Análise de variantes do criptossistema McEliece e da versão candidata a padrão de encriptação baseada em códigos

Guilherme G. Martinez[1], **Denise Hideko Goya**[1]

[1]Centro de Matemática, Computação e Cognição - Universidade Federal do ABC (UFABC)
Santo André, São Paulo, Brasil

{martinez.guilherme}@outlook.com.br, denise.goya@ufabc.edu.br

Abstract. *This work discusses the security of some variants of the McEliece cryptosystem. Therefore, works that, from structural and/or parametric changes, sought to make it more efficient are analyzed. It is observed that, among the variants evaluated, those based on Goppa Code maintained the same level of security and achieved the proposed improvements, in part with the change of parameters and optimizations through list decoding; the others, based on less conservative structures, were broken, but inspire other forms of optimization.*

Resumo. *Este trabalho discorre sobre a segurança de algumas variantes do criptossistema McEliece. Para tanto, são analisados trabalhos que, a partir de mudanças estruturais e/ou paramétricas, buscaram torná-lo mais eficiente. Observa-se que, dentre as variantes avaliadas, as baseadas em Código Goppa mantiveram o mesmo nível de segurança e alcançaram as melhorias propostas, em parte com a mudança de parâmetros e otimizações através de list decoding; as demais, baseadas em estruturas menos conservadoras, foram quebradas, porém inspiram outras formas de otimização.*

1. Introdução

Com o advento da tecnologia, diversos novos dispositivos passaram a estar conectados à Internet. Junto com isso, aumentaram a quantidade e a sensibilidade das informações processadas, armazenadas e transmitidas por esses dispositivos. Neste cenário, criptografia, sendo a comunicação na presença de adversários [Van Leeuwen and Leeuwen 1990] tem um papel fundamental em garantir a transmissão segura destas informações, seja por meio da promoção do sigilo, da autenticação e integridade de mensagens ou autenticidade de entidades via assinatura digital [Zhang et al. 2011].

Sistemas criptográficos do tipo assimétrico utilizam funções que são facilmente computáveis em uma direção, mas que são difíceis de calcular a inversa sem o conhecimento de um segredo. O RSA, por exemplo, é um sistema criptográfico dessa classe [Goldwasser and Bellare 1996]. Nele, para encriptar uma mensagem, basta realizar uma potenciação modular, porém, para decriptar essa mensagem, é necessário conhecer a chave secreta ou ser capaz de efetuar a fatoração do produto de dois números primos grandes selecionados previamente [Yan 2009]. No modelo computacional clássico, acredita-se que não há solução de tempo polinomial para o problema da fatoração e isso é crucial para a segurança do RSA.

Contudo, em [Shor 1999] apresenta-se um algoritmo quântico que possibilita fatorar inteiros e calcular logaritmos discretos em tempo polinomial. Portanto, criptossistemas que baseiam sua segurança nesses problemas computacionais estão vulneráveis à

construção de um computador quântico com capacidade para executar o algoritmo de Shor [Barreto et al. 2013]. Dessa forma, qualquer mensagem trocada usando tais sistemas poderá vir a ser decifrada em algum momento. Felizmente, há criptossistemas que aparentam ser resistentes ao ataque de Shor. Dentre eles se destaca o criptossistema McEliece, baseado em códigos corretores de erros [McEliece 1978].

O McEliece original é relativamente pouco eficiente e há diversos esforços que visam sua otimização para torná-lo mais atrativo [Berger et al. 2009, Bernstein et al. 2008, Baldi et al. 2016]. Citam-se duas das principais abordagens: redução do tamanho das chaves pública e privada do sistema; e redução do *overhead* do espaço utilizado por uma mensagem após ser encriptada [Barreto et al. 2013]. Porém algumas variantes desenvolvidas com o objetivo supracitado falharam.

Por exemplo, em [Otmani et al. 2010] é realizada a criptoanálise de duas dessas variantes. Ambas buscam reduzir o tamanhos das chaves públicas do algoritmo original, contudo não foram capazes de manter o mesmo nível de segurança da implementação original, conforme ataques demonstrados. Em [Faugere et al. 2010], apresenta-se a análise de outras variantes do McEliece que também tiveram sua segurança comprometida (algumas, com nível de segurança de 256-bits, tiveram suas chaves privadas expostas em até 0,06 segundos).

O objetivo deste trabalho é apresentar uma análise de algumas variantes do criptossistema McEliece, incluindo a variante candidata a padrão de encriptação baseada em códigos. Como trabalho relacionado, pode-se citar o de [Engelbert et al. 2007] que se diferencia da presente proposta por apresentar apenas ataques feitos à implementação de [McEliece 1978] e [Niederreiter 1986]. O trabalho de [Zajac 2014] também discorre aspectos gerais de ataques ao McEliece e a algumas variantes com, por exemplo, resistência a *CCA2 (adaptive chosen-ciphertext attack)*, mas não as compara entre si.

O restante deste trabalho está organizado como segue: na Seção 2, são apresentados conceitos em criptografia baseada em códigos corretores de erros; na Seção 3, são apresentadas algumas variantes do sistema McEliece; na Seção 4, há uma discussão sobre as análises, seguida de considerações finais.

2. Fundamentação Teórica

A criptografia de chave pública (também conhecida como criptografia assimétrica), permite que sejam trocadas mensagens com confidencialidade entre dois ou mais indivíduos por meio de um canal não seguro e sem o compartilhamento prévio da chave privada [Buchmann 2013]. Isso é possível devido à sua principal característica: a chave de encriptação é diferente da chave de decriptação, i.e., a chave usada para codificar a mensagem é diferente da chave usada para revelar a mensagem original. Em sistemas simétricos, ambas as chaves são iguais [Yan 2009].

2.1. Códigos corretores de erros

Nesta subseção, apresentaremos alguns conceitos básicos sobre códigos corretores de erros para, na subseção seguinte, descrevermos como tais fundamentos podem ser aplicados em algoritmos de encriptação.

Frequentemente, ao transmitirmos mensagens por canais ruidosos, e.g., enviar arquivos através da Internet, realizar ligações de celular, etc., partes dos dados podem ser

alteradas até chegarem ao seu destinatário. A alteração pode ser tamanha a ponto de ser inviável compreender seu conteúdo. Por exemplo, uma mensagem simples como "oi", pode ser transformada em uma sequência aleatória de caracteres. Para contornar tais situações, diversas técnicas de detecção e correção de erros foram desenvolvidas [Kurose et al. 2007]. Dentre elas há, por exemplo, a adição de dígitos verificadores em documentos de identidade, bits de paridade em sequências binárias, além de outras informações redundantes para não apenas detectar erros como corrigi-los.

A Figura 1 ilustra essa situação em que uma mensagem *m* é codificada e transmitida através de uma canal com ruídos. E então, durante a decodificação, seu significado original é inferido a partir de algumas técnicas de recuperação de erros, resultando em *m'* que pode ou não ser igual a mensagem *m* original (dependendo da capacidade de correção). Por exemplo, suponha que só possamos transmitir a palavra "oi" e a palavra "tchau", que, após serem codificadas, resultam respectivamente, nas seguintes cadeias binarias: 000 e 111. Seja "oi" a mensagem que queremos transmitir e que após ser codificada e enviada pelo canal ruidoso, tem um erro adicionado, resultando na cadeia 001. O receptor, nesse caso, perceberá com facilidade que um erro ocorreu pois a cadeia 001 não está definida. Uma possível estratégia nesse caso é aproximar, por tentativa e erro, a palavra recebida ao código mais próximo conhecido (no caso, como a quantidade de palavras para este código é pequena é fácil verificar que é 000 = "oi"). Considerando-se que a *distância Hamming* é definida pelo número de posições que dois vetores diferem, basta identificar o código válido cuja *distância Hamming* em relação ao código recebido é a menor. Define-se *Distância mínima* de um código como a menor distância entre todas as palavras daquele código.

Um passo muito importante para decodificar uma mensagem com ruído é ser capaz de detectar e corrigir os erros da mensagem; para tanto há técnicas mais sofisticadas do que apenas estimar a palavra válida mais próxima por tentativa e erro, que pode ser um processo custoso caso haja um grande número de palavras possíveis para o código. Para explicar uma dessas técnicas, introduziremos o conceito da *matriz geradora* e o de *matriz de paridade*. Por meio da primeira, que chamaremos de G, é possível gerar qualquer palavra pertencente a um código através do produto entre ela e a mensagem que se deseja codificar. Com a finalidade de verificar se uma dada palavra pertence ao código faz-se uso da *matriz de paridade H*. Esse processo consiste em multiplicar *H* pela transposta da palavra codificada que se quer verificar. O resultado desta operação é um vetor denominado *síndrome*. Caso este seja o vetor nulo então a palavra pertence ao código, caso contrário é possível identificar o erro introduzido a partir deste valor, que indica o vetor de erros adicionado (dado que o código tenha capacidade para corrigir a quantidade de erros inseridos). Basta então removê-lo e então obter a mensagem original [Hill 1986].

2.2. Encriptação baseada em códigos corretores de erros

Dada a dificuldade de recuperar as mensagens após a ocorrência de um certo número de erros, [McEliece 1978] propôs inserir, propositalmente, erros como uma forma de encriptar as mensagens. Se apenas o destinatário tiver a capacidade de recuperar a mensagem original, por meio de correção dos erros e com base em um conhecimento privilegiado (a chave secreta), formula-se a ideia de Encriptação baseada em códigos corretores de erros. Ou seja, a primitiva algorítmica de criptossistemas

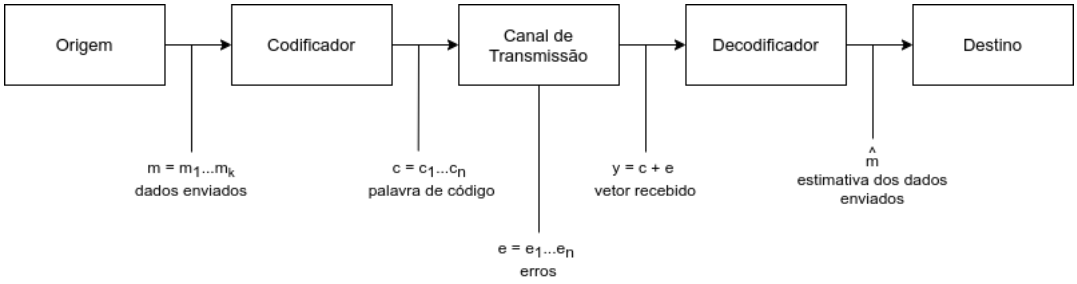

Figura 1. Correção de erros ocorridos durante transmissão de dados com base em códigos corretores de erro [Barreto et al. 2013].

dessa categoria faz uso de um código corretor de erros. A primitiva pode consistir tanto em adicionar erros a uma palavra pertencente ao código (como já mencionado), como também em computar a síndrome relativa à matriz de paridade do código usado. [Overbeck and Sendrier 2009, Repka and Cayrel 2014, Barreto et al. 2013].

Dentre os criptossistemas cuja encriptação é baseada em códigos, podem ser citados: McEliece [McEliece 1978], alvo desta pesquisa, e Niederreiter [Niederreiter 1986], que consiste em uma variante do criptossistema proposto por McEliece. Essa variante possui tamanho de chave pública inferior e é equivalente ao criptossistema criado por McEliece quando utilizadas determinadas configurações, conforme demonstrado em [Li et al. 1994].

2.3. Códigos cíclicos

Um código linear C sobre o corpo finito $GF(q)$ é um *código cíclico* se, para qualquer palavra $c = (c_0, c_1, c_2, ..., c_{n-1})$ pertencente ao código C, então $c' = (c_{n-1}, c_0, c_1, c_2, ..., c_{n-2})$ também é uma palavra pertencente à C. Isto é, para obter uma palavra do código, basta rotacionar uma palavra existente para a direita [Blahut 2003]. Cada vetor pertencente à $GF(q)$ pode ser representado através de um polinômio de grau $\leq n - 1$, tal que n é o comprimento do vetor. Cada componente do vetor pode ser identificada pelo coeficiente do polinômio. Por exemplo, o vetor binário $a = (1, 1, 0, 0, 1)$ pode ser representado pelo polinômio $1x^4 + 1x^3 + 0x^2 + 0x^1 + 1x^0 = 1x^4 + 1x^3 + 1$.

2.4. Criptossistema McEliece

O criptossistema McEliece é, como já mencionado, resistente ao ataque de Shor, além de ter uma baixa complexidade algorítmica ($O(n^2)$ contra $O(n^3)$ do RSA sobre um código de tamanho n) para encriptar e decriptar mensagens quando comparado com os demais criptossistemas [Barreto et al. 2013]. A seguir, uma descrição do funcionamento deste sistema:

Geração de chaves: Primeiramente, é escolhido um código linear (n, k) $\mathcal{C}$ capaz de corrigir t erros. Frequentemente são utilizados códigos binários Goppa [Goppa 1970], que foi o código utilizado na descrição original do sistema [McEliece 1978]. Em seguida, é gerada G, a matriz geradora $k \times n$ do código $\mathcal{C}$ (que pode ou não estar na forma canônica). Então, com a finalidade de "embaralhar" G, são geradas as matrizes aleatórias P e S. Onde P é uma matriz de permutação $n \times n$ e S é uma matriz não-singular (inversível) $k \times k$. Por fim, é gerada a chave de encriptação pública G', definida como

$$G' = SGP$$

Encriptação: Para encriptar os dados, é necessário dividi-los em blocos de k-bits. Cada bloco u será multiplicado pela chave de encriptação pública G' e terá um vetor aleatório de erros z de comprimento *n* e *peso t* adicionado, onde o *peso* é a quantidade de entradas não nulas do vetor z.

$$u' = uG' + z$$

Após isso, a mensagem está pronta para ser enviada.

Decriptação: Com a mensagem encriptada u' em posse do destinatário, é iniciada a decriptação. Este processo envolve realizar o produto entre u' e a inversa matriz de permutação, P^{-1}. Então, é aplicado o código $\mathcal{C}$ para remover o vetor de erros do resultado obtido anteriormente. O resultado dessa operação será uma palavra pertencente à $\mathcal{C}$. Por fim, basta multiplicar essa palavra código pela inversa da matriz geradora G^{-1} e em seguida pela inversa de S, obtendo-se assim u. Dessa forma, fica evidente a necessidade de manter as matrizes P, G e S secretas. A expressão abaixo explicita tais operações, onde z' representa o vetor de erros permutado.

$$u'P^{-1} = (uG' + z)P^{-1} = uSG + zP^{-1} = uSG + z' \xrightarrow[\text{remoção erro}]{\text{decodificação}} ((uSG)G^{-1})S^{-1} = u$$

3. Algumas variantes do McEliece

Nesta seção, são apresentadas algumas das variantes do McEliece que buscaram realizar melhorias no algoritmo original para torná-lo mais competitivo, por exemplo reduzindo o tamanho de sua chave, mas que ao fazê-lo podem ter aberto brechas de segurança.

3.1. Criptossistema McEliece baseado em código cíclico com matriz de paridade esparsa

Os autores de [Baldi and Chiaraluce 2007] propõem um criptossistema que faz uso de matrizes de paridade esparsas e código cíclico com o intuito de reduzir o tamanho da chave do algoritmo original proposto por [McEliece 1978]. Nesta variante, a matriz de paridade é definida como $H = (H_0...H_{no})$, onde cada matriz H_{ji} é uma matriz esparsa circulante de tamanho $p \times p$. A técnica *Random Difference Families - RDF* é usada para gerar cada entrada da matriz H, que garante a ausência de ciclos de tamanho 4. Há, também, uma matriz de *embaralhamento* $S_{k\times k}$ aleatória e não-singular e uma matriz esparsa de *transformação* $Q_{n\times n}$ também não-singular. Uma matriz geradora G reduzida na sua forma escalonada por linhas também é criada. Essas quatro matrizes se constituem na chave privada. A chave pública G' é dada pela seguinte expressão: $G' = S^{-1}GQ^{-1}$. Este processo é mostrado na Figura 2.

A encriptação de uma mensagem é feita de maneira semelhante à já apresentada. A versão encriptada de uma mensagem u é dada pela seguinte expressão $u' = uG' + e$ em que e é um vetor de erro de tamanho n e *peso* $t' <= t/m$, onde o valor de t é estimado através de simulações numéricas e m é o *peso* das colunas das matrizes S e Q. Para decodificar u', inicialmente computa-se $u'Q = (uS^{-1}GQ^{-1})Q + eQ = uS^{-1}G + eQ$. O erro eQ é então corrigido, o que é viável já que este tem *peso* máximo = t', resultando $uS^{-1}G$. Por fim, $u = ((uS^{-1}G)G^{-1})S$.

O criptossistema foi baseado em uma variante do McEliece que faz uso de Matrizes de paridade esparsas [Monico et al. 2000], mas este provou-se não resistente

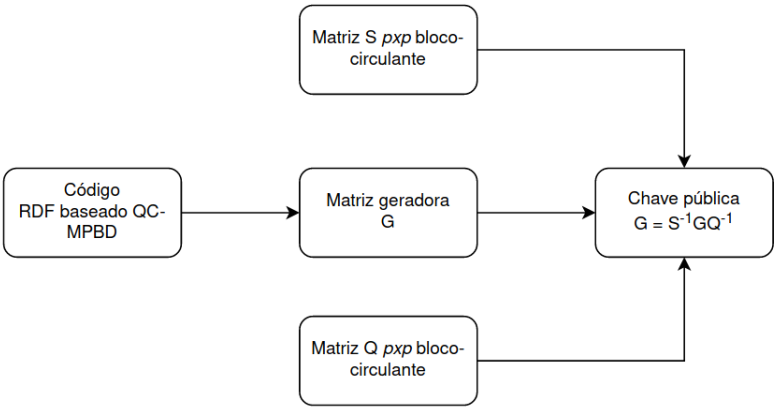

Figura 2. Geração chave pública do código cíclico com matriz de paridade de baixa densidade

a ataques baseados em palavras de baixo peso. Com intuito de corrigir essa falha, [Baldi and Chiaraluce 2007] propuseram alterações que visaram deixá-lo seguro contra tais ataques, além de ataques de força bruta, ataques baseados na decodificação do conjunto de informações (*information set deconding*), ataques de reenvio de mensagem e de mensagens relacionadas e ataques focados no código *Dual*.

Embora inicialmente o criptossistema tenha se mostrado resistente aos ataques mencionados acima, [Otmani et al. 2008] demonstram que o sistema poderia ser quebrado em até 3 minutos explorando a natureza quasi-cíclica inerente a ele. Para tanto, na primeira parte do trabalho, são buscados polinômios de baixo *peso*, divisores do polinômio público, cujo produto é a chave pública. Este é obtido através da chave pública, lembrando que por ser uma matriz quasi-cíclica pode ser representada através de um polinômio. Para realizar esta busca, duas estratégias são empregadas, ambas utilizam-se do fato de que a chave pública contém, com alta probabilidade, fragmentos deslocados do polinômio $q(x)$ que descreve a matriz Q. A primeira estratégia é feita com complexidade $O(\binom{m^2}{m}.p^2)$ enquanto que a segunda tem complexidade $O(p^3)$, onde p é o grau do polinômio menos 1 e m é o *peso* da linha e coluna da matriz Q. Por fim, com aproximadamente 2^{37} operações, é possível recuperar a matriz de paridade do código cíclico com matriz de paridade esparsa buscando por palavras de baixo *peso* do código.

3.2. Criptossistema McEliece baseado em código cíclico com matrizes de paridade de baixa e média densidade

Em [Moufek et al. 2016] é proposto o uso em conjunto de matrizes de paridade de baixa e média densidade com o objetivo de se obter uma chave menor que a implementação original proposta por [McEliece 1978]. A matriz de paridade de média densidade (MPMD) com *peso* por linha igual a $O(\sqrt{n})$ foi introduzida para buscar contornar os problemas de segurança enfrentados pelas matrizes de paridade de baixa densidade (MPBD) com *peso* por linha $O(1)$, além de utilizar um gerador diferente para obter uma estrutura mais aleatória a fim de evitar ataques estruturais para um código (n, k, w) onde w é o *peso* por linha da matriz. A seguinte notação é utilizada para representar, respectivamente, a

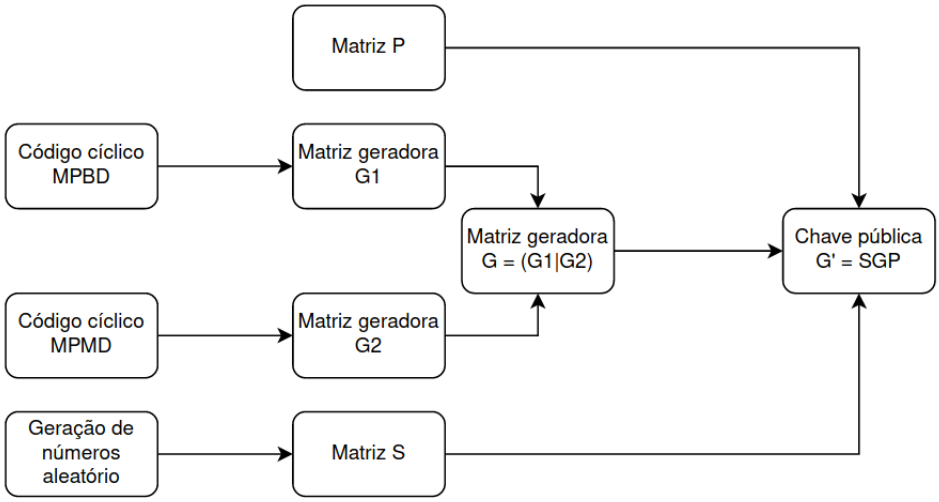

Figura 3. Geração chave pública do código cíclico com matrizes de paridade de baixa e média densidade

MPBD e a MPMD, (n_1, k, w_{mpbd}) e (n_2, k, w_{mpmd}).

O sistema proposto é composto por três algoritmos. O primeiro consiste no algoritmo usado para gerar a chave privada de decriptação *cpd* e a sua chave pública de encriptação *cpe*. Primeiramente, é escolhida uma matriz geradora G_1 para o (n_1, k, w_{mpbd}) código MPBD e uma matriz geradora G_2 para o (n_2, k, w_{mpmd}) código MPMD, com capacidade para corrigir, respectivamente, t_1 e t_2 erros. Então, é escolhida uma matriz de embaralhamento $S_{k\times k}$ não-singular aleatória e uma matriz de permutação $P_{n\times n}$, onde $n = n_1 + n_2$. Então, a chave pública G' é dada por $G' = SGP$ e a chave privada por (G, S, P) onde $G = (G_1|G_2)$. Este processo pode ser visto na Figura 3. Para encriptar uma mensagem m, basta gerar, de maneira aleatória, um vetor e de *peso* máximo t_1 nas primeiras n_1 posições e de *peso* máximo t_2 nas últimas n_2 posições. E, em seguida, computar $m' = mG' + e$. Já para decriptar a mensagem, basta computar $u = m'P^{-1} = mSG + eP^{-1}$, determinar $z = mS$ através da correção dos erros em u utilizando uma modificação do algoritmo de decodificação Gallager e, por fim, obter a mensagem m computando $m = zS^{-1}$.

O criptossistema descrito acima foi mostrado como sendo inseguro em [Dragoi and Kalachi 2017]. A insegurança advém do uso da matriz de baixa densidade, que é suscetível a ataques decodificação do conjunto de informações (*information set decoding*), de maneira semelhante ao criptossistema citado anteriormente também baseado em matrizes de paridade de baixa densidade. Embora, inicialmente, tenha se mostrado seguro sob esse ataque de acordo com os autores, estes falharam em considerar a possibilidade de tomar vantagem da estrutura do código, em especial, da matriz de paridade de baixa densidade, usada para gerar G_1 na Figura 3.

O ataque consiste na recuperação da chave, para tanto é mostrado que há evidências de que um número suficiente de palavras pertencentes ao código com peso w_{mpbd} estão na chave pública. Portanto, o ataque se inicia com a busca das palavras do código com *peso* w_{mpbd} partindo do pressuposto que o atacante tem conhecimento do valor desse parâmetro. Através da aplicação de algoritmos semelhantes ao de Dumer ou ou-

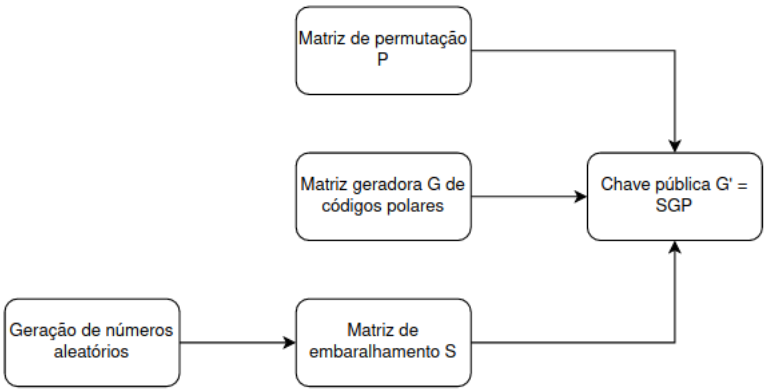

Figura 4. Geração chave pública do criptossistema baseado em códigos polares

tras variantes *ISD (Information Set Decoding)*, é possível construir uma matriz $H^*_{k^*,n_1+n_2}$ que é capaz de gerar o mesmo código que $(H_{mpbd}|0)P$ e, sendo assim, possui n_2 colunas iguais a zero, onde $k^* = n_1 - k$ e H_{mpbd} é a matriz de paridade do código MPBD. Em seguida, é demonstrado que é possível gerar uma matriz de permutação alternativa P^* que pode ser computada com complexidade $O(n)$, através da identificação das n_2 colunas com valores zero da matriz H^*.

Dessa forma, é obtido um código MPBD equivalente capaz de corrigir o mesmo número de erros que o secreto, podendo, com alta probabilidade, decodificar um texto criptografado, com a complexidade sendo, no pior caso, dominada pelo custo do algoritmo *ISD* utilizado. Vale ressaltar que o ataque funciona independente da estrutura do segundo código utilizado, sendo baseado exclusivamente em identificar a estrutura do código MPBD. Portanto, a maior fraqueza, neste caso, para este sistema advém do uso da matriz de paridade de baixa densidade.

3.3. Criptossistema McEliece baseado em códigos polares

Os autores de [Shrestha and Kim 2014] propõem, como alternativa ao código Goppa utilizado na proposta original do criptossistema McEliece, o uso de códigos polares, com a finalidade de obter um criptossistema mais atrativo, assintoticamente mais eficiente com o aumento das palavras do código. O construção do algoritmo é mostrada a seguir. Primeiramente, é criada uma matriz geradora de códigos polares de tamanho $G_{k\times n}$, onde as linhas pertencentes a *canais ruins* são descartadas. Em seguida, semelhante às demais variantes, duas matrizes são geradas: matriz de permutação $P_{n\times n}$ e uma matriz de *embaralhamento* $S_{k\times k}$ não singular, construída com números gerados aleatoriamente. A chave pública G' é então dada por $G' = SGP$. Vide a Figura 4.

O processo de encriptação é dado pelo produto da mensagem m com a chave pública G', mais a adição de um vetor de erros e, portanto temos: $m' = mG' + e = mSGP + e$, onde o vetor de erros afeta principalmente os *canais ruins* devido à transformação de canal. Já o processo de decriptação é feito da seguinte maneira: primeiro é computado $y = m' * P^{-1}$ para remover a matriz de permutação da mensagem cifrada. Então é aplicado o decodificador de cancelamento sucessivo (*successive cancellation decoder*) para decodificar y, removendo os erros inseridos, gerando $y' = mS$. Por fim, a mensagem m original pode ser obtida por $m = y'S^{-1} = mSS^{-1}$. Este processo

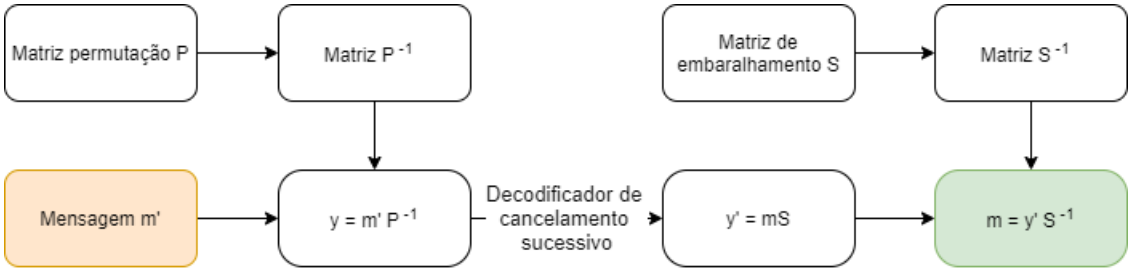

Figura 5. Encriptação e decodificação de mensagem do criptossistema baseado em códigos polares

pode ser observado na Figura 5.

Este criptossistema foi considerado inicialmente seguro por, após análise de seus criadores, ter resistido a ataques em que o atacante busca adivinhar as matrizes S e P de maneira aleatória e tenta aplicar o decodificador de cancelamento sucessivo. Em seguida, é mostrado que, embora tenham as suas similaridades, a variante proposta utilizando códigos polares se difere suficientemente de variantes baseadas em códigos Reed-Muller para que seja resistente ao ataque *Sidelnikov* [Minder and Shokrollahi 2007].

Porém, a criptoanálise desse criptossistema feita em [Bardet et al. 2016] mostrou que não pode ser considerado seguro. Para tanto, é resolvido o problema de equivalência de códigos para recuperar o código polar utilizado, fazendo-se uso do fato de que códigos polares contêm palavras de baixo peso que podem ser recuperadas com algoritmos padrão de busca. Parte do ataque feito utiliza parte da estrutura de ataques de sucesso propostos contra a variante do Criptossistema McEliece baseada em códigos Reed-Muller.

O problema de equivalência de códigos para essa variante pode ser considerado uma instância difícil do Algoritmo de divisão de suporte (*Support Splitting Algorithm*), responsável por indicar se dois códigos são permutacionalmente equivalentes, i.e., se um pode ser obtido através da permutação das palavras do outro [Sendrier 2000]. Apesar disso, foi mostrado ser possível resolver este problema de maneira relativamente eficiente para os parâmetros propostos no trabalho original.

3.4. Variantes com ajustes de parâmetros

Nesta subseção, são apresentadas variantes que, em vez de realizarem grandes alterações estruturais do McEliece original, se concentram em ajustes de parâmetros visando melhorar a competitividade e segurança do sistema.

Com o aumento significativo da capacidade de processamento de *CPUs* nos últimos anos, assim como uma sofisticação maior nos ataques empregados, os parâmetros originais propostos junto com o criptossistema McEliece começaram a apresentar sinais de defasagem. Um exemplo disso, está em [Bernstein et al. 2008], que demonstrou em 2008, que utilizando um único processador, com apenas um núcleo, seria possível quebrar o criptossistema original em apenas 1400 dias. Tempo esse que poderia ser reduzido para apenas 7 utilizando um *cluster* de aproximadamente 200 computadores.

Visando sanar tal problema, são propostos novos conjuntos de parâmetros para garantir segurança de 80, 128 e 256-bit, ao mesmo tempo que tentam minimizar o tamanho da chave pública, como as variantes anteriores também tentaram. Além disso,

também é evidenciado que o criptossistema original não é resistente a *CCA2*. Isso porque uma mesma mensagem pode ter mais de uma cifra diferente, dessa forma, a encriptação de uma mesma mensagem, várias vezes, pode ser usada para revelar partes da mensagem original ou da chave. Após realizar essa correção, a chave pública poderá ser colocada em forma sistemática, isto é, [Matriz identidade, matriz de paridade]. Dessa forma, o número de bits necessários podem ser reduzidos de $k * n$, para $k * (n - k)$.

Como sugestão, os autores de [Bernstein et al. 2008], propõem o aumento de n, a utilização de decodificação de listas (do inglês, *list decoding*), que permite a correção de mais erros, ou seja, aumento do nível de segurança, mantendo a chave pública do mesmo tamanho. Já os parâmetros propostos, todos com taxas de código de 0,75, consistem em:

- Segurança 80-bit: [1632, 1269] códigos Goppa, com t=33 e 34 erros;
- Segurança 128-bit: [2960, 2288] códigos Goppa, com t=56 e 57 erros;
- Segurança 256-bit: [6624, 5129] códigos Goppa, com t=115 e 117 erros.

Em [Canteaut and Sendrier 1998] é apresentado um novo ataque ao criptossistema McEliece original, com os parâmetros propostos em [McEliece 1978], i.e., códigos Goppa irredutíveis de tamanho 1024, dimensão 524 e 101 de distância mínima. Para tanto, é proposto um novo algoritmo para encontrar palavras de peso mínimo no código. Este ataque funciona também para outros código lineares. Esse trabalho aponta, assim como o trabalho anterior, que o criptossistema como ele foi originalmente proposto, é suscetível a *CCA (chosen-ciphertext attack)*. Com o ataque proposto por [Canteaut and Sendrier 1998], a decriptação de uma única mensagem cifrada de 10.000 leva aproximadamente 2 meses e 14 dias. Porém, esse trabalho serviu de base para o ataque proposto por [Bernstein et al. 2008], assim como também para as melhorias sugeridas ao criptossistema original.

No trabalho de [Bernstein et al. 2010] é apresentada uma nova variante com o uso da já mencionada decodificação de lista para reduzir o tamanho da chave pública, em conjunto com a utilização de *Wild Goppa Codes* (em tradução literal, *Códigos Goppa Selvagens*). Estes são códigos em subcorpos sobre um campo F_q pequeno que tem a capacidade de correção de erros aumentada por um fator de $q/(q-1)$. Para o McEliece original, $q = 2$. Como sugestão, o autor propõe, como alternativa ao original, o uso de q diferente de 2 e $3 \leq q \leq 32$, além da decodificação de listas. Apenas com o uso de códigos Goppa na forma $\tau_q(a_1, ..., a_n, g^{q-1})$, é possível a utilização a adição de erros de *peso* $\lfloor q * t/2 \rfloor$ sem alterar a dimensão do código. A implementação de *Códigos Goppa selvagens* aliada com a decodificação de listas aumenta a correção de erros para $n - \sqrt{n * (n - qt)} > \lfloor q * t/2 \rfloor$.

3.5. Classic McEliece

No final de 2016, o NIST (National Institute of Standards and Technology) dos Estados Unidos iniciou um processo em busca de algoritmos de chave pública seguros simultaneamente nos modelos computacionais clássico e quântico [NIST 2017].

Com o objetivo de cumprir com essa requisição, [Albrecht et al. 2020] propõem uma variante do criptossistema McEliece original denominada *Classic McEliece* para construção de um *KEM* (*Key encapsulation mechanisms*, em português, Mecanismo de encapsulamento de chave), em outras palavras, o objetivo é a troca de chaves simétricas

com proteção a *CCA2*. O enfoque dessa variante está, novamente, na eficiência do algoritmo de forma que seja mantida ou aumentada a segurança do criptossistema. Tomando como base diversos dos artigos apresentados neste trabalho, o *Classic McEliece* utiliza a chave pública em sua forma sistemática, para reduzir o tamanho da chave de kn para $k(n-k)$ bits.

A encriptação da mensagem agora é dada pela expressão $m' = H * m$, onde m é a mensagem gerada, de tamanho n e *peso* t e H é a chave pública gerada anteriormente. Essa implementação é semelhante à variante proposta por [Niederreiter 1986], porém utilizando Códigos Goppa. Dessa forma, a mensagem cifrada m' é uma síndrome do código e, portanto, também tem o tamanho reduzido se comparado à implementação proposta em [McEliece 1978], ficando com tamanho final de $n-k$ bits, ou aproximadamente 256 *bytes*, com uma taxa k/n de cerca de $0,8$.

Segundo seus autores, *Classic McEliece* se mostrou, até este momento, resistente a ataques como *Information-set Decoding*, recuperação de chave, i.e., recuperar o conjunto de elementos e o polinômio utilizado no Código Goppa, e é seguro contra ataques *CCA*. Entretanto, os autores ressaltam que é preciso implementar com cautela para evitar ataques do tipo *side-channel*, em que o atacante faz, por exemplo, uma análise do tempo de execução do algoritmo conforme a entrada e a utiliza para facilitar outros tipos de ataques. Este problema acomete principalmente algoritmos que não possuem tempo de execução constante.

O algoritmo se encontra atualmente na terceira rodada e é finalista da competição do *NIST* e são propostas diversas combinações de parâmetros que podem ser lidas em [Albrecht et al. 2020].

4. Análise de resultados

Considerando os trabalhos apresentados, pode-se ver que as variantes que mantiveram códigos Goppa em sua implementação conseguiram manter o nível de segurança da implementação original, porém ao custo de otimização, ainda que tenham conseguido mostrar melhorias interessantes em relação ao que foi proposto por [McEliece 1978].

Na Tabela 1, é exibido o resultado das análise de algumas das variantes apresentadas neste trabalho. Nela também foi incluído o algoritmo original para efeito de comparação. Vale ressaltar que a complexidade do criptossistema baseado em códigos polares proposto por [Shrestha and Kim 2014] não está completa, já que conforme exposto no trabalho original, esse passo é repetido no passo indutivo da criptoanálise de acordo com um termo probabilístico e representa até 20% da computação total necessária. Para os demais criptossistemas, salvo melhor juízo, não foram apresentadas até o presente momento, criptoanálises capazes de quebrarem o algoritmo original sem comprometer também o criptossistema proposto por [McEliece 1978]. A complexidade do ataque proposto por [Both and May 2018] a códigos lineares consegue reduzir o coeficiente em relação a propostas anteriores, entretanto, o ataque permanece exponencial em função do tamanho n do código, i.e., $O(2^{0,0885n})$. Portanto, tanto [McEliece 1978] quanto [Albrecht et al. 2020] permanecem com complexidade de criptoanálise exponenciais. O número de operações binárias considerada na coluna *Busca palavras baixo peso* leva em conta os parâmetros propostos pelos autores dos trabalhos citados.

A implementação baseada em códigos Quasi-cliclícos com matrizes de paridade

Tabela 1. Comparação de variantes do Criptossistema McEliece

Variante	**Complexidade criptoanálise**	**Busca palavras baixo peso**	**Taxa de código aprox.**
McEliece Original [McEliece 1978]	$O(2^{0,0885n})$	–	$0,5$
QC-MPBD [Baldi and Chiaraluce 2007]	$O(n^3)$	2^{37}	$0,75$
QC-MPBD e QC-MPMD [Moufek et al. 2016]	$O(e^{-w_{mpbd}ln(k/(n_1+n_2))(1+O(1))})$	2^{19}	$0,75$
Polar [Shrestha and Kim 2014]	$O(e^{-wln(k/n)(1+O(1))})*$	–	$0,30$
Classic McEliece [Albrecht et al. 2020]	$O(2^{0,0885n})$	–	$0,80$

de baixa densidade se mostrou a menos robusta em comparação com as demais. Portanto, seu uso é desencorajado, já que pode servir como ponto de fraqueza em variantes futuras. O mesmo não se pode afirmar, entretanto, sobre variações baseadas em códigos Quasi-cíclicos com matrizes de moderada densidade, como por exemplo o candidato alternativo da terceira rodada, BIKE [Aragon et al. 2017].

5. Considerações Finais

Este trabalho apresentou algumas variantes do McEliece, para encriptação baseada em códigos. Observou-se que variantes que mantiveram Código Goppa, se basearam em ajuste de parâmetros e uso de decodificação de listas têm se mostrado mais robustas em termos de segurança, mantendo complexidade de criptoanálise exponencial em relação ao tamanho do código linear.

Referências

Albrecht, M. R., Bernstein, D. J., Chou, T., Cid, C., Gilcher, J., and Lange, T. (2020). Classic McEliece. `https://classic.mceliece.org/`. Submissão NIST, Acesso em: 21/06/2021.

Aragon, N., Barreto, P., Bettaieb, S., Bidoux, L., Blazy, O., Deneuville, J.-C., Gaborit, P., Gueron, S., Guneysu, T., Melchor, C. A., et al. (2017). BIKE: bit flipping key encapsulation. `https://bikesuite.org/`. Submissão NIST, Acesso em: 21/06/2021.

Baldi, M., Bianchi, M., Chiaraluce, F., Rosenthal, J., and Schipani, D. (2016). Enhanced public key security for the mceliece cryptosystem. *Journal of Cryptology*, 29(1):1–27.

Baldi, M. and Chiaraluce, F. (2007). Cryptanalysis of a new instance of mceliece cryptosystem based on qc-ldpc codes. In *2007 IEEE International Symposium on Information Theory*, pages 2591–2595. IEEE.

Bardet, M., Chaulet, J., Dragoi, V., Otmani, A., and Tillich, J.-P. (2016). Cryptanalysis of the mceliece public key cryptosystem based on polar codes. In *Post-Quantum Cryptography*, pages 118–143. Springer.

Barreto, P., Biasi, F. P., Dahab, R., César, J., Pereira, G., and Ricardini, J. E. (2013). Introdução à criptografia pós-quântica. *Minicursos do XIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, SBSeg*.

Berger, T. P., Cayrel, P.-L., Gaborit, P., and Otmani, A. (2009). Reducing key length of the mceliece cryptosystem. In *International Conference on Cryptology in Africa*, pages 77–97. Springer.

Bernstein, D. J., Lange, T., and Peters, C. (2008). Attacking and defending the mceliece cryptosystem. In *International Workshop on Post-Quantum Cryptography*, pages 31–46. Springer.

Bernstein, D. J., Lange, T., and Peters, C. (2010). Wild mceliece. In *International Workshop on Selected Areas in Cryptography*, pages 143–158. Springer.

Blahut, R. E. (2003). *Algebraic codes for data transmission*. Cambridge university press.

Both, L. and May, A. (2018). Decoding linear codes with high error rate and its impact for lpn security. In *International Conference on Post-Quantum Cryptography*, pages 25–46. Springer.

Buchmann, J. (2013). *Introduction to cryptography*. Springer Science & Business Media.

Canteaut, A. and Sendrier, N. (1998). Cryptanalysis of the original mceliece cryptosystem. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 187–199. Springer.

Dragoi, V. and Kalachi, H. T. (2017). Cryptanalysis of a public key encryption scheme based on qc-ldpc and qc-mdpc codes. *IEEE Communications letters*, 22(2):264–267.

Engelbert, D., Overbeck, R., and Schmidt, A. (2007). A summary of mceliece-type cryptosystems and their security:. *Journal of Mathematical Cryptology*, 1(2):151–199.

Faugere, J.-C., Otmani, A., Perret, L., and Tillich, J.-P. (2010). Algebraic cryptanalysis of mceliece variants with compact keys. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 279–298. Springer.

Goldwasser, S. and Bellare, M. (1996). Lecture notes on cryptography. *Summer course "Cryptography and computer security" at MIT*, 1999:1999.

Goppa, V. D. (1970). A new class of linear correcting codes. *Problemy Peredachi Informatsii*, 6(3):24–30.

Hill, R. (1986). *A first course in coding theory*. Oxford University Press.

Kurose, J. F., Ross, K. W., and Zucchi, W. L. (2007). *Redes de Computadores e a Internet: uma abordagem top-down*. Pearson Addison Wesley.

Li, Y. X., Deng, R. H., and Wang, X. M. (1994). On the equivalence of mceliece's and niederreiter's public-key cryptosystems. *IEEE Transactions on Information Theory*, 40(1):271–273.

McEliece, R. J. (1978). A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116.

Minder, L. and Shokrollahi, A. (2007). Cryptanalysis of the sidelnikov cryptosystem. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 347–360. Springer.

Monico, C., Rosenthal, J., and Shokrollahi, A. (2000). Using low density parity check codes in the mceliece cryptosystem. In *2000 IEEE International Symposium on Information Theory (Cat. No. 00CH37060)*, page 215. IEEE.

Moufek, H., Guenda, K., and Gulliver, T. A. (2016). A new variant of the mceliece cryptosystem based on qc-ldpc and qc-mdpc codes. *IEEE Communications Letters*, 21(4):714–717.

Niederreiter, H. (1986). Knapsack-type cryptosystems and algebraic coding theory. *Prob. Control and Inf. Theory*, 15(2):159–166.

NIST (2017). Post-Quantum Cryptography PQC - Call for Proposals. `https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/Call-for-Proposals`. Acesso em: 21/06/2021.

Otmani, A., Tillich, J.-P., and Dallot, L. (2008). Cryptanalysis of mceliece cryptosystem based on quasi-cyclic ldpc codes. In *Proceedings of First International Conference on Symbolic Computation and Cryptography*, pages 69–81. LMIB Beihang University.

Otmani, A., Tillich, J.-P., and Dallot, L. (2010). Cryptanalysis of two mceliece cryptosystems based on quasi-cyclic codes. *Mathematics in Computer Science*, 3(2):129–140.

Overbeck, R. and Sendrier, N. (2009). Code-based cryptography. In *Post-quantum cryptography*, pages 95–145. Springer.

Repka, M. and Cayrel, P.-L. (2014). Cryptography based on error correcting codes: A survey. In *Multidisciplinary Perspectives in Cryptology and Information Security*, pages 133–156. IGI Global.

Sendrier, N. (2000). Finding the permutation between equivalent linear codes: The support splitting algorithm. *IEEE Transactions on Information Theory*, 46(4):1193–1203.

Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332.

Shrestha, S. R. and Kim, Y.-S. (2014). New mceliece cryptosystem based on polar codes as a candidate for post-quantum cryptography. In *2014 14th International Symposium on Communications and Information Technologies (ISCIT)*, pages 368–372. IEEE.

Van Leeuwen, J. and Leeuwen, J. (1990). *Handbook of theoretical computer science: Algorithms and complexity*, volume 1. Elsevier.

Yan, S. Y. (2009). *Primality testing and integer factorization in public-key cryptography*. Springer.

Zajac, M. R. (2014). Overview of the mceliece cryptosystem and its security. *Tatra Mt. Math. Publ*, 60:57–83.

Zhang, Q., Li, Z., and Song, C. (2011). The improvement of digital signature algorithm based on elliptic curve cryptography. In *2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC)*, pages 1689–1691. IEEE.