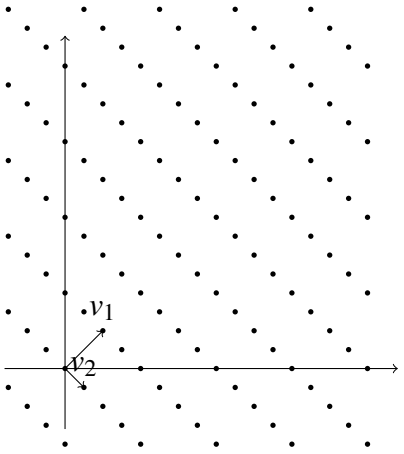

Figura 4.3. Base boa

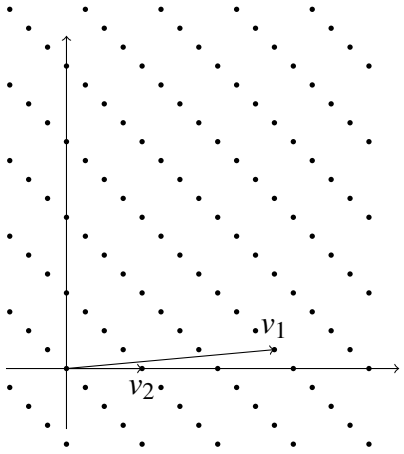

Figura 4.4. Base ruim

A figura 4.3 mostra um reticulado em dimensão 2, com base dada pelos vetores v_1 e v_2, praticamente ortogonais.

Porém, conforme menor a ortonormalidade da base conhecida e maior a dimensão do reticulado, mais difícil é o problema CVP. Desta forma, a chave pública pode ser definida por uma base B_{pub} do reticulado, tal que $\mathscr{H}(B_{\text{pub}})$ seja aproximadamente zero. Por outro lado, com o conhecimento da chave privada B_{priv}, o algoritmo de Babai [Bab86], descrito a seguir, pode ser utilizado para recuperar o texto claro.

Algoritmo 4.1.1 Algoritmo de Babai

ENTRADA o reticulado $\mathscr{L}$ de dimensão n; o vetor $c = (c_1, ..., c_n)$, onde $c_i \in \mathbb{R}$; e uma base $B_{\text{priv}} = (s_1, ..., s_n)$, suficientemente ortonormal.

SAÍDA o vetor $m \in \mathscr{L}$ que resolve o problema CVP com relação a c e $\mathscr{L}$.

Resolva um sistema de n equações, $c = t_1 s_1 + ... + t_n s_n$, nas variavéis t_i, onde $1 \leq i \leq n$.

para $i = 0$ até $i = n$ **faça**

$a_i \leftarrow \lfloor t_i \rceil$

retorne $m \leftarrow a_1 s_1 + ... a_n s_n$

A ideia geral do algoritmo de Babai é representar o vetor c na base privada B_{priv}, resolvendo um sistema de n equações lineares. Como $c \in \mathbb{R}^n$, para obter um elemento do reticulado $\mathscr{L}$, cada coeficiente $t_i \in \mathbb{R}$ é aproximado para o inteiro mais próximo a_i, onde esta operação de arredondamento é denotada por $a_i \leftarrow \lfloor t_i \rceil$. Este procedimento simples funciona bem desde que a base B_{priv} seja suficientemente ortonormal, reduzindo os erros do arredondamento.

4.1.3. De 1976 até 2009

No modelo de criptografia convencional (criptografia simétrica), Alice e Bob compartilham uma única chave k, gerada por um algoritmo KeyGen, com a qual podem comunicar-se de forma segura. São definidos os domínios $\mathscr{K}$, $\mathscr{P}$ e $\mathscr{C}$ como sendo respectivamente o espaço de chaves e o espaço de texto claro e texto encriptado. Além disso, são definidos algoritmos de encriptação Enc : $\mathscr{P} \times \mathscr{K} \rightarrow \mathscr{C}$ e de decriptação

$\text{Dec} : \mathscr{C} \times \mathscr{K} \to \mathscr{P}$, tal que $\text{Dec}(k, \text{Enc}(k,m)) = m$. De agora em diante, denominamos o conjunto $\mathscr{E} = \{\text{KeyGen}, \text{Enc}, \text{Dec}\}$ um ***esquema de encriptação simétrica*** ou então um ***criptossistema de chave privada***. Existe um problema imediato neste modelo: a quantidade de chaves que precisam ser gerenciadas é quadrática, isto é, em um grupo com n pessoas são necessárias $n(n-1)/2$ chaves para tornar possível a comunicação de quaisquer 2 pessoas deste grupo, de forma que o gerenciamento dessas chaves compartilhadas é um obstáculo a ser superado.

Em 1976, Diffie e Hellman [DH76] publicaram o artigo *New directions in cryptography*, introduzindo o conceito de criptografia de chave pública (criptografia assimétrica). Neste modelo, Alice utiliza o algoritmo KeyGen para gerar um par chaves $(\text{sk}_A, \text{pk}_A) \in \mathscr{K}_{\text{pub}} \times \mathscr{K}_{\text{priv}}$. A chave privada sk_A deve ser mantida em segredo enquanto a chave pública pk_A deve ser divulgada de alguma maneira. Os algoritmos de encriptação e decriptação são definidos respectivamente por $\text{Enc} : \mathscr{P} \times \mathscr{K}_{\text{pub}} \to \mathscr{C}$ e $\text{Dec} : \mathscr{C} \times \mathscr{K}_{\text{priv}} \to \mathscr{P}$, tal que $\text{Dec}(\text{sk}, \text{Enc}(\text{pk}, m)) = m$, para (sk, pk) um par de chaves válido, isto é, gerado por KeyGen. Com estas características, $\mathscr{E} = \{\text{KeyGen}, \text{Enc}, \text{Dec}\}$ é denominado ***esquema de encriptação assimétrica*** ou então ***criptossistema de chave assimétrica***.

Neste mesmo artigo, Diffie e Hellman propuseram um algoritmo que utiliza o par de chaves de Alice e Bob para estabelecer uma chave secreta adequada para criptografia convencional. Dado o grupo G, tal que $|G| = n$ e um gerador g deste grupo. O algoritmo KeyGen gera $a \in [0,n)$ aleatoriamente, calcula $A \equiv g^a \pmod{n}$ e retorna $(\text{sk}_A, \text{pk}_A) = (a, A)$ para Alice. Analogamente, Bob obtém como par de chaves os valores (b, B), com $b \in [0,n)$ escolhido aleatoriamente e $B \equiv g^b \mod n$. Alice usa a chave pública de Bob, B e sua chave privada a para calcular

$$B^a = (g^b)^a = g^{ab} \pmod{n}.$$

Similarmente, Bob usa a chave pública de Alice, A, e a sua chave privada b para calcular

$$A^b = (g^a)^b = g^{ab} \pmod{n}.$$

Desta maneira Alice e Bob conseguem computar um valor em comum, que pode ser utilizado como chave secreta no modelo de criptografia simétrica. De fato, estavam sugerindo uma nova forma de criptografia, sem dizer quais algoritmos e estruturas matemáticas satisfariam o novo modelo e, além disso, estavam resolvendo o problema de acordo de chaves do modelo antigo.

Dois anos depois, em 1978, Rivest, Shamir e Adleman [RSA83] resolveram o problema desenvolvendo o primeiro criptossistema de chave pública, o RSA, usando uma ideia bem parecida com a que foi apresentada no acordo de chaves de Diffie e Hellman. Resumidamente, $n = p.q$, onde p e q são primos grandes. O algoritmo KeyGen retorna o par (d,e), tal que $d.e \equiv 1 \pmod{\phi(n)}$. O algoritmo de encriptação computa $c = \text{Enc}(m,e) = m^e \pmod{n}$, enquanto o algoritmo de decriptação computa $\text{Dec}(c,d) = c^d \pmod{n}$. A corretude é garantida porque $\text{Dec}(\text{Enc}(m,e),d) = \text{Dec}(m^e \pmod{n}, d) = m^{e.d} \pmod{n} \equiv m \pmod{n}$.

Em especial, dados dois textos encriptados $c_1 = \text{Enc}(m_1, e)$ e $c_2 = \text{Enc}(m_2, e)$, temos que $c_1.c_2 = m_1^e.m_2^e = (m_1.m_2)^e \pmod{n}$. Em geral, dados k textos encriptados $c_1, ..., c_k$, temos que $\prod c_i = \text{Enc}(\prod m_i, e)$. Assim, o RSA preserva a estrutura da operação de multiplicação e uma pergunta natural que surge é sobre a possibilidade de obter um esquema que preserve ambas as operações de soma e multiplicação. Matematicamente, uma função assim é denominada ***homomorfismo***.

Ainda em 1978, Rivest, Adleman e Dertouzos [RAD78] definiram o conceito de ***homomorfismos secretos*** - *privacy homomorphisms* - como sendo um mapeamento entre sistemas algébricos, compostos por operações, predicados e constantes (preservados pelo mapeamento). Em outras palavras, é um esquema $\mathscr{E} = \{\text{KeyGen}, \text{Enc}, \text{Dec}, \text{Eval}\}$, onde o algoritmo Eval é capaz de avaliar circuitos algébricos de um domínio permitido, denotado por $\mathbf{S_C}$, compostos pelas operações de soma e multiplicação sobre textos encriptados. Ou seja, $\text{Eval} : \mathscr{K}_{\text{pub}} \times \mathbf{S_C} \times \mathscr{C}^n \to \mathscr{C}$, tal que para cada circuito $\mathbf{C} \in \mathbf{S_C}$, se $\Psi = \langle \psi_1, ..., \psi_n \rangle$ são textos encriptados tais que $\psi_i = \text{Enc}(\text{pk}, m_i)$, então temos que $m = \mathbf{C}(m_1, ..., m_n)$ e $m = \text{Dec}(\text{sk}, \text{Eval}(\text{pk}, \mathbf{C}, \Psi))$. O conjunto de algoritmos $\mathscr{E} = \{\text{KeyGen}, \text{Enc}, \text{Dec}, \text{Eval}\}$ é denominado ***criptossistema completamente homomórfico (CCH)***, se $\mathbf{S_C}$ for equivalente ao conjunto de todos os circuitos booleanos. Formalmente é necessário estabelecer condições para que o criptossistema seja prático. Por exemplo, o texto encriptado não pode crescer muito em comparação com o tamanho do circuito que desejamos avaliar. Além disso. os algoritmos de geração de chaves, encriptação, decriptação e avaliação precisam ter complexidade polinomial em relação ao parâmetro de segurança. Estes detalhes serão definidos na seção 4.2.2. Uma nomenclatura alternativa para CCH é ***encriptação completamente homomórfica (ECH)***.

Como vimos, a ideia básica do ***RSA*** [RSA83] pode ser utilizada para construir um criptossistema parcialmente homomórfico, preservando a multiplicação, pois dados os textos encriptados $c_1 = m_1^e \pmod{n}$ e $c_2 = m_2^e \pmod{n}$, é possível computar $c_1.c_2 = (m_1.m_2)^e \pmod{n}$. No próprio artigo de Rivest, Adleman e Dertouzos [RAD78], são propostos esquemas para criação de homomorfismo secreto, mas todos eles foram quebrados.

Uma propriedade importante para a construção de um homomorfismo secreto é a segurança semântica. Se temos conhecimentos de um conjunto $M = \{m_1, m_2, ..., m_k\}$ de textos claros e desejamos saber se um determinado texto encriptado c corresponde a algum m_i e se o algoritmo de encriptação for determinístico, então basta encriptar cada um dos m_i's e comparar o resultado com c. Para ter segurança semântica, um esquema criptográfico deve estar protegido contra este tipo de ataque, e portanto o algoritmo de encriptação deve ser aleatorizado, ou seja, a cada vez que é executado, um novo texto encriptado é gerado, diferente do anterior (com grande probabilidade).

O RSA é um criptossistema determinístico, portanto não possui segurança semântica. Assim, o ***ElGamal*** é uma alternativa imediata, por não ser determinístico e oferecer homomorfismo multiplicativo como o RSA. Dado um número primo p grande, um gerador g do grupo multiplicativo $\mathbb{Z}_p$, a chave secreta de Alice é um valor a escolhido aleatoriamente entre 0 e $p-1$. A chave pública é dada por $A = g^a \pmod{p}$. Dada uma mensagem $m \in \mathbb{Z}_p$, e um inteiro aleatório k entre 0 e $p-1$, computa-se o texto encriptado como $(c_1, c_2) = (g^k, A^k.m)$. Para decriptar, Alice calcula $m = (c_1^a)^{-1}.c_2 \pmod{p}$. Dados

2 textos encriptados, (c_1, c_2) e (c'_1, c'_2), definimos a multiplicação componente a componente, isto é, $(c_1, c_2).(c'_1, c'_2) = (c_1.c'_1, c_2.c'_2)$. Sendo assim, é fácil ver que o ElGamal é um homomorfismo, pois $(c_1.c'_1, c_2.c'_2) = (g^{k_1+k'_1}, g^{a(k_1+k'_1)}.m)$. De fato, este homomorfismo mapeia a operação de multiplicação na operação de soma.

O primeiro homomorfismo secreto com demonstração de segurança semântica foi proposto por ***Goldwasser-Micali*** [GM82], usando como base o problema de computar o resíduo quadrático de um elemento de $\mathbb{Z}_N$, onde $N = p.q$, com p e q primos grandes. Calcular o resíduo quadrático em $\mathbb{Z}_p$ ou $\mathbb{Z}_q$ é fácil. Portanto, a chave privada é dada por (p,q), a fatoração de N, enquanto que a chave pública é dada por (N,z), onde z é um elemento de $\mathbb{Z}_N$ tal que $z^{p-1/2} \equiv 1 \pmod N$ e z não é um resíduo quadrático em $\mathbb{Z}_N$. Dada uma mensagem $m \in \{0,1\}$, se $m = 0$, o algoritmo de encriptação retorna um resíduo quadrático aleatório em $\mathbb{Z}_N$, caso contrário, se $m = 1$, o algoritmo de encriptação retorna um resíduo não-quadrático c tal que $c^{p-1/2} \equiv 1 \pmod N$. A decriptação só pode ser realizada com o conhecimento da fatoração de N, de modo que os resíduos quadráticos possam ser calculados separadamente em $\mathbb{Z}_p$ e $\mathbb{Z}_q$, usando o teorema chinês do resto para calcular o resíduo quadrático em $\mathbb{Z}_N$. Em especial, dados dois resíduos quadráticos, sabemos que a sua multiplicação resulta em um resíduo quadrático. E também é fácil ver que a multiplicação de resíduos não-quadráticos c_1 e c_2, tais que $c_i^{p-1/2} \equiv 1 \pmod N$, resulta em um novo elemento $c = \mathbb{Z}_N$, tal que $c^{p-1/2} \equiv 1 \pmod N$. Logo, o esquema é homomórfico com relação a multiplicação e pode ser utilizado como homomorfismo secreto.

Particularmente importante é o que criptossistema ***Paillier*** , cuja segurança também é baseada (embora não haja demonstração de que seja equivalente) ao problema de fatoração de um número composto $N = p.q$, com p e q tendo a mesma quantidade de bits. Este esquema utiliza o grupo $\mathbb{Z}^*_{N^2}$. Dado que $N = p.q$, temos que $\mathbb{Z}^*_{N^2}$ é isomorfo a $\mathbb{Z}_N \times \mathbb{Z}^*_N$. De fato, o isomorfismo é dado pela relação $f : \mathbb{Z}_N \times \mathbb{Z}^*_N$, tal que:

$$f(a,b) = (1+N)^a.b^N \pmod{N^2}.$$

A chave pública é o próprio valor de N, enquanto que a chave privada é dada pelo par (p,q). Para criptografar uma mensagem $m \in \mathbb{Z}^*_N$, é computado o valor $c = (1+N)^m.r^N \pmod{N^2}$. Por sua vez, o algoritmo de decriptação computa

$$m = \frac{[c^{\phi(N)} \pmod{N^2}] - 1}{N}.\phi(N)^{-1} \pmod N.$$

A encriptação é homomórfica com relação a soma, já que

$$\begin{aligned} \mathrm{Enc}(N,m_1).\mathrm{Enc}(N,m_2) &= ((1+N)^{m_1} r_1^N).((1+N)^{m_2} r_2^N), \\ &= (1+N)^{m_1+m_2 \pmod N} (r_1 r_2)^N \pmod{N^2}. \end{aligned}$$

Em geral, a função f é tal que $f(a_1,b_1).f(a_2,b_2) = f(a_1+a_2, b_1.b_2)$.

Outro criptossistema que permite a construção de homomorfismo secreto é o ***Polly Cracker***, proposto por Fellow e Koblitz [FK94], onde um anel polinomial

$R = \mathbb{F}_q[x_1,...,x_n]$ contém um ideal I gerado por um conjunto de polinômios públicos, $\{p_1(x_1,...,x_n),...,p_k(x_1,...,x_n)\}$, com uma raíz $\alpha = (\alpha_1,...,\alpha_n)$ em comum, mantida em segredo. Dada uma mensagem $m \in \mathbb{F}_q$, o algoritmo de encriptação computa o polinômio $c(\mathbf{x}) = \sum p_i(\mathbf{x}).r_i(\mathbf{x})$, onde r_i são polinômios escolhidos aleatoriamente, para obteção de um elemento aleatório de I. Para decriptar, basta avaliar o polinômio $c(\mathbf{x})$ em α. A segurança do Polly Cracker é um problema em aberto, porque apesar dos ataques que surgiram, adaptações foram realizadas de modo a sanar as vulnerabilidades.

O criptossistema ***BGN*** [BGN05] é um esquema prático que permite avaliação fórmulas quadráticas, ou seja, permite circuitos com um nível de multiplicação e um número arbitrário de adições. Sejam $N = p.q$ e considere os grupos $\mathbb{G}$ e $\mathbb{G}_1$, de ordem N, e um emparelhamento bilinear $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$. Dado um gerador $g \in \mathbb{G}$, computa-se $h = g^p$ e a chave pública é dada por (N,h,g), enquanto que a chave privada é a fatoração (p,q) de N. O espaço de texto claro $\mathscr{P}$ é $\mathbb{Z}_p$ e o algoritmo de encriptação computa $c = g^{m+kp}$, com k aleatório e $m \in \mathbb{Z}_p$. O algoritmo de decriptação computa $c^q \equiv g^{mq} \pmod N$ e posteriormente resolve o problema do logaritmo discreto com base g^q. Para que o logaritmo discreto seja eficiente, m precisa corresponder a um elemento de um conjunto com tamanho polinomial, ao invés de ser um elemento qualquer de $\mathbb{Z}_p$. Pelo fato de ser usado o logaritmo discreto como problema difícil subjacente, temos que a multiplicação de textos encriptados corresponde a soma de textos claros. Além disso, dados os textos encriptados $c_1 = g^{m_1+k_1p}$ e $c_2 = g^{m_2+k_2p}$, o emparelhamento bilinear $e(c_1,c_2)$ é igual a $e(g,g)^{m_1.m_2+d.p}$, para um inteiro d.

A segurança de todas as propostas que discutimos anteriormente está relacionada a dificuldade do ***problema de participação em ideal*** (*ideal membership problem*).

Em 2009, Craig Gentry [Gen09b] utilizou reticulados gerados por polinômios ideais para construir o primeiro esquema de ECH, resolvendo assim um problema que ficou em aberto por 31 anos. Devido à complexidade de avaliação das multiplicações e ao tamanho da chave pública, tal proposta ainda não pode ser usada na prática. Porém, algumas otimizações foram propostas, [vDGHV09, SS10, SV09], fazendo-nos acreditar que o ECH está cada vez mais próximo ser se tornar realidade. Com isso, um novo tipo de segurança criptográfica poderá ser oferecido, especialmente no contexto de computação em nuvem. A nova proposta de Craig Gentry está relacionada a um problema ligeiramente diferente, denominado ***problema de classes laterais em ideais*** (*ideal coset problem*).

Em resumo, é possível descrever um modelo genérico do esquema de Craig Gentry como segue:

Geração de chaves. O algoritmo KeyGen escolhe ideais J primo com I e gera as bases B_J^{sk} e B_J^{pk}. Além disso, é determinado uma distribuição $\mathscr{D}_{B_I}(m)$ que gera elementos aleatórios da classe lateral $m+I$.

Encriptação. Dada uma mensagem $m \in R \pmod{B_I}$, utiliza-se a distribuição $\mathscr{D}_{B_I}(m)$ para computar m' e depois é realizada uma redução módulo B_J^{pk}, como segue:

$$c = m' = \mathscr{D}_{B_I}(m) \pmod{B_J^{\text{pk}}}.$$

Decriptação. Para decriptar é computado o valor

$$m = [c \pmod{B_{\text{sk}}}] \pmod{B_I}.$$

Graig Gentry utiliza ideais polinomiais para obter um esquema de ***encriptação homomórfica restrita*** (*somewhat homomorphic encryption*). Este esquema é capaz de somar e multiplicar textos encriptados de maneira homomórfica, mas conforme as operações são realizadas, é acrescentado um ruído ao texto encriptado. O algoritmo de decriptação funciona desde que tal ruído não ultrapasse um certo limiar. Usando um conceito que chamou de ***autoinicialização*** (*bootstrapping*), Craig Gentry propõe a construção de um novo esquema, que pode decriptar e reduzir o ruído homomorficamente. Porém, esta adaptação acarreta diretamente no aumento do tamanho dos parâmetros, tornando inviável a implementação do esquema na prática.

Também em 2009, Craig Gentry [Gen09a] propôs um esquema baseado em reticulados ideais, cuja ideia é utilizar um ruído r na encriptação, de modo que a decriptação só funcione caso este ruído seja menor que um determinado limiar $\mathscr{R}$, como ocorre, por exemplo, no criptossistema GGH?[GGH97]. É possível efetuar as operações de soma e multiplicação de textos encriptados, alterando o ruído r proporcionalmente a r e r^2, respectivamente, para cada operação. Com isso, é possível avaliar circuitos algébricos de profundidade multiplicativa máxima $\log_2(\mathscr{R})$. Para construir um esquema capaz de avaliar circuitos de profundidade arbritrária, Craig Gentry alterou esta ideia para que o algoritmo de decriptação pudesse ser expresso como um circuito de baixa profundidade multiplicativa, de modo que fosse possível reduzir o ruído por meio de uma operação que foi denominada ***reencriptação*** (*recryption*).

Para ter segurança equivalente a 2^λ, a performance do esquema proposto por Craig Gentry, após algumas otimizações, é capaz de computar cada operação de um determinado circuito em tempo quase linear em função de λ^6, enquanto a chave pública tem tamanho λ^7. Ao invés de utilizar reticulados, é possível aplicar as mesmas ideias sobre os números inteiros, conduzindo a um esquema com eficiência semelhante, porém com chave pública de tamanho da ordem de λ^{10}. Em trabalhos recentes, o tamanho da chave pública foi reduzido para λ^7 [CMNT11] e λ^5 [CNT11].

Implementações recentes mostram que a eficiência do esquema ainda é um ponto crítico, levando 2,2 horas para geração de chaves e 31 minutos para computar a reencriptação, no caso de reticulados ideais [GH11b] e, no caso de inteiros, levando 43 minutos para geração de chaves e 14 minutos e 33 segundos para computar a reencriptação. É importante salientar que esses dados não correspondem a uma segurança equiparável [CMNT11].

Utilizando criptografia parcialmente homomórfica, é possível realizar computações limitadas sobre os dados encriptados. Embora o esquema de Craig Gentry não seja eficiente para ser usado na prática, ele provê uma forma eficiente de computar **parcialmente** (circuitos com profundidade multiplicativa limitada) com os dados encriptados, permitindo a construção de aplicações de grande interesse [NLV11a]. Em especial, uma recente proposta [BGV11] permite a construção de um esquema capaz de avaliar circuitos algébricos de profundidade multiplicativa L em tempo $O(\lambda L^3)$.

4.2. Criptografia completamente homomórfica

Nas próximas seções serão descritos em detalhes as propostas de criptografia completamente homomórfica. Primeiramente, será apresentado o esquema simplificado, que utiliza apenas números inteiros e contém os principais conceitos que também serão utilizados no esquema baseado em reticulados ideais. Ambas as propostas seguem a mesma estratégia, que pode ser resumida de acordo com os seguintes passos:

1. obtenção de um esquema capaz de lidar com uma classe limitada de circuitos, isto é, um esquema de encriptação homomórfica restrita;
2. redução da profundidade do circuito de decriptação;
3. implementação da autoinicialização, permitindo construir um esquema completamente homomórfico em nível.

4.2.1. Segurança

A segurança de um criptossistema contra ***ataque adaptativo de texto encriptado escolhido*** (CCA2 - *chosen-ciphertext attack*) é definida levando em consideração o seguinte jogo:

Configuração. O desafiante obtém $(\mathrm{sk}, \mathrm{pk}) = \mathrm{KeyGen}(\lambda)$ e envia pk para o adversário $\mathscr{A}$.

Consultas. $\mathscr{A}$ envia textos encriptados para o desafiante, antes ou depois do desafio, que retorna o texto claro correspondente.

Desafio. O adversário gera aleatoriamente dois textos claros $m_0, m_1 \in \mathscr{P}$ e manda para o desafiante, que escolhe um bit $b \in \{0,1\}$ aleatoriamente e computa o texto encriptado $c = \mathrm{Enc}(\mathrm{pk}, m_b)$. O desafiante envia c para $\mathscr{A}$.

Resposta. $\mathscr{A}$ manda um bit b' para o desafiante e ganha o jogo se $b' = b$.

O esquema é seguro se não houver um adversário polinomial capaz de vencer este jogo com probabilidade não negligível.

Uma definição ligeiramente diferente, que permite consultas apenas antes de ser feito o desafio, é denotado pela sigla CCA1. Um criptossistema $\mathscr{E}$ é denominado seguro contra ***ataque adaptativo de texto claro escolhido*** se forem permitidas apenas consultas sobre texto claros e não sobre textos encriptados, portanto é um modelo de ataque menos restritivo. Se não for permitida nenhuma consulta, o sistema é denominado ***semanticamente seguro***. Este último modelo, é o mais restritivo, porque garante que o texto encriptado não contém informação a respeito de nenhuma função que possa ser computada eficientemente a partir do texto claro. A segurança com relação a este modelo implica diretamente na impossibilidade de obter um criptossistema que seja capaz de responder consultas de comparação de valores, como por exemplo é necessário para ordenar sequências de valores. Em modelos de computação que utilizam ***ramificação condicional***, isto é, verificam se um valor x é maior ou igual a zero, é possível representar um algoritmo por meio de um programa que contém instruções como laços, saltos condicionais, etc. Esta é uma representação bastante prática em comparação com circuitos algébricos puros (formados apenas por somas e multiplicações), mas infelizmente não é possível

obter segurança semântica neste contexto. A possibilidade de verificar se um texto encriptado corresponde a um texto claro cujo valor seja maior que zero, sem conhecimento da chave privada, ou seja, a existência um algoritmo eficiente para computar a função $f : \mathscr{K}_{\text{pub}} \times \mathscr{C} \to \{0,1\}$ tal que $f(\text{pk}, c) = 1$ se e somente se $c = \text{Enc}(\text{pk}, m)$ e $m \geq 0$, é justamente um exemplo de função que contém informação relevante do texto encriptado e fere a definição de segurança semântica.

Duas distribuições são ***indistinguíveis*** caso a adaptação trivial do jogo descrito acima não puder ser ganho por um adversário polinomial.

4.2.2. Homomorfismos secretos

Definição 4.2.1. ***Corretude.*** O esquema $\mathscr{E}(\text{KeyGen}, \text{Dec}, \text{Enc}, \text{Eval})$ é *correto* se, para um determinado circuito C e se para qualquer par de chaves (sk, pk) gerado por KeyGen quaisquer tuplas de mensagens $(m_1, ..., m_t)$ e seus respectivos textos encriptados $\Psi = \langle \psi_1, ..., \psi_t \rangle$, ou seja, $\psi_i = \text{Enc}(\text{pk}, m_i)$ para $1 \leq i \leq t$, então temos que

$$\text{Dec}(\text{sk}, \text{Eval}(\text{pk}, C, \Psi)) = C(m_1, ..., m_t).$$

Além disso, os algoritmos KeyGen, Dec, Enc e Eval devem ter complexidade polinomial.

Encriptação completamente homomórfica. O esquema $\mathscr{E}$ é *correto para uma classe* $\mathbf{S_C}$ de circuitos, se for correto para cada $\mathbf{C} \in \mathbf{S_C}$. Além disso, $\mathscr{E}$ é denomicado *completamente homomórfico* se for correto para todo circuito algébrico. Alternativamente, podemos basear a construção em circuitos booleanos, já que ambos os modelos de computação são equivalentes.

Privacidade do circuito. Dizemos que um esquema $\mathscr{E}$ tem *privacidade de circuito* se as seguintes funções forem indistinguíveis:

$$\text{Enc}(\text{pk}, C(m_1, ..., m_t)) \approx \text{Eval}(\text{pk}, C, \Psi).$$

Encriptação homomórfica compacta. O esquema $\mathscr{E}$ é *compacto* se para todo circuito C, todo conjunto de textos encriptados Ψ, a partir de qualquer chave pública válida, isto é, gerada por KeyGen, então o tamanho do texto encriptado gerado pelo algoritmo Eval é polinomial em relação ao parâmetro de segurança λ e independente do tamanho de C.

Circuito de decriptação aumentado. Seja $\mathscr{E}$ um esquema tal que a decriptação é implementado por um circuito que depende apenas do parâmetro de segurança λ. Define-se o ***conjunto de circuitos de decriptação aumentado*** como sendo o conjunto formado por dois circuitos que recebem como entrada a chave privada e dois textos encriptados. O primeiro circuito, $D^{(+)}_{\mathscr{E}}$, decriptar os textos encriptados de Ψ e soma os resultados, enquanto que o segundo, $D^{\times}_{\mathscr{E}}$, faz o mesmo e ao final multiplica os resultados.

Encriptação com autoinicialização. Seja $\mathscr{E}$ um esquema de encriptação homomórfica. Se $\mathbf{S_C}$ representa o conjunto dos circuitos para o qual $\mathscr{E}$ é correto, e se

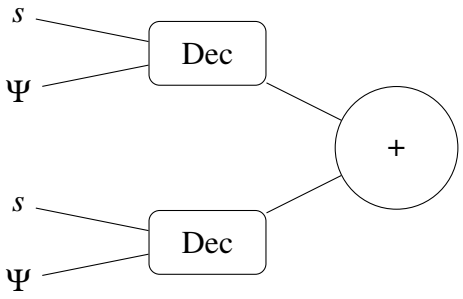

Figura 4.5. $D^{+}_{\mathscr{E}}$

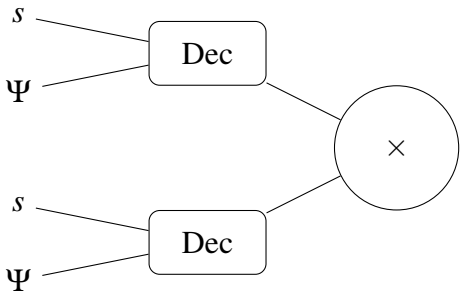

Figura 4.6. $D^{\times}_{\mathscr{E}}$

$D_{\mathscr{E}} \subseteq \mathbf{S_C}$, onde $D_{\mathscr{E}}$ representa o conjunto de circuitos de decriptação aumentado, então $\mathscr{E}$ é denominada ***autoinicializável***.

Encriptação homomórfica em nível. Seja $\mathscr{E}$ um esquema correto para os circuitos de decriptação aumentado, ou seja, $\mathscr{E}$ é ***autoinicializável***, então é possível construir um novo esquema $\mathscr{E}^{(d)}$, correto, compacto e homomórfico para todos os circuitos booleanos de profundidade d. Além disso, $\mathscr{E}^{(d)}$ é semanticamente seguro se $\mathscr{E}$ também é. Especificamente, um ataque de com vantagem ε sobre $\mathscr{E}$ pode ser transformado em um ataque com vantagem $\varepsilon/\ell d$, onde ℓ é o tamanho da chave privada em $\mathscr{E}$.

Este novo esquema utiliza o mesmo $D_{\mathscr{E}}$ e possui mesmo tamanho de chave privada e de texto encriptado. A chave pública consiste de $d+1$ chaves públicas de $\mathscr{E}$, $(\mathrm{pk}_1, ..., \mathrm{pk}_{d+1})$, acrescidas da encriptação de s_i usando pk_{i+1}.

Em cada nível i de um circuito C, os textos são novamente encriptados, utilizando pk_{i+1} e cada soma ou multiplicação do circuito original é substituída por um circuito de decriptação aumentado equivalente. Sendo assim, existe um algoritmo, denotado por Rec, que reencripta a mensagem trocando a chave pública pk_i por pk_{i+1}, de modo que a mensagem sempre está protegida por um nível de encriptação.

Algoritmo 4.2.1 Reencriptação

ENTRADA pk_{i+1}, $D_{\mathscr{E}}$, $\bar{s}_i = \mathrm{Enc}(\mathrm{pk}_{i+1}, s_i)$ e Ψ_i.
SAÍDA um conjunto de textos encriptados usando a chave pk_{i+1} e o conjunto Ψ_i.
$\bar{\Psi}_i = \mathrm{Enc}(\mathrm{pk}_{i+1}, \Psi_i)$.
retorne $\Psi_{i+1} = \mathrm{Eval}(\mathrm{pk}_{i+1}, D_{\mathscr{E}}, (\bar{s}_i, \bar{\Psi}_i))$.

A figura 4.7 mostra um exemplo com um circuito de dois níveis, utilizando quatro mensagens, m_1, m_2, m_3 e m_4, de modo que definimos as seguintes variáveis:

$$\Psi_1 = (\mathrm{Enc}(\mathrm{pk}_1, m_1), \mathrm{Enc}(\mathrm{pk}_1, m_2)),$$
$$\Psi_i^{\mathrm{add}} = \mathrm{Enc}(pk_i, m_1 + m_2),$$
$$\Omega_1 = (\mathrm{Enc}(\mathrm{pk}_1, m_3), \mathrm{Enc}(\mathrm{pk}_1, m_4)),$$
$$\Omega_i^{\mathrm{add}} = \mathrm{Enc}(pk_i, m_3 + m_4).$$

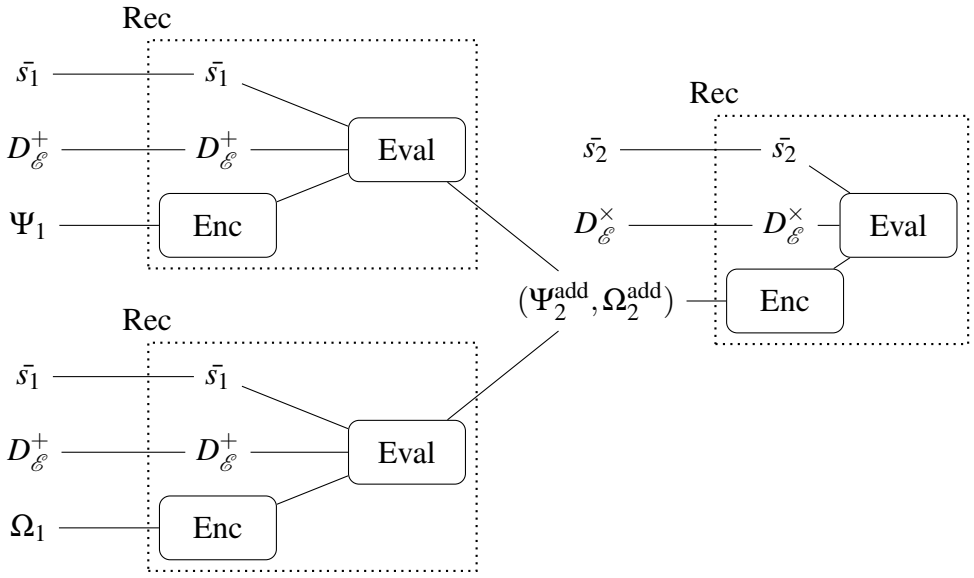

Figura 4.7. $\text{Enc}((m_1+m_2)\times(m_3+m_4))$

Para simplificar a notação, vamos utilizar $\bar{s_i}$ para denotar o vetor composto pela encriptação de cada bit de s_i usando a chave pública pk_{i+1}. Analogamente, denotamos por $\bar{\Psi}_i$ (ou $\bar{\Omega}_i$) a encriptação de Ψ_i (ou Ω_i) usando a chave pública pk_{i+1}.

Definição 4.2.2. ***Segurança circular.*** Dado um esquema $\mathscr{E}$, dizemos que $\mathscr{E}$ tem segurança circular se for seguro criptografar a chave privada com sua própria chave pública.

Se o esquema $\mathscr{E}$ tiver segurança circular, podemos utilizar a própria chave pública, pk, para encriptar a chave privada sk e portanto não é preciso uma cadeia de $d+1$ chaves públicas.

4.2.3. O esquema sobre os inteiros

Nesta seção será descrito um esquema simplificado, baseado em números inteiros, com a intenção de ilustrar o funcionamento da matemática em dimensão 1. A proposta original de Craig Gentry [Gen09a] estende as mesmas ideias para dimensão n. Será inicialmente descrita a versão simétrica e depois será introduzido o uso do problema SSP para ao mesmo tempo tornar o esquema assimétrico e também para otimizar o algoritmo de decriptação.

4.2.3.1. Versão simétrica

Definição 4.2.3. Seja λ o parâmetro de segurança. O algoritmo KeyGen gera aleatoriamente um inteiro ímpar p com λ^2 bits. Para encriptar um bit m, o algoritmo Enc escolhe m' com λ bits, de modo que m' tenha a mesma paridade de m. É utilizado um inteiro q, com λ^5 bits e o texto encriptado c é calculado da seguinte forma:

$$c = m' + pq.$$

O algoritmo de decriptação computa $m = \mathrm{Dec}(c,p) = \lfloor c \rceil_p \pmod 2$, obtendo de volta o bit encriptado. É simples ver que a encriptação é homomórfica com relação a soma e também com relação a multiplicação. Porém, a decriptação só funciona caso $|m'|$ seja menor que $p/2$, pois a redução módulo p terá a mesma paridade que m.

O problema de encontrar p dados os textos encriptados c_1, c_2, ..., c_k, tal que $c_i = m'_i + pq_i$ e $m'_i \ll p$, denominado ***máximo divisor comum (mdc) aproximado***, foi estudado no contexto de criptoanálise [HG01]. O tamanho de q_i é escolhido para resistir ao ataque descrito neste trabalho.

4.2.3.2. Versão assimétrica

Para tornar o esquema assimétrico a chave privada continua sendo p e a chave pública é formada por encriptações do zero, isto é, inteiros na forma $x_i = 2r_i + pq_i$, para valores de r_i e q_i escolhidos nos mesmos intervalos, de modo que exista um subconjunto cuja soma seja igual a $1/p$. Dada uma mensagem m, o algoritmo de encriptação soma m com um subconjunto aleatório da chave pública. A segurança do esquema passa a depender da dificuldade do problema SSP (*subset sum problem*).

Assim, tendo a solução do problema SSP é possível computar a chave privada, enquanto a chave pública é um conjunto de inteiros que é a entrada do problema SSP. Ou seja, a chave pública é dada por $(s_1, ..., s_k)$ e existe um subconjunto S dos índices tal que $1/p = \sum_{i \in S} s_i$. O algoritmo de encriptação retorna o vetor $(cs_1, ..., cs_k)$ e, para decriptar, calcula-se a soma $\sum_{i \in S} cs_i \pmod 2 = c/p \pmod 2$.

É importante ressaltar que esta ideia possui uma vantagem em relação à decriptação, porque a nova proposta é mais eficiente, já que o cálculo de c/p pode ser efetuado facilmente usando a solução do problema SSP.

Parâmetros. A construção a seguir utiliza diversos parâmetros, cujos tamanhos são polinomiais em relação ao parâmetro de segurança λ:

- γ é o comprimento em bits dos valores de x_i. Este parâmetro deve ser escolhido de maneira tal que $\gamma = \omega(\eta^2 \log \lambda)$, pois assim evita ataques contra o problema do mdc aproximado;

- η é o comprimento em bits da chave secreta p, respeitando a desigualdade $\eta \geq \rho\Theta(\lambda \log^2 \lambda)$, para permitir que o esquema seja capaz de avaliar homomorficamente o circuito reduzido de decriptação;
- ρ é o comprimento em bits do ruído r_i. Este parâmetro deve ser escolhido de forma que $\rho = \omega(log\lambda)$, para que o esquema resista a ataque de força bruta contra o ruído;
- τ é a quantidade de x_i's na chave pública, sendo escolhido de modo que $\tau \geq \gamma + \omega(\log \lambda)$, para que seja possível utilizar o *leftover hash lemma* na redução ao problema do mdc aproximado;
- $\rho' = \rho + \omega(\log \lambda)$ é um parâmetro secundário utilizado no algoritmo de decriptação.

Uma sugestão dada instancia os parâmetros da seguinte forma: $\rho = \lambda$, $\rho' = 2\lambda$, $\eta = \tilde{O}(\lambda^2)$, $\gamma = \tilde{O}(\lambda^5)$ e $\tau = \gamma + \lambda$ [vDGHV09]. Considerando uma escolha de parâmetros como esta, definimos a seguinte distribuição:

$$\mathscr{D}_{\gamma,\rho}(p) = \{x = pq + r \mid q \leftarrow \mathbb{Z} \cap [0, 2^\gamma/p), r \leftarrow \mathbb{Z} \cap (-2^\rho, 2^\rho)\}.$$

Definição 4.2.4. Geração de chaves. Obtenha um inteiro ímpar p aleatório com η bits. Para $0 \leq i \leq \tau$, compute $x_i = \mathscr{D}_{\gamma,\rho}(p)$. Renomeie os índices de modo que x_0 corresponda ao maior elemento. Faça isso até que x_0 seja ímpar e $x_0 \pmod p$ seja par. A chave pública é dada por $\mathscr{K}_{\text{pub}} = (x_0, ..., x_\tau)$ e a chave privada é dada por $\mathscr{K}_{\text{priv}} = p$.

Encriptação. Escolha um subconjunto aleatório $S \subset \{0, 1, ..., \tau\}$ e um inteiro r aleatório no intervalo $(-2^{\rho'}, 2^{\rho'})$ e compute $c = [m + 2r + \sum_{i \in S} x_i]_{x_0}$.

Decriptação. Retorne $m = [c]_p \pmod 2$.

Avaliação. Dado o circuito C e t textos encriptados, execute as operações de C aos textos encriptados, portanto sobre inteiros grandes, e retorne o valor encontrado.

É importante ressaltar que na medida em que p é ímpar, a decriptação pode ser efetuada da seguinte maneira:

$$m' = [c - \lfloor c/p \rceil]_2 = (c \pmod 2) \oplus (\lfloor c/p \rceil \pmod 2).$$

4.2.3.3. Corretude

Vamos agora demonstrar a corretude do esquema da definição 4.2.4. Contudo, antes disso, serão dadas as definições de circuito generalizado e circuito permitido, que até certo ponto são definições que tornam a demonstração da corretude uma mera aplicação das definições que seguem.

Definição 4.2.5. ***Circuito generalizado.*** Considerando um circuito c cujas portas correspondam a operações de soma e multiplicação módulo 2, o circuito generalizado $g(c)$ é formado por operações equivalentes, porém ao invés de efetuar cálculos sobre bits, computa a soma e multiplicação de números inteiros.

Definição 4.2.6. ***Circuitos permitidos.*** Considerando um circuito c e o circuito generalizado $g(c)$ correspondente, define-se a *classe de circuitos permitidos*, denotada por $C_{\mathscr{E}}$, como sendo aqueles circuitos cujas entradas sejam inteiros valor absoluto no máximo $2^{\alpha(\rho'+2)}$ e cuja saída tenha valor absoluto no máximo $2^{\alpha(\eta-4)}$.

Na verdade, a definição 4.2.6 é uma abordagem que permite demonstrar a corretude do esquema de forma direta, pois os parâmetros foram escolhidos justamente para satisfazê-la. De fato, o ruído máximo de um texto encriptado novo é $2^{\rho'+2}$, enquanto que o algoritmo de decriptação espera um texto encriptado com ruído cujo valor absoluto seja limitado por $p/2$, isto é, um valor estritamente menor que $2^{\gamma-2}$. Para permitir a redução do circuito de decriptação, este limite é reduzido para $p/8$, correspondendo portanto ao valor anunciado $2^{\eta-4}$.

Além disso, a partir desta definição não fica claro que tipo de circuito pode ser avaliado. Em especial, o ruído de uma multiplicação é elevado ao quadrado, enquanto que na soma o ruído tem um crescimento linear. Desta forma, interpretando o circuito como um polinômio multivariável, o grau deste polinômio representa a ***profundidade multiplicativa*** do circuito.

Lema 4.2.1. Dado um circuito c e o circuito generalizado $g(c)$ correspondente, construímos o polinômio $f(x_1,...,x_t)$ equivalente a este circuito. Seja d um inteiro correspondente ao grau de f, então se $|f|(2^{\rho'+2})^d \leq 2^{\eta-4}$, onde $|f|$ representa a somatória dos coeficientes de f, temos que c é um circuito permitido, ou seja, $c \in \mathbf{S_C}$.

Prova. De acordo com 4.2.3.2, temos que $c = [m + 2r + \sum_{i\in S} x_i]_{x_0}$. Como x_0 é o valor máximo entre todos os valores de x_i, para $0 \leq i \leq \tau$, então existe um inteiro k, tal que $|k| < \tau$, satisfazendo a seguinte equação

$$c = (m + 2r + \sum_{i\in S} x_i) + kx_0.$$

Pela definição de x_i, temos que $x_i = q_i p + 2r_i$, para $|r_i| \leq 2^{\rho}$. Com isso, temos que

$$c = k(q_0 p + 2r_0) + (m + 2r + \sum_{i\in S}(q_i p + 2r_i)),$$

$$c = p(kq_0 + \sum q_i) + (m + 2r + 2kr_0 + \sum_{i\in S} 2r_i),$$

$$c = p(kq_0 + \sum q_i) + (m + 2(r + kr_0 + \sum_{i \in S} r_i)).$$

Considerando que $\rho' \geq 2\rho$ e $\tau \leq 2^\rho$, já que $\tau = \lambda^5 + \lambda \leq 2^\lambda$ ($\lambda > 23$ é suficiente para garantir essa condição), o termo mais a direita tem valor absoluto no máximo,

$$\begin{aligned} |1 + 2(2^{\rho'+1} + \tau 2^{\rho+1} + \tau 2^{\rho+1})| &\leq |1 + 2(2^{\rho'+1} + \tau 2^{\rho+2})|, \\ &\leq |1 + 2^{\rho'+2} + \tau 2^{\rho+3}|, \\ &\leq |2^{\rho'+3}|. \end{aligned}$$

Portanto, o esquema pode avaliar polinômios cujo grau d respeite a seguinte desigualdade:

$$d \leq \frac{\eta - 4 - \log|f|}{\rho' + 2}.\square$$

Polinômios que satisfaçam essa condição são denominados ***polinômios permitidos.***

Lema 4.2.2. Seja $(\mathrm{sk}, \mathrm{pk})$ um par de chaves gerado por KeyGen. Seja $c = \mathrm{Enc}(\mathrm{pk}, m)$, com $m \in \{0,1\}$. Então, temos que $c \pmod p$ é da forma $2a + m$, ou seja, $c \pmod p$ possui a mesma paridade que m. Além disso, $|2a + m| < 2^{\rho'+2}$.

Prova. De acordo com 4.2.3.2, temos que $c = [m + 2r + \sum_{i \in S} x_i]_{x_0}$. Como x_0 é o valor máximo entre todos os valores de x_i, para $0 \leq i \leq \tau$, então a redução modular de cada x_i por x_0 resulta em um inteiro negativo. Desconsiderando esta parte negativa, sabemos que $2r$ por definição é no máximo $2^{\rho'+2}$. $\square$

Lema 4.2.3. Considerando um circuito permitido $\mathbf{C}$, o resultado de $\mathrm{Eval}(\mathrm{pk}, \mathbf{C}, c_1, ..., c_t)$, onde c_i são textos encriptados válidos, é um texto encriptado cujo rúido é no máximo $p/8$.

Prova. O ruído de $\mathrm{Eval}(\mathrm{pk}, \mathbf{C}, c_1, ..., c_t)$ é dado pela avaliação de $\mathbf{C}$ nos ruídos de c_i, isto é, podemos separar a avaliação do circuito C em duas partes, sendo que a parte múltipla de p resulta em um novo múltiplo de p, enquanto que a avaliação dos ruídos separadamente, resulta no ruído final. Como o ruído de cada c_i é limitado por $2^{\rho'+2}$ de acordo com o lema 4.2.2, então pela definição de circuito permitido, temos que o ruído final é no máximo $2^{\eta-4} = p/8$. $\square$

Infelizmente o esquema $\mathscr{E}$ não possui decriptação com profundidade multiplicativa suficientemente curta para ser completamente homomórfica. Alguns ajustes serão feitos adiante e serão responsáveis pela construção de um novo esquema com circuito de decriptação adequado.

4.2.3.4. Segurança

Os detalhes da demonstração de segurança do esquema da seção anterior podem ser encontrados no trabalho original [vDGHV09], onde mostra-se que a existência de um ataque ao esquema proposto permite resolver o problema do mdc aproximado. Em linhas gerais, o problema é encontrar um divisor comum dentre um conjunto de múltiplos aproximados desse divisor. Supondo a existência de um algoritmo que seja capaz de descobrir um bit do texto claro, é utilizado o algoritmo do mdc binário para construir uma solução para o problema do mdc aproximado.

Neste momento vale apontar o motivo pelo qual foi utilizado o parâmetro secundário ρ'. Basicamente, escolhendo um ruído com $\rho' = 2\rho$ bits, temos que o texto encriptado está protegido por um *ruído alto* ρ', enquanto que a chave pública contém encriptações do zero, realizado com *ruído baixo* ρ. Esta diferença é um ponto chave na redução do criptossistema baseado em ruído alto para o problema do mdc aproximado de ruído baixo.

Definição 4.2.7. O ***problema do mdc aproximado***, parametrizado por (ρ, η, γ), consiste em: dados um número polinomial de elementos da distribuição $\mathscr{D}_{\gamma,\rho}(p)$, para um inteiro ímpar p escolhido aleatoriamente, revele p.

Teorema 4.2.1. Para a escolha de parâmetros realizada na definição 4.2.3.2 e o parâmetro de segurança λ, qualquer ataque $\mathscr{A}$ com vantagem ε sobre o esquema $\mathscr{E}$ pode ser convertido em um algoritmo $\mathscr{B}$ para resolver o problema do mdc aproximado com vantagem pelo menos $\varepsilon/2$. A complexidade de $\mathscr{B}$ é polinomial no tempo de execução de $\mathscr{A}$ e também sobre λ e $1/\varepsilon$.

4.2.3.5. Redução da profundidade do circuito de decriptação

Nesta seção serão apresentadas as ideias utilizadas para reduzir a profundidade do circuito de decriptação. Para construir um esquema completamente homomórfico é preciso que o algoritmo de decriptação possa ser computado por um circuito de profundidade multiplicativa suficientemente baixa. A decriptação é calculada pela expressão $m = [c - [c/p]]_2$, que não parece possuir um circuito com as características desejadas. Para resolver este problema serão acrescentadas ao texto encriptado, informações que ajudam a decriptar a mensagem sem comprometer o esquema. A seguir é apresentado um esquema capaz de avaliar seu próprio circuito de decriptação.

Parâmetros. Essa construção utiliza três novos parâmetros: $\kappa = \gamma\eta/\rho'$, $\theta = \lambda$ e $\Theta = \omega(\kappa \log \lambda)$, ou seja, todos possuem tamanho polinomial no parâmetro de segurança λ.

Geração de chaves. Compute sk e pk como na definição 4.2.3.2. Compute $x_p = \lfloor 2^\kappa/p \rceil$, escolha aleatoriamente um vetor, $s = \langle s_1, ..., s_\Theta \rangle$, com Θ bits e ***peso de Hamming*** θ. O conjunto S é definido como

$$S = \{i \mid s_i = 1\}.$$

Escolha aleatoriamente inteiros u_i, onde $1 \leq i \leq \Theta$, com no máximo κ bits, tais que $\sum_{i \in S} u_i = x_p \pmod{2^{\kappa+1}}$. Compute $y_i = u_i/2^{\kappa}$, de forma que cada y_i é um inteiro positivo menor ou igual a dois, com κ bits de precisão após a vírgula. Assim, temos que $[\sum_{i \in S} y_i]_2 = (1/p) - \Delta_p$, para $\Delta_p < 2^{-\kappa}$.

A chave privada é dada pelo vetor $(s_1, ..., s_\kappa)$ e a chave pública é dada por pk e o vetor $(y_1, ..., y_\Theta)$.

Encriptação. Compute c como no esquema inicial. Para $1 \leq i \leq \Theta$, calcule $z_i = [cy_i]_2$, mantendo apenas $\lceil \log \theta \rceil + 3$ de precisão para cada z_i. Retorne c e o vetor $(z_1, ..., z_\Theta)$.

Decriptação. Retorne $m' = [c - \lfloor \sum_{i \in S} s_i z_i \rceil]_2$.

Avaliação. A soma e multiplicação continuam sendo efetuadas por meio das operações canônicas de números racionais.

Lema 4.2.4. O esquema modificado é correto para circuitos permitidos $C \in \mathbf{S}_{\mathscr{C}}$. Além disso, dado um texto encriptado $(z_1, ..., z_\Theta)$, gerado pela avaliação de um circuito permitido qualquer, temos que $s_i z_i - \lfloor \sum s_i z_i \rceil \leq 1/4$.

Prova. Dado que a chave pública contém o vetor $(y_1, ..., y_\Theta)$, sabe-se que os valores de y_i foram escolhidos de forma que $[\sum s_i y_i]_2 = 1/p + \Delta_p$, onde $\Delta_p \leq 2^{-\kappa}$.

Dado um circuito permitido C, tal que $c^* = \text{Eval}(\text{pk}, C, c_1, ..., c_t)$, para textos encriptados c_i válidos, temos que $[c^* y_i]_2 = z_i - \Delta_i$, com $\Delta_i \leq 1/16\theta$, já que apenas $\lceil \log \theta \rceil + 3$ da precisão é mantida em relação a z_i. Com isso, temos que

$$\begin{aligned}
[(c^*/p) - \sum s_i z_i]_2 &= [(c^*/p) - \sum s_i [c^* y_i]_2 + \sum s_i \Delta_i]_2, \\
&= [(c^*/p) - c^* [\sum s_i y_i]_2 + \sum s_i \Delta_i]_2], \\
&= [(c^*/p) - c^* (1/p - \Delta_p) + \sum s_i \Delta_i]_2, \\
&= [(c^* \Delta_p + \sum s_i \Delta_i]_2.
\end{aligned}$$

Considerando este último termo, temos que $|c^* \Delta_p| \leq 1/16$, pois c^* é um texto encriptado retornado pelo algoritmo de avaliação, cuja entrada é formada por textos encriptados de tamanho no máximo $2^{\alpha(\rho'+2)}$. Assim, o algoritmo Eval retorna um valor com tamanho no máximo $2^{\alpha(\eta-4)}$. Em particular, os textos encriptados são limitados superiormente por 2^{γ}, de modo que c^* tem magnitude no máximo $2^{\gamma}(\eta-4)/(\rho'+2) < 2^{\kappa-4}$. Logo, como $\Delta_p < 2^{-\kappa}$, temos que $|c^* \Delta_p| < 1/16$. Já em relação à $|\sum s_i \Delta_i|$, como $|\Delta_i| < 1/16\theta$ e existem θ valores de i para os quais $i \in S$, então temos que $|\sum s_i \Delta_i| < 1/16$. Portanto, temos que

$$|[c^* \Delta_p + \sum s_i \Delta_i]_2| < 1/8. \square$$

4.2.3.6. Autoinicialização

Na seção anterior, além de obter um sistema assimétrico, a ideia fez com que a decriptação seja mais eficiente, permitindo avaliar homomorficamente o próprio circuito de decriptação. Se for possível decriptar o sistema desta forma, reduzindo o ruído, e ainda for possível realizar uma operação extra, de soma ou multiplicação, então conseguiríamos um novo esquema que é capaz de avaliar circuitos de qualquer tamanho.

Até o momento, o esquema descrito permite avaliar circuitos de tamanho limitado, portanto não é completamente homomórfico. Para resolver este problema, Craig Gentry utilizou a ideia que chamou de ***autoinicialização***, construindo uma função que permite reencriptar um texto encriptado de modo a reduzir o ruído. Para fazer isso, é inserida uma dica da chave privada no texto encriptado, com base no ***problema SSP***. Assim, usando um novo par de chaves é possível calcular uma nova encriptação, seguida de uma decriptação com a chave privada original. Com isso, o sistema continua protegido por um nível de encriptação, mas o ruído foi reduzido.

Teorema 4.2.2. Considerando o criptossistema da seção anterior e $D_{\mathscr{E}}$ o conjunto de circuitos de decriptação aumentado, então $D_{\mathscr{E}} \in \mathbf{S_C}$.

Prova. O objetivo é encontrar um circuito adequado para computar a seguinte equação:

$$m = c - \lfloor \sum s_i z_i \rceil \pmod 2.$$

Para auxílio será utilizada uma nova variável $a_i = s_i.z_i$, onde $1 \leq i \leq \Theta$. Com isso, $a_i = z_i$ quando $s_i = 1$ e $a_i = 0$ quando $s_i = 0$. Por definição, a_i possui $n = \lceil \log \theta \rceil + 3$ bits de precisão e existem θ valores de a_i diferentes de zero. Este último fato é crucial para encontrar um circuito adequado, porque permite reduzir a quantidade de variáveis que precisamos lidar. Esta redução é realizada encontrando $n+1 = \lceil \log \theta \rceil + 4$ números racionais w_j, tais que $\sum w_j = \sum a_i \pmod 2$.

Cada a_i é um número racional entre zero e dois. Portanto, a representação binária de a_i pode ser expressa da seguinte maneira:

$$a_i = a_{i,0}, a_{i,-1} a_{i,-2} \ldots a_{i,-n}.$$

Índices negativos são utilizados para reforçar o fato de que estes bits representam a expansão binária do número racional, ou seja, $a_i = 2^{-j} \sum_{j=0}^{n} a_{i,-j}$.

Antes de calcular w_j, vamos definir W_{-j} como sendo o ***peso de Hamming*** do vetor $\{a_{i,j}\}_{i=1}^{\theta}$, como mostra a tabela 4.1 a seguir:

Como não há mais que θ valores de a_i não nulos, então o valor de W_{-j} é no máximo θ e definindo $w_j = 2^{-j} W_{-j} \pmod 2$, temos que w_j pode ser representado por $\lceil \log \theta \rceil + 1$ bits de precisão.

$a_{1,0},$	$a_{1,-1}$	$a_{1,-2}$	...	$a_{1,-n}$
$a_{2,0},$	$a_{2,-1}$	$a_{2,-2}$	...	$a_{2,-n}$
$a_{3,0},$	$a_{3,-1}$	$a_{3,-2}$	...	$a_{3,-n}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
$a_{\theta,0},$	$a_{\theta,-1}$	$a_{\theta,-2}$	...	$a_{\theta,-n}$
W_0	W_{-1}	W_{-2}	...	W_{-n}

Tabela 4.1.

Lema 4.2.5. Considerando a sequência de bits $\overrightarrow{b} = (b_1, ..., b_k)$, o ***peso de Hamming*** de $\overrightarrow{b}$, denotado por $H_{\overrightarrow{b}}$ pode ser computado calculando cada um de seus bits. Se a representação binária de $H_{\overrightarrow{b}}$ for dada por $(h_n, ..., h_0)$, de modo que $H_{\overrightarrow{b}} = \sum 2^i h_i$, então h_i pode ser expresso por um polinômio de grau 2^i nas variáveis $\{b_i\}_1^k$. Além disso, existe um circuito de tamanho $k2^i$ que computa todos os valores de h_i simultaneamente.

Prova. O i-ésimo bit de $H_{\overrightarrow{b}}$ pode ser computado por δ_{2^i}, onde δ_i representa o i-ésimo ***polinômio simétrico elementar***. O grau de δ_{2^i} é exatamente 2^i e para calcular simultaneamente todos os valores de h_i basta computar o polinômio $p(z) = \prod(z - b_i)$, já que h_i corresponde ao coeficiente do termo z^{k-i} em $p(z)$.

O algoritmo a seguir computa os bits $h_0, ..., h_n$ e pode ser facilmente transformado em um circuito:

Algoritmo 4.2.2 Polinômios simétricos elementares

ENTRADA $b_1, ..., b_k$.
SAÍDA $\delta_1(b_1, ..., b_k), ..., \delta_n(b_1, ..., b_k)$.
  Inicialize $e_{0,0} = 1$ e $e_{i,0} = 0$ para $i = 1, 2, 3, ..., 2^n$.
  para $j = 1, 2, ..., k$ **faça**
    para $i = 2^\ell, 2^{\ell-1}, ..., 1$ **faça**
      Compute $e_{i,j} b_j e_{i-1,j-1} + e_{i,j-1}$ (aritmética polinomial).
  retorne $e_{1,k}, ..., e_{2^n,k}$.

As multiplicações de polinômios são realizadas com o auxílio da ***transformada rápida de Fourier (FFT)***. □

4.2.3.7. Segurança do novo esquema

Para que a nova proposta seja segura é preciso garantir que as informações inclusas na chave pública, ou seja, $(y_1, ..., y_\theta)$, não possam ser usadas para reconstruir a chave privada. Este problema foi considerado por Craig Gentry em 2009 [Gen09a] e é conhecido como ***problema da soma em subconjunto esparso*** (SSSP - *sparse subset sum problem*). Para que este problema seja difícil é preciso escolher θ suficientemente grande para evitar ataques de força bruta. Além disso, é necessário ter Θ maior que $\omega(\kappa \log \lambda)$, onde κ é o comprimento em bits dos números incluídos na chave pública.

4.3. O esquema sobre reticulados ideais

4.3.1. Introdução

Na seção anterior foi descrito um esquema baseado em números inteiros, com a intenção de simplificar os conceitos que serão necessários para descrever o esquema sobre reticulados. Existe portanto uma correspondência direta entre as ideias apresentadas nesta seção com a seção anterior, porque o caso sobre números inteiros é um caso particular da construção que será descrita nesta seção. Assim, o leitor pode esperar um grau de abstração maior, embora seja seguida a mesma cadeia de definições e teoremas até atingir o resultado desejado: um esquema capaz de avaliar o seu próprio circuito de decriptação acrescido de uma soma ou multiplicação. Com isso, será possível construir novamente um esquema que possua ***encriptação homomórfica em nível***.

Mas tendo em vista as dificuldades encontradas já na versão baseada em números inteiros, vamos utilizar definições mais abstratas para construir um esquema inicial, para depois concretizar a construção utilizando reticulados ideais, assim como foi feito por Gentry [Gen09b].

As definições da seção 4.2 serão utilizadas como base da construção aqui apresentada, portanto é pré-requisito para o entendimento do esquema que será descrito. Em particular, a estratégia de encontrar um esquema inicial, capaz de avaliar uma classe limitada de circuitos, para depois reduzir a profundidade do ***circuito de decriptação*** e, finalmente, utilizar a ***autoinicialização*** para tornar o esquema completamente homomórfico, será novamente o eixo principal que será seguido.

4.3.2. Resumo

As mesmas ideias que funcionam sobre os números inteiros podem ser usadas com anéis polinomiais ideais, onde os ideais I e J são utilizados com a mesma função dos inteiros (2) e (q). Isto é, um vetor m é encriptado computando $c = m + i + j$ e para decriptar, calcula-se

$$m = (c \pmod{B_I}) \pmod{B_J}.$$

Na prática, temos que $B_I = (2)$ e $B_J = (a(x))$, onde $a(x)$ tem grau suficientemente grande para que o espaço de busca por força bruta tenha tamanho λ e além disso, seja possível efetuar pelo menos uma operação de multiplicação de forma homomórfica.

Observando por um outro ângulo, a decriptação está relacionada ao problema do vetor mais próximo em reticulados, já que $\lfloor c \rceil_{B_J}$ é uma instância do ***problema CVP***. Assim, para que o esquema seja seguro, B_J deve ser suficientemente não ortonormal, para que não seja possível usar o algoritmo de Babai para decriptar a mensagem.

Portanto, o esquema de Gentry é semelhante ao criptossistema GGH, já que utiliza uma base boa B_J^{sk}, gerada por um polinômio com coeficientes pequenos. Já a chave pública B_J^{pk} é calculada usando a forma normal de Hermite desta primeira base. Com isso, a demonstração de segurança do esquema é baseada na complexidade dos problemas difíceis em reticulados.

4.3.3. Esquema Abstrato

O esquema abstrato será descrito em termos de anéis e ideais. Sendo assim, considere um anel genérico R gerado de acordo com o parâmetro de segurança λ. Seja $I \subset R$ um ideal e B_I uma base de I.

Dados R e B_I, o algoritmo IdealGen(R, B_I) retorna as bases pública e privada B_J^{pk} e B_J^{sk}, respectivamente, onde J é o ideal gerado independentemente por B_J^{pk} ou B_J^{sk}, tal que $I + J = R$, isto é, J é relativamente primo a I.

Dado um anel R, uma base $B_I \subset R$ e um elemento $r \in R$, então a notação $r \pmod{B_I}$ é utilizada para descrever o elemento representativo único $r^* \in R$ tal que $r * -r \in I$, isto é, dada a classe lateral $r + I$, existe um único elemento que a representa e este elemento pode ser diferente de acordo com a base escolhida. A notação $R \pmod{B_I}$ remete ao conjunto de elementos representativos distintos com respeito à base B_I.

Dado um anel R, as bases B_I e B_J dos ideais I e J, e um elemento $r \in R$, definimos o algoritmo $\mathscr{D}_{B_I,B_J}(r)$ simplesmente como uma forma de extrair aleatoriamente um elemento da classe lateral $r + I$.

Analogamente à definição 4.2.5, dado um circuito C composto de operações módulo B_I (espaço de texto claro), define-se o *circuito generalizado* como sendo o circuito construído a partir de C trocando-se as operações módulo B_I por operações equivalentes no anel R.

Seja X_{enc} a imagem de $\mathscr{D}_{B_I,B_J}$, de modo que todo texto encriptado é da forma $X_{\text{enc}} + J$. Além disso, seja $X_{\text{dec}} = R \pmod{B_J^{\text{sk}}}$, isto é, os elementos representativos das classes laterais de J com relação a base B_J^{sk}, de maneira que o algoritmo Dec só é capaz de decriptar textos encriptados pertencentes a X_{dec}. Assim como foi definido em 4.2.6, definimos o conjunto de *circuitos permitidos* como sendo

$$\mathscr{C}^*_{\mathscr{E}} = \{C \mid \forall (x_1, ..., x_t) \in X_{\text{enc}}^t, g(C)(x_1, ..., x_t) \in X_{\text{dec}}\}.$$

Mas diferentemente da definição 4.2.6, onde foram estabelecidos valores concretos para X_{enc} e X_{dec}, será utilizada esta versão mais abstrata da definição, permitindo demonstrar de forma simples a corretude do esquema abstrato.

A seguir definimos o esquema abstrato $\mathscr{E}(\text{KeyGen}, \text{Enc}, \text{Dec}, \text{Eval})$, onde o espaço de texto claro $\mathscr{P}$ é dado por $R \pmod{B_I}$ e o algoritmo Eval recebe como parâmetro um circuito cujas portas correspondem a operações realizadas módulo B_I.

Geração de chaves. O algoritmo KeyGen recebe como parâmetros o anel R e a base B_I e executa o algoritmo IdealGen(R, B_I) para obter as bases B_J^{pk} e B_J^{sk}. A chave pública corresponde a $\text{pk} = \{R, B_I, B_J^{\text{pk}}, \mathscr{D}_{B_I,B_J}\}$, enquanto a chave privada é dada por $\text{sk} = \{B_J^{\text{sk}}\}$.

Encriptação. Dada a chave pública pk e um texto claro $m \in \mathscr{P}$, o algoritmo Enc(pk, m) retorna $c = m + \mathscr{D}_{B_I,B_J} \pmod{B_J^{\text{pk}}}$.

Decriptação. Dada a chave privada B_J^{sk} e o texto encriptado c, o algoritmo Dec(B_J^{sk}, c) retorna $m = (c \pmod{B_J^{\text{sk}}}) \pmod{B_I}$.

Avaliação. Dada a chave pública B_J^{pk}, um circuito permitido $C \in \mathscr{C}_{\mathscr{E}}$ e um conjunto de textos encriptados $\Psi = (\psi_1, ..., \psi_t)$, o algoritmo $\text{Eval}(B_J^{\text{pk}}, C, \Psi)$ executa as operações Add e Mult do circuito generalizado $g(C)$ e retorne o texto encriptado $\psi = g(C)(\Psi)$.

Teorema 4.3.1. O esquema abstrato $\mathscr{E}$ é correto para circuitos permitidos $C \in \mathbf{S_C}$.

Prova. Para qualquer $\Psi = (\psi_1, ..., \psi_t)$, tal que $\psi_= m_k + i_k + j_k$, onde $i_k \in I$ e $j_k \in J$, ou seja, um conjunto de textos encriptados válidos, temos que

$$\begin{aligned} \text{Eval}(\text{pk}, C, \Psi) &= g(C)(\Psi) \pmod{B_J^{\text{pk}}}, \\ &\in g(C)(m_1 + i_1, ..., m_t + i_t) + J. \end{aligned}$$

Como $m_k + i_k \in X_{\text{enc}}$, temos que $g(C)(m_1 + i_1, ..., m_t + i_t) \in X_{\text{dec}}$, por definição. Logo,

$$\begin{aligned} \text{Dec}(\text{sk}, \text{Eval}(\text{pk}, C, \Psi) &= g(C)(m_1 + i_1, ..., m_t + i_t) + J, \\ &= g(C)(m_1 + i_1, ..., m_t + i_t) + \pmod{B_I}, \\ &= g(C)(m_1, ..., m_t) + \pmod{B_I}, \\ &= C(m_1, ..., m_t). \square \end{aligned}$$

4.3.4. Segurança do esquema abstrato

Definição 4.3.1. Problema da classe lateral em ideais (*ideal coset problem - ICP*). Dado um anel R, uma base B_I e os algoritmos IdealGen e $\mathscr{D}_{B_I,B_J}$, que retorna um elemento aleatório de R, o desafiante escolhe aleatoriamente um bit $b \in \{0,1\}$, gera $(B_J^{\text{sk}}, B_J^{\text{pk}}) = \text{IdealGen}(R, B_I)$. Se $b = 0$, ele computa $r = \mathscr{D}_{B_I,B_J}$ e $t = r \pmod{B}_J^{\text{pk}}$. Se $b = 1$, ele escolhe t uniformemente em $R \pmod{B}_J^{\text{pk}}$. O problema consiste em encontrar b dados (t, B_J^{pk}).

Resumidamente, o problema é distinguir entre uma distribuição uniforme e uma distribuição especial, induzida por $\mathscr{D}_{B_I,B_J}$.

Teorema 4.3.2. Suponha a existência de um algoritmo $\mathscr{A}$ capaz de atacar o esquema abstrato $\mathscr{E}$ com vantagem ε. Então existe um algoritmo $\mathscr{B}$, com tempo de execução polinomial em função do tempo de execução de $\mathscr{A}$, que resolve o problema ICP com vantagem $\varepsilon/2$.

Prova. O desafiante manda uma instância (t, B_J^{pk}) do problema ICP para o algoritmo $\mathscr{B}$, que escolhe $s \in I$. $\mathscr{A}$ solicita um desafio sobre o par de textos claros $(m_0, m_1) \in \mathscr{P}$, $\mathscr{B}$ escolhe aleatoriamente o bit $b \in \{0,1\}$ e devolve para $\mathscr{A}$ o valor $\psi = m_b + ts \pmod{B}_J^{\text{pk}}$. O algoritmo $\mathscr{A}$ retorna o palpite β' e $\mathscr{B}$ computa seu próprio palpite $b' = \beta \oplus \beta'$.

Se $b = 0$, ψ corresponde a um texto encriptado válido, pois temos que $\psi = m_b + rs \pmod{B_J^{\text{pk}}}$. Com isso, o algoritmo $\mathscr{A}$ tem vantagem ε. Se $b = 1$, t é uniforme módulo J. Como o ideal gerado por s é primo com J, ts é uniforme módulo J e consequentemente ψ é um elemento aleatoriamente uniforme em $R \pmod{B}_J^{\text{pk}}$, portanto é independente de β, de forma que $\mathscr{A}$ tem vantagem zero. Juntando ambas as possibilidades, temos que a vantagem de $\mathscr{B}$ é $\varepsilon/2$. □

4.3.5. Ideais em anéis polinomiais e reticulados ideais

Considere o anel polinomial $R = \mathbb{Z}[x]/f(x)$, onde $f(x)$ é um polinômio irredutível de grau N sobre $\mathbb{Z}[x]$. Seja $a(x) \in \mathbb{Z}[x]/f(x)$, o ideal gerado por $a(x)$ é formado por todos os polinômios múltiplos de $a(x)$ módulo $f(x)$. Este ideal pode ser representado por um reticulado $\mathscr{L}_R$, onde a base deste reticulado é dada pelos vetores gerados pelos coeficientes dos polinômios $a(x) \pmod{f(x)}$, $x.a(x) \pmod{f(x)}$, $x^2.a(x) \pmod{f(x)}$, ..., $x^{N-1}.a(x) \pmod{f(x)}$, denominada *base de rotação*. Estes vetores são linearmente independentes no espaço vetorial com a base canônica. Em geral, não é necessário que o ideal seja gerado por apenas um polinômio, assim como também não é obrigatório utilizar uma base de rotação.

Dado um ideal sobre um anel polinomial, é possível determinar um reticulado tal que todo ponto do reticulado corresponde a um polinômio do ideal. Este reticulado é denominado ***reticulado ideal***.

A ***forma normal de Hermite*** (*Hermite normal form - HNF*) de um reticulado $\mathscr{L}_R$ é uma ***base triangular superior***, que pode ser computada eficientemente a partir de uma base qualquer do mesmo reticulado, sendo apropriada para ser utilizada como chave pública.

Reticulados ideais são apropriados para concretizarem o esquema abstrato discutido anteriormente, porque a operação de redução modular por uma base B_I é facilmente computada como o elemento pertencente ao paralelepípedo fundamental centralizado $\mathscr{P}(B_I)$. Dado $t \in R$, a redução modular $t \pmod{B_I}$ é computada da seguinte forma

$$t - B_I.\lfloor B_I^{-1} t \rceil.$$

4.3.6. O esquema concreto

Nesta seção será apresentado o esquema $\mathscr{E}$ que concretiza a proposta abstrata discutida anteriormente, utilizando reticulados ideais.

Definição 4.3.2. Seja r_{enc} o menor valor tal que $X_{\text{enc}} \in \mathscr{B}(r_{\text{enc}})$, onde $\mathscr{B}(r)$ é a esfera de raio r. Analogamente, r_{dec} é definido como o menor valor tal que $X_{\text{dec}} \in \mathscr{B}(r_{\text{dec}})$.

Com isso, o conjunto de circuitos permitidos é dado por

$$\mathbf{S_C} = \{\mathbf{C} \mid \forall (x_1, ..., x_t) \in \mathscr{B}(r_{\text{enc}})^t, g(\mathbf{C})(x_1, ..., x_t) \in \mathscr{B}(r_{\text{dec}})\}.$$

Comparando com o esquema abstrato, X_{enc} e X_{dec} foram substituídos por $\mathscr{B}(r_{\text{enc}})$ e $\mathscr{B}(r_{\text{dec}})$, respectivamente.

Para compreender que classe de circuitos $\mathscr{E}$ consegue avaliar, é preciso estabelecer limites para o tamanho dos vetores resultantes da soma e multiplicação de quaisquer dois vetores. Dados u e v, pela desigualdade triangular, temos que $||u+v|| \leq ||u|| + ||v||$. Já a multiplicação depende do anel R para que possamos estabelecer tal limite. Sendo assim, dizemos que $||u.v|| \leq \gamma_{\text{Mult}}(R).||u||.||v||$, onde $\gamma_{\text{Mult}}(R)$ é um fator dependente de R.

Teorema 4.3.3. Suponha que $r_{\text{enc}} \geq 1$. Dado um circuito $\mathbf{C}$, cujas portas aditivas possuam γ_{Mult} parâmetros de entrada e cujas portas multiplicativas possuam dois parâmetros de entrada, se a profundidade de $\mathbf{C}$ é no máximo $\log\log r_{\text{dec}} - \log\log \gamma_{\text{Mult}}.r_{\text{enc}}$, então $\mathbf{C}(x_1,...,x_t) \in \mathscr{B}(r_{\text{dec}})$, para todo $(x_1,...,x_t) \in \mathscr{B}(r_{\text{enc}})$.

Prova. Considere um circuito $\mathbf{C}$ com profundidade d. Seja r_i um limite superior para a norma dos valores de $\mathbf{C}$ no nível i, onde o nível d representa a entrada do circuito e r_0 representa a sua saída. Uma porta aditiva no nível i gera uma saída $v_+ \in R$, tal que $||v_+|| \leq \gamma_{\text{Mult}}.r_i$, enquanto uma porta multiplicativa no mesmo nível, gera uma saída $v_\times \in R$, tal que $||v_\times|| \leq \gamma_{\text{Mult}}.r_i^2$. Portanto, na pior das hipóteses temos que $r_{i-1} \leq \gamma_{\text{Mult}}.r_i^2$. Dado que $\mathbf{C}$ recebe textos encriptados válidos como entrada, temos que $r_d \leq r_{\text{enc}}$, e com isso, obtemos

$$r_0 \leq (\gamma_{\text{Mult}}.r_{\text{enc}})^{2^d}.\square$$

Portanto, para maximizar a profundidade de circuito que o esquema $\mathscr{E}$ é capaz de lidar homomorficamente, é preciso minimizar γ_{Mult} e r_{enc}. Por outro lado, é preciso maximizar r_{dec}. Porém, a segurança semântica de $\mathscr{E}$ está relacionada à razão $r_{\text{dec}}/r_{\text{enc}}$ [Gen09b]. Para que o esquema seja capaz de decriptar, é necessário que $r_{\text{dec}} < \lambda_1(J)$, e para que o algoritmo ***LLL*** não possa ser usado para atacar o esquema, é necessário que $\lambda_1(J)/r_{\text{enc}}$ não seja muito grande. Isto é, $r_{\text{dec}} = 2^{n_1^c}$ e $\lambda_1(J)/r_{\text{enc}} = 2^{n_2^c}$, para $0 < c_1, c_2 < 1$ é uma escolha que permite avaliar circuitos de profundidade $(c_1 - c_2)\log n$.

Lema 4.3.1. Seja B uma base de um determinado reticulado e $B^* = (B^{-1})^T$. Seja r o raio da maior esfera centrada na origem tal que $\mathscr{P}(B)$ a circunscreve. Então $r = 1/(2.||B^*||)$. Em particular,

$$r_{\text{dec}} = 1/(2.||((B_J^{\text{sk}})^{-1})^T||).$$

Suponha que $||t|| \leq r$, então cada coeficiente de $B^{-1}t$ tem magnitude no máximo $1/2$.

Prova. Cada coeficiente de $B^{-1}t$ é o produto interno de t e uma coluna de B^*, portanto tem magnitude no máximo $||t||.||B^*|| < 1/2$, o que implica que $\lfloor B^{-1}t \rceil = 0$. Assim, $t = t \pmod B$ e portanto $t \in \mathscr{B}$. Seja v o maior vetor de B^* e seja x um vetor paralelo a v. Então, se o produto interno entre v e x for estritamente maior que $1/2$, temos que $x \notin \mathscr{P}(B)$, se e somente se $||x|| > 1/(2||B^*||)$. $\square$

O algoritmo IdealGen pode computar B_J^{sk} por meio da base de rotação de um vetor v pequeno e paralelo a $e_1 = (1,0,...,0)$ e $B_J^{\text{pk}} = HNF(B_J^{\text{sk}})$. Com isso, obtemos $r_{\text{dec}} = ||v||/2$.

4.3.7. Redução do circuito de decriptação

Nesta seção serão introduzidos dois ajustes que serão responsáveis por tornar o circuito de decriptação do esquema capaz de ser avaliado homomorficamente.

Ajuste 1. Redefina o conjunto de circuitos permitidos $\mathbf{S_C}$ substituindo $\mathscr{B}(r_{\text{dec}})$ por $\mathscr{B}(r_{\text{dec}})/2$.

Lema 4.3.2. Após o ajuste 1, os coeficientes de $(B_J^{\text{sk}})^{-1}\psi$ distam $1/4$ de um inteiro, onde ψ é um texto encriptado válido.

Ajuste 2. Compute um vetor pequeno $v_J^{\text{sk}} \in J^{-1}$, tal que existe $u \in I+1$ e $u(v_J^{\text{sk}})^{-1} \in I+1$. Além disso, modifique $\mathbf{S_C}$ de modo a usar o seguinte limite para a decriptação

$$\mathscr{B}(2r_{\text{dec}}/(n^{1.5}\gamma_{\text{Mult}}(R)^2||B_I||)).$$

Com este segundo ajuste, a decriptação pode ser realizada da seguinte forma:

$$\psi - B_J^{\text{sk}}.\lfloor (B_J^{\text{sk}})^{-1}\psi \rceil \pmod{B_I} = \psi - \lfloor v_J^{\text{sk}}\psi \rceil \pmod{B_I}.$$

Basicamente a mesma estratégia que foi adotada com números inteiros será também seguida com reticulados ideais. Ou seja, o problema SSSP é introduzido de modo que o texto encriptado passa a conter uma sequência de valores, que ao serem somados permitem a decriptação da mensagem. Para calcular a somatória é utilizada a mesma técnica baseada em polinômios simétricos elementares para computar os bits do ***peso de Hamming***. Porém, o espaço de texto claro $\mathscr{P}$ precisa ser restrito a $\{0,1\}$ e o ideal I deve ser simplesmente $(2.e_1)$ para obter o resultado desejado.

São definidos dois novos algoritmos para formalizar as ideias anteriormente descritas: SplitKey e ExpandCT. O primeiro é responsável por modificar o par de chaves original, acrescentando uma instância do problema SSSP, de modo que a nova chave privada passa a conter os índices i dos elementos t_i que devem ser somados para obter o vetor v_J^{sk}, assim como ocorreu anteriormente para esconder $1/p$. O segundo modifica o texto encriptado retornado pelos algoritmos Enc e Eval, de modo que, dado ψ válido, o novo texto encriptado é uma sequência de valores $t_i\psi$. Isto é, parte da computação do algoritmo Dec já é realizada pelo algoritmo ExpandCT, restando apenas calcular a somatória dos valores onde i corresponde a um índice válido para a solução do ***problema SSSP***.

Modificação da chave. Sejam as funções $\gamma_{\text{set}}(n)$, simultaneamente pertencente a $\omega(n)$ e a $\text{poly}(n)$ (correspondente a Θ) e $\gamma_{\text{subset}}(n)$, tais que $\gamma_{\text{subset}}(n)$ é ao mesmo tempo $\omega(1)$ e $o(n)$ (correspondente a θ). O algoritmo SplitKey gera $\gamma_{\text{set}}(n)$ valores de t_i em $J^{-1} \pmod{B_I}$, de modo que exista um conjunto T, composto de $\gamma_{\text{subset}}(n)$ valores de t_i,

que somados resultam em $v_J^{\mathrm{sk}} + I$. Os índices são representados por um conjunto de bits $\{s_i\}_0^{\gamma_{\mathrm{set}}(n)}$, onde $s_i = 1$ se e somente se t_i pertence a T. A chave pública é modificada para incluir os valores de t_i, enquanto a chave privada é alterada para conter os índices i da solução do problema SSSP.

Expansão do texto encriptado. Dado ψ um texto encriptado válido segundo o esquema $\mathscr{E}$ original, então o algoritmo ExpandCT retorna $c_i = t_i \psi \pmod{B_I}$.

4.3.8. Autoinicialização

Considere a existência de um esquema criptográfico $\mathscr{E}$ capaz de avaliar compactamente uma classe $\mathbf{S_C}$ de circuitos. Em outras palavras, dado um circuito $\mathbf{C} \in \mathbf{S_C}$, o esquema $\mathscr{E}$ é homomórfico em relação à $\mathbf{C}$, portanto existe um circuito $\mathbf{C}'$, estruturalmente idêntico a $\mathbf{C}$, mas que aceita como entrada $\mathrm{Enc}(\mathrm{pk}, m)$ ao invés de m. Então é possível utilizar $\mathscr{E}$ para obter um novo criptossistema $\mathscr{E}'$, capaz de avaliar circuitos de profundidade arbitrária. Para que isso seja possível é necessário que $\mathscr{E}$ seja capaz de avaliar seu próprio circuito de decriptação, acrescentado de uma operação de soma ou multiplicação.

Utilizando d para denotar a profundidade do circuito, representamos por $\mathscr{E}^{(d)}$ o esquema criptográfico capaz de avaliar compactamente circuitos de profundidade no máximo d. O esquema $\mathscr{E}$ pode ser usado para construir $\mathscr{E}^{(d)}$ repetindo o procedimento descrito no parágrafo anterior d vezes.

Mas até o momento ainda não foi descrito o circuito de decriptação que será utilizado. Seguindo a estratégia da seção 4.2.3.6, define-se a variável $a_i = s_i.c_i$, isto é, aqueles valores de c_i correspondentes a uma solução do problema SSSP. Define-se também um conjunto de valores $\{w_i\}_0^{\gamma_{\mathrm{subset}}(n)+1}$ cuja soma, após tomado o inteiro mais próximo de cada coordenada, seja igual a soma dos valores de a_i. Assim, é possível utilizar o algoritmo 4.2.3.6 para computar eficientemente o circuito de decriptação.

De acordo com a escolha de parâmetros, atacar o problema SSSP tem complexidade $2^{\gamma_{\mathrm{subset}}(n)}$. Porém, quanto maior é o valor de $\gamma_{\mathrm{subset}}(n)$, maior é o fator de aproximação para o ***problema CVP***. Considerando que um fator de aproximação de 2^k leva tempo $2^{n/k}$, é preciso escolher $\gamma_{\mathrm{subset}}(n)$ de maneira que ambos os problemas sejam difíceis de atacar. Ou seja, se $\gamma_{\mathrm{subset}}(n) = \sqrt{n}$, a complexidade do problema é aproximadamente $2^{\sqrt{n}}$. Assim, n é escolhido de forma que $n \approx \lambda^2$. Utilizando FFT, é possível obter complexidade aproximadamente de λ^6 para o algoritmo de decriptação [Gen09b]. Em relação ao esquema definido sobre inteiros, o tamanho do parâmetro $q \approx \lambda^5$, para garantir que o problema do mdc aproximado seja difícil, é um fator que torna o esquema baseado em inteiros menos eficiente que a versão baseada em reticulados ideais.

4.4. Trabalhos recentes

Nesta seção será apresentada uma compilação de trabalhos recentes, que propõe a utilização do problema LWE polinomial para obter um esquema homomórfico restrito, isto é, capaz de avaliar uma classe limitada de circuitos algébricos. Além disso, será discutido o uso prático deste tipo de criptossistema, já que muitas aplicações interessantes não requerem a existência de um esquema completamente homomórfico.

4.4.1. ECH sem *autoinicialização*

Até o momento, todas as propostas de ECH apresentadas seguem o modelo desenvolvido por Craig Gentry, onde primeiramente é construído um esquema homomórfico restrito, seguido de uma redução do circuito de decriptação, para finalmente utilizar a ***autoinicialização***. Porém, este modelo possui limites claros em relação à performance mínima que pode ser obtida. Em um trabalho recente são apresentados argumentos que mostram que a *autoinicialização* tem complexidade mínima de $\Omega(\lambda^4)$ [BGV11]. Stehlé e Steinfeld propuseram uma otimização (não tem sido utilizada) que permite reduzir o grau da decriptação para $O(\sqrt{\lambda})$, de modo que a complexidade mínima da ***autoinicialização*** pode tornar-se $\Omega(\lambda^{3.5})$.

Recentemente, em dois trabalhos distintos, Gentry e Halevi [GH11a] e Brakerski e Vaikuntanathan [BV11] encontraram formas de desviar deste modelo principal. No primeiro trabalho, o circuito de decriptação é descrito por polinômios simétricos, cuja computação pode ser realizada por um circuito de profundidade 3, onde o primeiro e terceiro níveis são constituídos apenas de somas, enquanto o segundo nível é constituído de multiplicações. Assim, para evitar o aumento quadrático do ruído, as multiplicações são realizadas utilizando um criptossistema como o ElGamal, capaz de multiplicar homomorficamente. No segundo trabalho, são utilizadas duas técnicas: ***redução de dimensão*** e ***redução de módulo***. Para isso, o esquema é baseado no problema LWE, introduzindo uma importante mudança na construção de ECH eficiente.

Mas apesar das novas ideias, a performance ainda era limitada inferiormente por $\Omega(\lambda^4)$. Em outro trabalho, Gentry, Brakerski e Vaikuntanathan [BGV11] utilizam algumas das ideias anteriores e o problema LWE em anéis (RLWE) para obter um esquema que não precisa de autoinicialização.

Definição 4.4.1. O ***problema LWE*** consiste em encontrar o vetor $s \in \mathbb{Z}_q^n$, dadas as equações

$$\begin{array}{rcl} \langle s, a_1 \rangle & \approx_{\mathscr{D}} & b_1 \pmod q \\ \langle s, a_2 \rangle & \approx_{\mathscr{D}} & b_2 \pmod q \\ & \vdots & \end{array}$$

A notação $\approx_{\mathscr{D}}$ significa uma tolerância na igualdade, de acordo com a distribuição $\mathscr{D}$. Ou seja, $\langle s, a_i \rangle$ difere de b_i e esta diferença é determinada pela distribuição $\mathscr{D}$, geralmente tomada como sendo a distribuição normal. Alternativamente, podemos escrever $\langle s, a_i \rangle = b_i + e_i$, onde $e_i \in \mathscr{D}$.

A quantidade de equações contribui relativamente pouco para a solução do problema. Existe um compromisso entre o número de equações e o tempo de execução para encontrar a solução do problema, mesmo com uma quantidade arbitrária de equações a complexidade é na melhor das hipóteses subexponencial [BKW03].

Em 2005, Oded Regev apresenta uma redução quântica do problema LWE ao pior caso de problemas em reticulados. Além disso, este trabalho mostra um novo criptossis-

tema, cuja performance é consideravelmente melhor que outros esquemas baseados em reticulados [Reg05].

Lyubaskevsky, Peikert e Regev definiram uma versão similar ao problema LWE, mas usando anéis polinomiais [LPR10]. Seja $f(x) = x^d + 1$, onde d é uma potência de 2. Dado um inteiro q e um elemento $s \in R = \mathbb{Z}_q[x]/f(x)$, o **problema LWE em anel** sobre R, com relação a uma distribuição $\mathscr{D}$, é definido equivalentemente, ou seja, é preciso encontrar s que satisfaça as seguintes equações:

$$\begin{aligned} s.a_1 &\approx_{\mathscr{D}} b_1 \pmod R \\ s.a_2 &\approx_{\mathscr{D}} b_2 \pmod R \\ &\vdots \end{aligned}$$

onde a_i e b_i são elementos de R e a redução modular em R é o mesmo que reduzir o polinômio resultante módulo $f(x)$ e seus coeficientes módulo q.

4.4.2. Criptossistema homomórfico restrito

A seguir é apresentado o criptossistema que será utilizado como base da construção final, sendo usada a notação $\mathscr{E}_R(\lambda, \mu)$ para referência a este esquema.

Definição 4.4.2. Configuração. Dado o parâmetro de segurança λ e um parâmetro secundário μ, escolhemos um inteiro q com μ bits e $N = \lceil 3 \log q \rceil$.

Geração de chaves. Utilize a distribuição $\mathscr{D}$ para obter o polinômio s^*, denotando por s o vetor de tamanho 2 formado pelos polinômios 1 e s^*. A chave privada é dada por $\mathrm{sk} = s$. Gere aleatoriamente uma matriz A' de N linhas e uma coluna, cujos elementos sejam polinômios com coeficientes uniformemente escolhidos em $\mathbb{Z}_q$. Utilize a distribuição $\mathscr{D}$ para gerar N polinômios e_i e compute $b = A's^* + 2e$. Compute a matriz A de duas colunas, sendo a primeira igual a b e a segunda igual a $-A'$. A chave pública é dada por $\mathrm{pk} = A$. Por construção, temos que $As = 2e$.

Encriptação. Dada uma mensagem $m \in \{0,1\}$, define-se a matriz m' de duas linhas, onde a primeira é o próprio m e a segunda é igual a zero. Gere aleatoriamente a matriz de polinômios binários r, com N linhas. Compute

$$c = m' + A^T r.$$

Decriptação. Compute $m = [[\langle c, s \rangle]_q]_2$.

A corretude deste criptossistema é facilmente verificada usando a relação $As = 2e$ e o fato de q ter sido escolhido suficientemente grande para que o acúmulo de erro não ultrapasse $q/2$, semelhantemente ao caso sobre números inteiros.

4.4.3. Redução de dimensão

O algoritmo de decriptação descrito anteriormente assemelha-se ao ElGamal, porque o texto encriptado é composto por dois polinômios, $c = [c_0, c_1]$, enquanto a chave privada é dada por $s = [1, s^*]$. Portanto, a decriptação pode ser representada por

$$m = [c_0 + c_1 s^*]_q \pmod 2.$$

A expressão $c_0 + c_1 s^*$ é um polinômio de grau 1 em $s?*$. Para multiplicar dois textos encriptados, $c = \text{Enc}(\text{pk}, m)$ e $c' = \text{Enc}(\text{pk}, m')$, podemos computar

$$(c_0 + c_1 s^*)(c'_0 + c'_1 s^*) = c_0 c'_0 + (c_0 c'_1 + c'_0 c_1) s^* + c_1 c'_1 (s^*)^2.$$

Se q for suficientemente grande, ao substituir s pela chave privada na expressão anterior, obtemos um polinômio que pode ser usado para recuperar $m.m'$. Porém, a multiplicação faz com que o texto encriptado esteja em um espaço de dimensão maior. Para que o criptossistema seja compacto, o texto encriptado não pode crescer desta maneira, de modo que é preciso um algoritmo para redução da dimensão. Esta tarefa será realizada por meio de um algoritmo denominado SwitchKey, que, com base em parâmetros públicos, retorna um texto encriptado que pode ser normalmente decriptado.

Dado um polinômio x, considere o algoritmo BitDecomp, que retorna $\log q$ polinômios binários x_i, computados pela representação dos coeficientes de x na base 2. Isto é,

$$x = \sum 2^i x_i.$$

Além disso, considere o algoritmo PowerOf2, que retorna $\log q$ polinômios na forma $2^i x$, como segue:

$$\text{PowerOf2}(x) = [x, 2x, ..., 2^{\lfloor \log q \rfloor} x].$$

Por construção, temos que

$$\langle \text{BitDecomp}(c), \text{PowerOf2}(s) \rangle = \langle c, s \rangle \pmod q.$$

Assim, é possível definir da seguinte forma o algoritmo SwitchKeyGen:

1. dado um vetor de polinômios $\bar{s}$, derivado da chave privada s, compute uma nova chave pública $\bar{A}$, correspondente a $\bar{s}$, com $\bar{N}$ linhas, onde $\bar{N} = 3\log^2 q$;
2. retorne $\bar{B} = \bar{A} + \text{PowerOf2}(\bar{s})$, onde $\text{PowerOf2}(\bar{s})$ é adicionado a primeira coluna de $\bar{B}$.

Com isso, dado um texto encriptado expandido $\bar{c}$, o algoritmo SwitchKey pode ser definido simplesmente como

$$\text{SwitchKey}(\bar{c}) = \text{BitDecomp}(\bar{c})^T \bar{B}.$$

Em resumo, a matriz $\bar{B}$ funciona como alternativa ao uso do problema SSP, descrito nos esquemas anteriores. Ou seja, é a encriptação da chave privada usando sua

própria chave pública, de modo que estamos novamente assumindo segurança circular. É possível redefinir os algoritmos SwitchKeyGen e SwitchKey, de maneira a utilizar uma cadeia de chaves, mas para simplificar a exposição, foi adotada esta estratégia.

4.4.4. Redução de módulo

Os criptossistemas definidos até agora possuem um problema em comum: o ruído cresce de forma quadrática a cada multiplicação. Para que um esquema seja considerado completamente homomórfico, é necessário que ele seja capaz de avaliar uma quantidade arbitrária de operações de soma ou multiplicação. Portanto, o crescimento quadrático do ruído é um problema que deve ter atenção especial. Para superar este obstáculo, Brakerski e Vaikuntanathan [BV11] propuseram uma nova técnica para gerenciamento do ruído.

Basicamente, se o ruído inicial é proporcional a r, após k multiplicações este ruído passa a ser proporcional a r^{2^k}. A solução encontrada foi utilizar uma cadeia decrescente de módulos $q_i \approx q/r^i$. Após a primeira multiplicação, ajusta-se o texto encriptado c, multiplicando-o por $1/r$ e corrigindo a paridade se necessário, e troca-se o módulo q por q/r. Esta mudança parece não trazer nenhum ganho e não pode ser realizada arbitrariamente, pois a cadeia decrescente chega rapidamente (linearmente em relação a profundidade do circuito) em um valor mínimo. Porém, é fácil mostrar que o ruído é reduzido na mesma proporção $1/r$, ou seja, após a k-ésima multiplicação, obtemos ruído proporcional a r^k, ao invés de r^{2^k}. Logo, há um ganho exponencial nesta transformação. Quando a cadeia decrescente chega ao fim, é necessário usar a autoinicialização para retornar ao topo da cadeia.

Sendo assim, dado um vetor de polinômios x, o algoritmo $\text{Scale}(x, q_i, q_{i+1})$ computa o vetor de polinômios mais próximo a $(q_{i+1}/q_i)x$, tal que

$$\text{Scale}(x, q_i, q_{i+1}) = x \pmod 2.$$

4.4.5. BGV

Com isso, definimos nesta seção o esquema ***BGV*** (Brakerski, Gentry, Vaikuntanathan [BGV11]), capaz de avaliar circuitos de profundidade multiplicativa L.

Definição 4.4.3. Configuração. Dado o parâmetro de segurança λ e a profundidade multiplicativa L, compute $\mu = \theta(\log\lambda + \log L)$. Para i variando de L a 0, configure o esquema $\mathscr{E}_{R,i}(\lambda, (i+1)\mu)$, obtendo uma cadeia decrescente de módulos, começando com q_L, que possui $(L+1)\mu$ bits, até q_0, que possui μ bits.

Geração de chaves. Utilize a geração de chaves do esquema $\mathscr{E}_R$ para cada nível i do circuito. Compute $s'_i = s_i \otimes s_i$ e $s''_i = \text{BitDecomp}(s_i, q_i)$, onde BitDecomp recebe q_i por parâmetro, já que agora existem $L+1$ possibilidades para q_i. Finalmente, compute $\bar{B}_i = \text{SwitchKeyGen}(s''_i, s_{i-1})$, para $i > 0$. A chave privada é formada pelos valores de s_i, enquanto a chave pública corresponde às chave públicas de $\mathscr{E}_{R,i}$, acrescidas de $\bar{B}_i$.

Encriptação. Dado o bit m, compute $\mathscr{E}_{R,L}.\text{Enc}(\text{pk}_L, m)$.

Decriptação. Dado um texto encriptado c, no nível k do circuito, utilize a chave privada s_k para computar $m = \mathscr{E}_{R,k}.\text{Dec}(s_k, c)$.

A soma de textos encriptados é realizada pela soma individual dos polinômios, enquanto a multiplicação é realizada pelo produto tensorial dos textos encriptados, obtendo assim um vetor composto por 3 polinômios, denominado texto encriptado expandido, sendo então necessário utilizar o algoritmo Recrypt, definido a seguir, de modo que o texto encriptado volte a ser composto por 2 polinômios.

Dado o texto encriptado expandido $\bar{c}$, q_i e q_{i+1}, o algoritmo Recrypt calcula

$$c_1 = \text{PowerOf2}(\bar{c}, q_i).$$

Neste momento, a seguinte condição é válida: $\langle c_1, s''_j \rangle = \langle c, s'_j \rangle$. Agora, é possível utilizar os algoritmos de redução de dimensão e redução de módulo, isto é, calcula-se $c_2 = \text{Scale}(c_1, q_{i+1}, q_i)$ e a saída do algoritmo é dada por

$$\text{SwitchKey}(c_2, q_i, \bar{B}_i).$$

4.4.6. Operações em bloco

Nesta seção será descrita uma importante otimização sobre o esquema anterior. A ideia consiste em utilizar o teorema chinês dos restos para permitir a operação simultânea sobre um vetor de mensagens. Na literatura, é feita uma analogia ao modelo SIMD, pela capacidade de operar sobre vetores de palavras [SV11].

Com esta otimização é possível reduzir a computação homomórfica de cada nível do circuito para complexidade polilogarítmica, representando assim um grande ganho em relação ao limite anterior de $\Omega(\lambda^{3.5})$. Porém, o circuito a ser avaliado homomorficamente deve ter largura média de $\Omega(\lambda)$.

Além disso, a capacidade de realizar somas e multiplicações sobre vetores não é um modelo computacional completo, porque não é equivalente ao modelo de circuitos algébricos. É necessária uma maneira de permutar os elementos dentro de um determinado vetor, caso contrário não seria possível computar funções relacionando elementos de diferentes posições do vetor. Para resolver este problema foi utilizado o automorfimo de

Frobenius, que permite rotacionar os elementos do vetor, e uma rede de permutação, que permite combinar rotações a esquerda e a direita para realizar permutações mais complicadas.

Matematicamente, sejam m e q inteiros tais que $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ contém uma raíz m-ésima primitiva da unidade, $w \in \mathbb{Z}_q$, então o m-ésimo polinômio ciclotômico, $\Phi_m(x)$, pode ser fatorado em termos lineares módulo q

$$\Phi_m(x) = \prod (x - w^i) \pmod{q}.$$

Um polinômio em $\mathbb{Z}_q[x]/\Phi_m(x)$ pode ser representado por seus coeficientes, como vínhamos fazendo até agora, ou então pode ser representado por sua avaliação exatamente nas raízes m-ésimas primitivas da unidade. Assim, existem duas possíveis representações: por coeficientes, ou por avaliação. A primeira denotaremos por $\overrightarrow{c}$, enquanto a segunda denotaremos por $\overrightarrow{d}$, de modo que as representações estão relacionadas pela matriz de Vandermonde V_m da seguinte maneira

$$\overrightarrow{d} = V_m \overrightarrow{c}.$$

Considerando q como o produto de primos p_i, podemos de fato utilizar duas vezes o teorema chinês dos restos (TCR), já que os próprios elementos de $\mathbb{Z}_q$ podem ser decompostos de acordo os fatores de q. Esta representação é chamada de **TCR dupla**.

4.4.7. AES homomórfico

Recentemente, Craig Gentry Shai Halevi e Nigel Smart [GHS12b] apresentaram um resultado prático importante: a avaliação homomórfica do AES-128. A implementação foi realizada utilizando a biblioteca NTL sobre GMP. Foi usado o esquema BGV, com a representação TCR dupla, de forma que o modelo SIMD foi utilizado para computar sobre os blocos do AES. A execução de uma rodada completa do AES demorou aproximadamente uma semana, realizada por um computador com 256 GB de memória RAM.

A escolha do AES é de especial importância, porque por um lado possui estrutura que permite o uso das técnicas descritas na seção anterior, para operações em bloco, e, por outro lado, permite transformar um texto encriptado obtido pelo AES em um texto encriptado homomórfico, de modo que o dado pode ser manipulado sem que em nenhum momento o fique desprotegido, isto é, em claro.

Este trabalho utiliza a versão mais eficiente de ECH construída até o momento, propondo uma técnica nova para o gerenciamento dinâmico do ruído. Uma estimativa do ruído é acrescentada ao texto encriptado. Assim, foi possível minimizar a quantidade de vezes que a reencriptação é necessária. Por sua vez, isto é interessante porque este algoritmo precisa transformar da representação TCR dupla para a representação em coeficientes e esta transformação requer bastante processamento. De fato, a transformação entre representações é realizada pelo algoritmo FFT e FFT inverso de um vetor de polinômios. Com isso, este trabalho pode ser considerado como um marco importante para a encriptação homomórfica.

4.5. Aplicações

Existem diversas aplicações, tanto práticas como teóricas para homomorfismos secretos. Muitas dessas aplicações não requerem a existência de homomorfismos completos, isto é, que permitem uma quantidade arbitrária de operações de soma e multiplicação. Na tese de doutorado de Dörte K. Rappe [Rap06], são descritas diversas aplicações de criptossistemas baseados em homomorfismos. Fazemos aqui um breve resumo dessas aplicações.

4.5.1. Agentes móveis

Uma aplicação interessante do uso de homomorfismo secreto é na proteção de agente móveis [ST98]. Uma preocupação neste cenário é a possibilidade de ataques de um servidor malicioso para obtenção de dados sigilosos ou para deduzir informações a respeito de determinada computação. Com o uso de criptografia baseada em homomorfismos, é possível computar sobre dados criptografados ou então é possível computar sobre funções criptografadas. Dependendo do cenário em questão, é possível utilizar a alternativa mais adequada.

4.5.2. Computação multiparte

Neste cenário, um grupo de indivíduos estão interessados em calcular uma função f, de modo que cada indivíduo contribua com uma parte dos parâmetros de entrada da função, e tal que terceiros, que previamente não tinham conhecimento desses parâmetros, não passem a conhecê-los após a computação de f. Homomorfismos secretos permitem que a função f seja computada utilizando a forma encriptada dos parâmetros de entrada, obtendo como retorno a encriptação da saída de f computada sobre os parâmetros em claro.

4.5.3. Compartilhamento de segredo

Neste contexto, o homomorfismo algébrico implica que a composição dos segredos compartilhados é igual ao compartilhamento dos segredos compostos, solucionando o problema de forma elegante.

4.5.4. Assinaturas

Dado um conjunto de ***assinaturas*** válidas para um determinado conjunto de dados, é possível construir uma nova assinatura, que correspondente a avaliação de uma função f, sobre um subconjunto desses dados. Este é um tema que tem sido pouco explorado, como argumenta Patrick Schmidt [Sch11] em sua dissertação. Em outro trabalho interessante, Dan Boneh e David Freeman [BF11] apresentam um esquema capaz de avaliar uma classe restrita de funções. A proposta é semelhante a ECH sobre reticulados ideais de Craig Gentry.

4.5.5. Conhecimento nulo

De forma simplificada, em contextos que utilizam o conceito de *conhecimento nulo* alguém (Bob) deseja demonstrar (para Alice) que possui uma determinada informação sem que seja necessário revelá-la. De fato, Bob deseja que **nenhuma** informação seja revelada. Esta primitiva criptográfica tem grande importância teórica e prática. A utilização de homomorfismos permite que Bob encripte a informação de forma que Alice ainda

consiga validar uma determinada propriedade algébrica desta informação. Como o homomorfismo preserva a estrutura algébrica, a encriptação da informação preserva as suas propriedades algébricas.

4.5.6. Eleições

Este é um contexto de peculiar importância já que, cada vez mais, países estão utilizando urnas eletrônicas na escolha de seus governantes. Este é um exemplo de cenário em que não é necessária a utilização de um homomorfismo completo, já que deseja-se apenas somar 1 a uma quantia de votos, mas não é necessário multiplicar quantias de votos. Sendo assim, esquemas parcialmente homomórficos são suficientes para tornar esta aplicação prática. Nos sistemas de votação Votebox e Helios [SDW08], é utilizada uma adaptação do criptossistema ElGamal para permitir contagem dos votos encriptados, de modo a garantir o sigilo de cada voto e decriptar o resultado apenas na hora da contagem de votos, utilizando a chave secreta do esquema.

4.5.7. Ofuscação

Em um outro trabalho [DMMQN11] é apresentada uma proposta para o uso de ECH no contexto de ofuscação. Contudo, usando ECH o programa ofuscado produz como saída um texto encriptado. Para lidar com a situação, o receptor do programa ofuscado precisa ser capaz de provar que não é malicioso, para, com o auxílio de um hardware com características especiais, conseguir decriptar a saída do programa.

Propostas anteriores não permitiam múltiplas execuções do mesmo programa ofuscado, ou então precisam de um hardware distinto para cada nível do circuito a ser avaliado. Portanto, mesmo com os resultados negativos sobre a possibilidade real de ofuscação, a ECH permite a construção de esquemas melhores quando se supõe um modelo de segurança menos restritivo.

4.5.8. Encriptação parcialmente homomórfica

As otimizações propostas, principalmente sobre o esquema BGV, além de contribuir diretamente na tarefa de tornar prática a ECH, permitem a construção de esquemas de encriptação parcialmente homomórfica (EPH), capazes de solucionar diversos problemas práticos [NLV11b]. Nesse trabalho, uma prova de conceito é desenvolvida na linguagem aritmética Magma, mostrando que existe uma liberdade na escolha de parâmetros do esquema BGV, de modo que é possível adaptá-lo de acordo com o circuito a ser avaliado homomorficamente. Diversas escolhas de parâmetros são sugeridas para diferentes cenários, dependendo da possibilidade de uso de operações em bloco, da quantidade de multiplicações envolvidas e da profundidade do circuito.

Recentemente, foi apresentado o sistema CryptDB [PRZB11], que é um banco de dados sobre dados encriptados. São utilizadas diversas primitivas criptográficas para permitir consultas SQL arbitrárias. Existem algumas limitações em relação a certos tipos de junções, que na prática são pouco frequentes. As operações foram muito bem definidas e organizadas nos seguintes grupos: (i) verificação de igualdade; (ii) comparação de ordem; (iii) operações aritméticas; e (iv) junções. O esquema é formado por camadas aninhadas de encriptações para resolver cada um desses grupos. Para a execução de operações arit-

méticas é usado o criptosisstema homomórfico Paillier, que é capaz de efetuar somas (mas não permite multiplicações). A construção oferece confidencialidade, considerando um modelo de adversário passivo, mas os próprios autores supõem uma segurança que não é perfeita, porque o adversário consegue, por exemplo, ordenar os dados. Por outro lado, é uma abordagem prática interessante, porque além de eficiente também é transparente para o usuário, já que o servidor interpreta dinamicamente as consultas SQL, mapeando-as em funções internas do banco de dados. Com isso, é possível oferecer proteção contra o próprio administrador do banco de dados.

Outro trabalho relacionado é a proposta de uma linguagem de domínio específico (*domain specific language*) para computação em nuvem [BMS+11]. É utilizada a linguagem funcional Haskell em conjunto com um esquema de encriptação parcialmente homomórfica para construção de uma plataforma para execução segura de código, permitindo oferecer confidencialidade das informações. Este trabalho é semelhante ao CryptDB, na medida em que se supõe um modelo de segurança menos rígido para resolver um problema com um escopo bem determinado.

4.6. Considerações finais

Neste minicurso foram apresentados os recentes trabalhos de Craig Gentry, que resolveram um problema que permaneceu em aberto por 31 anos, que principalmente hoje em dia, com a consolidação do modelo de computação em nuvem, oferece uma solução elegante ao permitir a computação sobre dados encriptados.

A construção representa um avanço teórico, pela solução da conjectura feita por Rivest, Adleman e Dertouzos em 1978, reunindo uma variedade de conceitos matemáticos interessantes. É importante ressaltar que os esquemas propostos possuem demonstração de segurança, com base em problemas difíceis em reticulados, um assunto que ganhou novamente a atenção da comunidade científica por resistir à ataques quânticos, isto é, que fazem uso de computadores quânticos. Além disso, as construções ainda podem ser adaptadas de acordo com o problema a ser resolvido. Sendo assim, o material aqui exposto reuniu o estado da arte em encriptação homomórfica, mostrando uma série de trabalhos recentes, que representam um grande avanço para a criptografia moderna.

Como vimos, apesar de todas as otimizações propostas, a encriptação completamente homomórfica ainda é inviável para ser usada na prática. Existem resultados negativos [Bra12] que confrontam a capacidade homomórfica de um criptossistema com a eficiência do algoritmo de decriptação, estabelecendo assim um limite inferior para a computação da autoinicialização. Concretamente, se o esquema for capaz de avaliar homomorficamente a função de majoridade, então a decriptação não pode ser linear.

Existem alguns problemas em aberto, dentre os quais vale destacar: a construção de um esquema de ECH que não seja baseado na existência de ruído. Em especial, a multiplicação de textos encriptados resulta em um elemento com dimensão maior que os valores iniciais. Encontrar uma forma de multiplicar sem que isto ocorra é um problema interessante em aberto.

Uma outra linha de pesquisa possível é sobre o uso de encriptação parcialmente homomórfica. Foi mostrado que o esquema BGV pode ser adaptado para diferentes

profundidades multiplicativas, permitindo encontrar uma boa configuração para diversos problemas práticos. Além disso, também foram apresentados outros criptossistemas que podem ser utilizados em alguns cenários, como por exemplo com o uso do ElGamal no contexto de eleições eletrônicas no projeto VoteBox.

A tabela a seguir mostra a complexidade por operação homomórfica, após otimizações, para esquemas de encriptação homomórfica sobre inteiros e reticulados, além da recente proposta com base no problema RLWE. Diversos trabalhos estão surgindo propondo modificações ao esquema BGV, obtendo vantagem em determinados cenários [GHPS12, GHS12a].

Versão	Complexidade a cada operação homomórfica
Inteiros	$O(\lambda^5)$
Reticulado	$O(\lambda^{3.5})$
RLWE (BGV)	$\tilde{O}(\lambda)$

4.7. Exercícios

1. Considerando o esquema simétrico sobre inteiros, onde a encriptação é calculada por $c = m + 2r + pq$, se $|2r| < 50$, $p = 5001$ e $10001 \leq q \leq 20001$. Na pior das hipóteses, quantas somas podemos realizar homomorficamente? E com relação às multiplicações?

2. Modificando apenas o valor de p no exercício anterior, calculamos os textos encriptados $c = 79818018$ e $c' = 80616104$. Implemente um programa para encontrar o novo valor de p, sabendo que possui a mesma quantidade de dígitos decimais que o valor anterior. Descubra também as mensagens m e m', correspondentes a c e c', respectivamente.

3. O esquema das questões anteriores pode ser facilmente adaptado para permitir espaço de texto claro $\mathbb{F}_3$. Para isso, a encriptação é realizada por $c = m + 3r + pq$, além disso, é necessário que $|3r| < 50$. Se o restante das condições permanecerem iguais, é possível multiplicar homomorficamente?

4. Para que o espaço de texto claro seja $\mathbb{F}_3$ o algoritmo de decriptação computa $m = ((c \pmod p) \pmod 3)$. Altere o programa feito no exercício 2 para usar este espaço de texto claro e compute quais seriam os respectivos valores de m e m'.

5. Considerando $f(x) = x^2 - 1$ e o anel $\mathbb{Z}[x]/f(x)$, responda as seguintes questões:

   (a) Quais são as classes laterais do ideal (2)?

   (b) Compute a base de rotação B do reticulado gerado por $(a(x))$, onde $a(x) = x + 2$.

(c) O polinômio $p(x) = 15x + 12$ pertence ao reticulado gerado por $(a(x))$?

(d) Compute $x + 10 \pmod{B}$.

6. Seja $R = \mathbb{Z}[x]/f(x)$, onde $f(x) = x^2 - 1$. Considerando o ideal polinomial J gerado por $(a(x))$, onde $a(x) = 5001x + 10002$. Dado par de chaves (B_J^{sk}, B_J^{pk}) a seguir

$$\left(\begin{bmatrix} 10002 & 5001 \\ 5001 & 10002 \end{bmatrix}; \begin{bmatrix} 15003 & 5001000 \\ 0 & 10002 \end{bmatrix} \right)$$

e o texto encriptado $c = [-14980, 37]^T$, responda aos seguintes itens:

(a) Compute c^2, lembrando que c corresponde ao polinômio $c(x) = 37x - 14980$.

(b) Compute $c^2 \pmod{B_J^{pk}} \pmod 2$.

(c) O que é possível concluir a respeito de c?

7. No exercício anterior, a encriptação de uma mensagem m, interpretada como polinômio em $\mathbb{Z}[x]/(x^2 - 1)$, é calculada somando-se a m um polinômio da forma $2r_1x + 2r_0$, onde $|2r_i| < 50$, para $i \in \{0, 1\}$. Quantas multiplicações são permitidas por este esquema?

8. Implemente um programa para computar a chave secreta utilizada no exercício 6.

9. Considerando o esquema BGV sobre o anel $R_q = \mathbb{Z}_q[x]/(x^2 + 1)$, para $q = 5001$ e $N = 2$. Dada a chave pública

$$A = \begin{bmatrix} -1072x - 1604 & -1367x - 259 \\ -1310x + 326 & -521x + 811 \end{bmatrix},$$

a encriptação é computada por $c = m + A^T r$, onde r é um vetor coluna de polinômios em R_2, isto é, com coeficientes em $\{-1, 0, 1\}$. Dado o texto encriptado $c = [-2168x + 1693, -224 + 1916]^T$, responda aos seguintes itens:

(a) Compute c^2, lembrando que $c = [c_0, c_1]$ e pode ser interpretado como um polinômio $c(v) = c_0 + c_1 v$, onde v é uma variável simbólica nova. Assim, $c^2 = (c_0 + c_1 v)^2 = c_0^2 + 2c_0c_1v + c_1^2v^2$. Ou seja, para computar c^2, é preciso computar os coeficientes c_0^2, $2c_0c_1$ e c_1^2.

(b) Compute $\langle c^2, s \otimes s \rangle_q \pmod 2$.

(c) O que é possível concluir a respeito de c?

10. Sabendo que a chave pública do exercício anterior foi gerada usando erro e tal que $|2e| < 50$, quantas multiplicações são permitidas?

11. Implemente um programa para computar a chave secreta utilizada no exercício 9.

4.8. Material complementar

Para complementar este minicurso, foram reunidos exemplos, exercícios, textos e referências sobre encriptação homomórfica. Este material encontra-se disponível na web, no endereço `http://www.fhe.tecic.com.br`.

Referências

[Bab86] László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[BF11] Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In *EUROCRYPT*, pages 149–168, 2011.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In Joe Killian, editor, *Proceedings of Theory of Cryptography Conference 2005*, volume 3378 of *LNCS*, pages 325–342. Springer, 2005.

[BGV11] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. Fully homomorphic encryption without bootstrapping. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:111, 2011.

[BKW03] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50:506–519, July 2003.

[BMS+11] Alex Bain, John Mitchell, Rahul Sharma, Deian Stefan, and Joe Zimmerman. A Domain-Specific Language for Computing on Encrypted Data (Invited Talk). In Supratik Chakraborty and Amit Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2011)*, volume 13 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6–24, Dagstuhl, Germany, 2011. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Bra12] Zvika Brakerski. When homomorphism becomes a liability. Cryptology ePrint Archive, Report 2012/225, 2012. `http://eprint.iacr.org/`.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:109, 2011.

[CMNT11] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Fully homomorphic encryption over the integers with shorter public keys. In *Proceedings of the 31st annual conference on Advances in cryptology*, CRYPTO'11, pages 487–504, Berlin, Heidelberg, 2011. Springer-Verlag.

[CNT11] Jean-Sebastien Coron, David Naccache, and Mehdi Tibouchi. Optimization of fully homomorphic encryption. Cryptology ePrint Archive, Report 2011/440, 2011. `http://eprint.iacr.org/`.

[DF04] D.S. Dummit and R.M. Foote. *Abstract algebra*. Wiley, 2004.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DMMQN11] Nico Döttling, Thilo Mie, Jörn Müller-Quade, and Tobias Nilges. Basing obfuscation on simple tamper-proof hardware assumptions. Cryptology ePrint Archive, Report 2011/675, 2011. `http://eprint.iacr.org/`.

[FK94] M. Fellows and N. Koblitz. Combinatorial cryptosystems galore! In G. L. Mullen and P. J.-S. Shiue, editors, *Finite Fields: Theory, Applications, and Algorithms*, volume 168 of *Contemporary Mathematics*, pages 51–61. AMS, 1994.

[Gen09a] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 169–178, New York, NY, USA, 2009. ACM.

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In *CRYPTO*, pages 112–131, 1997.

[GH11a] Craig Gentry and Shai Halevi. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In *FOCS*, pages 107–109, 2011.

[GH11b] Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. In Kenneth Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148. Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-20465-4_9.

[GHPS12] Craig Gentry, Shai Halevi, Chris Peikert, and Nigel P. Smart. Ring switching in bgv-style homomorphic encryption. Cryptology ePrint Archive, Report 2012/240, 2012. `http://eprint.iacr.org/`.

[GHS12a] Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In *EUROCRYPT*, pages 465–482, 2012.

[GHS12b] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the aes circuit. Cryptology ePrint Archive, Report 2012/099, 2012. `http://eprint.iacr.org/`.

[GM82] S. Goldwasser and S. Micali. Probabilistic Encryption and How To Play Mental Poker Keeping Secret All Partial Information. In *Proc. 14th. ACM Symp. on Theory of Computing*, pages 270–299. ACM, 1982.

[HG01] Nick Howgrave-Graham. Approximate integer common divisors. In *CaLC*, pages 51–66, 2001.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Advances in Cryptology EUROCRYPT 2010*, 6110/2010(015848):1?23, 2010.

[MG09] Peter Mell and Tim Grance. The nist definition of cloud computing. *National Institute of Standards and Technology*, 53(6):50, 2009.

[NLV11a] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, CCSW '11, pages 113–124, New York, NY, USA, 2011. ACM.

[NLV11b] Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, CCSW '11, pages 113–124, New York, NY, USA, 2011. ACM.

[PRZB11] Raluca Ada Popa, Catherine M. S. Redfield, Nickolai Zeldovich, and Hari Balakrishnan. Cryptdb: protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 85–100, New York, NY, USA, 2011. ACM.

[RAD78] R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms, in r. a. demillo et al. In *Eds.), Foundations of Secure Computation*, pages 169–179. Academic Press, 1978.

[Rap06] Doerte K. Rappe. Homomorphic cryptosystems and their applications. Cryptology ePrint Archive, Report 2006/001, 2006. `http://eprint.iacr.org/`.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.

[RSA83] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 26:96–99, January 1983.

[Sch11] Patrick Schmidt. Fully homomorphic encryption: Overview and cryptanalysis. Diploma thesis, TU Darmstadt, Jul 2011.

[SDW08] Daniel Sandler, Kyle Derr, and Dan S. Wallach. Votebox: A tamper-evident, verifiable electronic voting system. In *USENIX Security Symposium*, pages 349–364, 2008.

[SS10] Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In *ASIACRYPT*, pages 377–394, 2010.

[ST98] Tomas Sander and Christian F. Tschudin. Protecting mobile agents against malicious hosts. In *Mobile Agents and Security*, pages 44–60, 1998.

[SV09] N.P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. Cryptology ePrint Archive, Report 2009/571, 2009. `http://eprint.iacr.org/`.

[SV11] N.P. Smart and F. Vercauteren. Fully homomorphic simd operations. Cryptology ePrint Archive, Report 2011/133, 2011. `http://eprint.iacr.org/`.

[vDGHV09] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. Cryptology ePrint Archive, Report 2009/616, 2009. `http://eprint.iacr.org/`.

