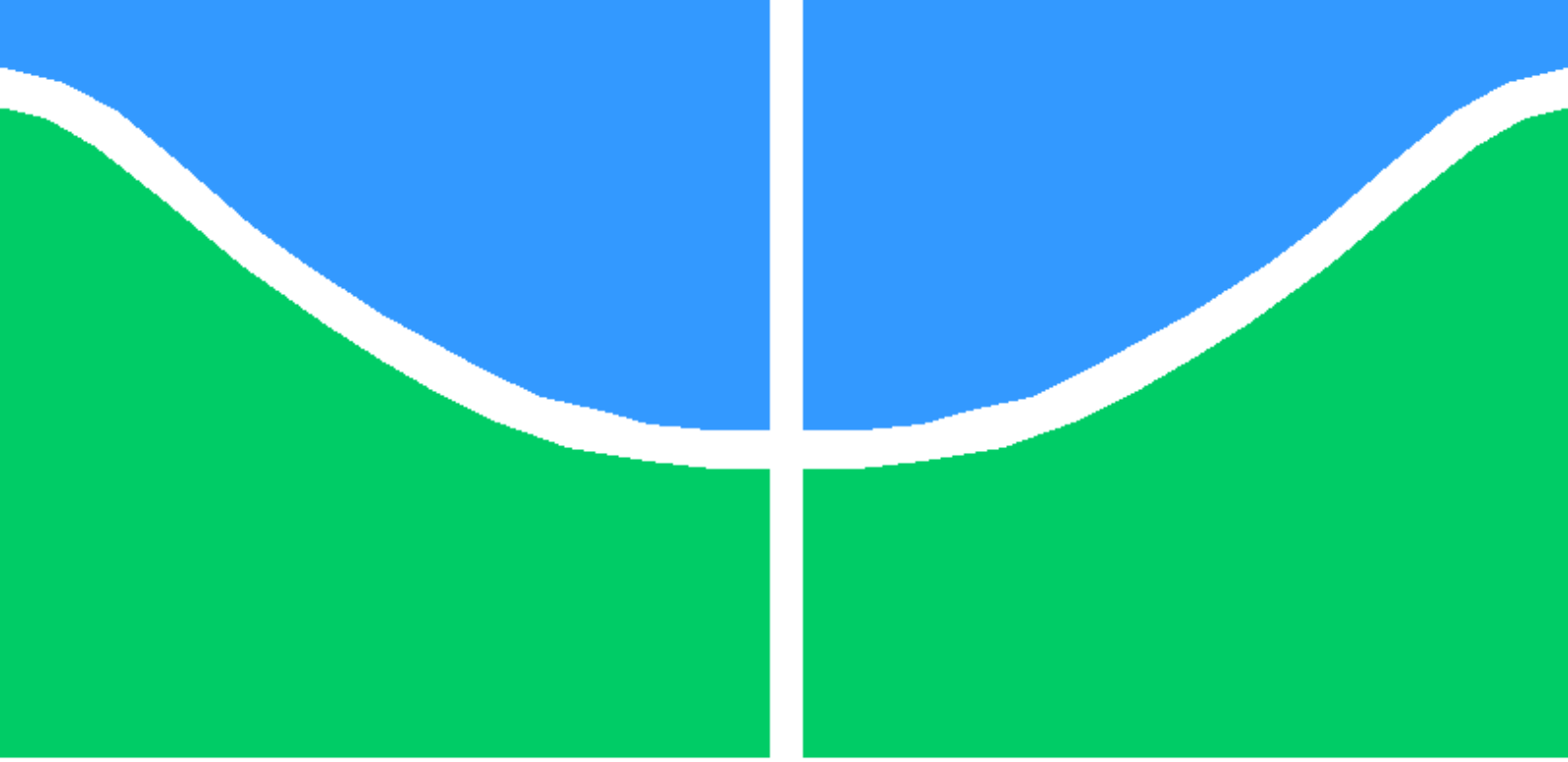

DISSERTAÇÃO DE MESTRADO

Protocolos com Segurança Demonstrável Baseados em Primitivas Criptográficas de Chave Pública

Rafael Baião Dowsley

Brasília, maio de 2010

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO

Protocolos com Segurança Demonstrável Baseados em Primitivas Criptográficas de Chave Pública

Rafael Baião Dowsley

Dissertação submetida ao Departamento de Engenharia Elétrica
como requisito parcial para obtenção do grau de
Mestre em Engenharia de Elétrica

Banca Examinadora

Anderson C. A. Nascimento - Ph.D.,
UnB/ENE (Orientador) _____

Ricardo Staciarini Puttini - Ph.D.,
UnB/ENE (Membro Interno) _____

Jeroen Antonius Maria van de Graaf - Ph.D.,
UFOP (Membro Externo) _____

FICHA CATALOGRÁFICA

DOWSLEY, RAFAEL BAIÃO. Protocolos com Segurança Demonstrável Baseados em Primitivas Criptográficas de Chave Pública [Distrito Federal] 2010.
viii, 54p. (ENE/FT/UnB, Mestre em Engenharia Elétrica, 2010)
Dissertação de Mestrado - Universidade de Brasília. Faculdade de Tecnologia. Departamento de Engenharia Elétrica.

1. Criptografia
2. Criptosistema de McEliece
3. Segurança IND-CCA2
4. Oblivious Transfer
I. ENE/FT/UnB
II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

DOWSLEY, RAFAEL BAIÃO (2010). Dissertação de Mestrado – *Protocolos com Segurança Demonstrável Baseados em Primitivas Criptográficas de Chave Pública* – Publicação 416/2010, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 54p.

CESSÃO DE DIREITOS

Rafael Baião Dowsley
SQSW 303, Bloco I, Apt. 206 - Sudoeste
CEP 70673-309 - Brasília - DF - Brasil.

A minha família.

Agradecimentos

Primeiramente eu gostaria de agradecer ao meu orientador, Prof. Anderson Nascimento, por me apresentar à pesquisa em criptografia, por me apresentar tantas idéias boas de tópicos de pesquisa e pela imensa e primordial ajuda durante esses anos no desenvolvimento deste trabalho.
Também gostaria de agradecer todas as pessoas com quem colaborei durante o desenvolvimento deste trabalho e cuja a ajuda foi essencial para o desenvolvimento do mesmo. Gostaria de agradecer especialmente os co-autores dos trabalhos que formam a base dessa dissertação: Jeroen van de Graaf, Goichiro Hanaoka, Hideki Imai, Jörn Müller-Quade e Anderson Nascimento. Gostaria também de agradecer ao Prof. Jeroen van de Graaf e ao Prof. Ricardo Puttini pela participação na banca de avaliação deste trabalho. Também gostaria de agradecer aos professores da UnB e os membros do LabRedes com que convivi durante esses anos de graduação e mestrado.
Por fim, um agradecimento muito especial a minha família e amigos que me ajudaram tanto durante todos esses anos de estudo.

Rafael Baião Dowsley

RESUMO

Nesse trabalho apresentamos um protocolo de Oblivious Transfer baseado nas hipóteses de McEliece. Também introduzimos um criptosistema de chave pública IND-CCA2 seguro (noção de segurança mais forte para criptosistemas de chave pública) baseado nas mesmas hipóteses. Devido ao fato de que fatorar números inteiros e calcular o logaritmo discreto são tarefas fáceis em computadores quânticos, vários outros protocolos de Oblivious Transfer e criptosistemas de chave pública se tornarão inseguros caso os computadores quânticos se tornem práticos. Os nossos protocolos são portanto alternativas no caso em que computadores quânticos se tornem práticos. Além disso também apresentamos uma versão modificada dos criptosistemas do tipo DDN que permite reduzir o tamanho do texto cifrado pela metade sem afetar os outros parâmetros.

ABSTRACT

In this work we show that a protocol of Oblivious Transfer based on McEliece assumptions. We also introduce a public-key encryption scheme based on the same assumptions that is IND-CCA2 secure (the strongest notion of security for public-key encryption schemes). Due to the fact that factoring integers and calculating the discrete logarithm are easy tasks for quantum computers, several other protocols of Oblivious Transfer and public-key encryption schemes will become insecure if quantum computers become practical, so our protocols are therefore alternatives in this case. We also show a technique to reduce by half the ciphertexts' size of DDN-like cryptosystems which does not affect the other parameters of the scheme.

CONTEÚDO

1 INTRODUÇÃO 1
1.1 CONTEXTO 1
1.1.1 OBLIVIOUS TRANSFER 1
1.1.2 CRIPTOSISTEMAS DE CHAVE PÚBLICA IND-CCA2 SEGUROS 2
1.2 RESULTADOS OBTIDOS 3
1.3 ORGANIZAÇÃO DO TRABALHO 5

2 PRELIMINARES 6
2.1 NOTAÇÃO 6
2.2 PROTOCOLOS DE OBLIVIOUS TRANSFER 6
2.3 PROTOCOLOS DE COMPROMETIMENTO DE BITS 7
2.4 CRIPTOSISTEMAS DE CHAVE PÚBLICA 8
2.5 CRIPTOSISTEMA DE MCELIECE 11
2.6 CRIPTOSISTEMA DE NIEDERREITER 12
2.7 HIPÓTESES DE MCELIECE 13
2.8 COMPROMETIMENTO DE BIT BASEADO NAS HIPÓTESES DE MCELIECE 14
2.9 CRIPTOSISTEMA DE CHAVE PÚBLICA ADMISSÍVEL 14
2.10 ASSINATURAS DIGITAIS 15
2.11 PROVAS DE CONHECIMENTO NULO NÃO-INTERATIVAS 16
2.12 CRIPTOSISTEMA DDN 17

3 OBLIVIOUS TRANSFER BASEADO NAS HIPÓTESES DE MCELIECE 19
3.1 PROTOCOLO DE OT SEGURO CONTRA ADVERSÁRIOS PASSIVOS 19
3.2 PROTOCOLO DE OT SEGURO CONTRA ADVERSÁRIOS MALICIOSOS 21
3.2.1 OT RANDOMIZADO COM ALTA PROBABILIDADE DE B TRAPAÇEAR 21
3.2.2 OT PARA ENTRADAS ESPECÍFICAS 24
3.2.3 REDUZINDO A PROBABILIDADE DE BOB TRAPAÇEAR 24
3.2.4 PROTOCOLO DE OT UTILIZANDO BCX 25
3.3 OT BASEADO NO CRIPTOSISTEMA DE NIEDERREITER 29

4 CRIPTOSISTEMA DE CHAVE PÚBLICA IND-CCA2 SEGURO BASEADO NAS HIPÓTESES DE MCELIECE SEM ORÁCULO ALEATÓRIO 31
4.1 CRIPTOSISTEMAS DE CHAVE PÚBLICA k-REPETIDOS 31
4.1.1 DEFINIÇÕES 31
4.1.2 SEGURANÇA IND-CCA2 A PARTIR DE UM CRIPTOSISTEMA DE CHAVE PÚBLICA k-REPETIDO 32
4.2 ESQUEMA DE MCELIECE RANDOMIZADO 36
4.2.1 SEGURANÇA DO ESQUEMA DE MCELIECE RANDOMIZADO k-REPETIDO 36

5 REDUZINDO O TAMANHO DO TEXTO CIFRADO EM CRIPTOSISTEMAS DO TIPO DDN 39
5.1 NOSSA CONSTRUÇÃO APERFEIÇOADA 39
5.1.1 O ESQUEMA 39
5.1.2 PROVA DE SEGURANÇA 40
5.1.3 PERFORMANCE: COMPARAÇÃO COM O ESQUEMA ORIGINAL DDN 44
5.2 EXPLICAÇÃO INTUITIVA DO NOSSO ARTIFÍCIO 44

6 CONCLUSÕES 47

REFERÊNCIAS 49

1 INTRODUÇÃO

1.1 CONTEXTO

No final da década de 70 Diffie e Hellman publicaram o famoso artigo [24] no qual definiram a criptografia de chave pública. Esse importante artigo juntamente com o crescimento das redes de computadores encorajaram a definição e implementação de novas primitivas necessárias para a execução segura de aplicações de redes, tais como: assinaturas digitais [40], dinheiro eletrônico [15], computação segura entre duas partes [71, 72], computação segura entre múltiplas partes [37, 14, 13] e provas de conhecimento nulo [38, 34, 6]. Também houve uma busca por definições de segurança mais fortes para criptosistemas de chave públicas, tais como: segurança semântica [39] e segurança IND-CCA2 [63]. Por fim, depois do advento da criptografia de chave pública, desenvolveu-se a área hoje conhecida como "segurança demonstrável". Um protocolo com segurança demonstrável possui uma prova que reduz a complexidade de violar determinada definição de segurança à dificuldade de resolver um certo problema computacional intratável. Apesar do crescimento explosivo de trabalhos em segurança computacional, percebemos, no início desse trabalho, que havia certos problemas de grande relevância, porém ainda sem solução na literatura. Em particular, percebemos que havia poucos protocolos com segurança demonstrável baseados em hipóteses diferentes das hipóteses de fatoração e da computação do logaritmo discreto. É sabido que um computador quântico poderia resolver esses problemas eficientemente, logo é necessário e importante procurarmos alternativas para o caso de um computador quântico se tornar realidade. De maneira mais específica, percebemos que ainda não existe na literatura nenhum protocolo de Oblivious Transfer baseado em hipóteses seguras no caso do advento de um computador quântico. Percebemos ainda que não existem criptosistemas de chave pública com segurança contra ataques de texto cifrado escolhidos de maneira adaptativa com segurança provada no modelo padrão (sem o uso de oráculos aleatórios) baseados em hipóteses computacionais resistentes contra ataques quânticos. Esta dissertação trata desses problemas. No restante do capítulo nós introduzimos alguns conceitos básicos necessários ao entendimento dos resultados obtidos.

1.1.1 Oblivious Transfer

1-2 Oblivious Transfer (OT) é uma primitiva de central importância na criptografia moderna, pois essa primitiva permite a implementação de computação segura entre duas partes [37, 51] e entre múltiplas

partes [19]. 1-2 OT é uma primitiva criptográfica na qual o emissor escolhe dois bits b_0 e b_1 e o receptor escolhe um bit c e recebe o bit b_c. O receptor não deve poder obter nenhuma informação sobre b_{1-c}. Além disso, o emissor não deve poder obter nenhuma informação sobre o bit c. Essa primitiva foi definida em [70, 30], uma primitiva similar foi definida em [62] e posteriormente outras variantes de OT foram apresentadas. Essas variantes são todas equivalentes, em [18] for demonstrado que OT pode ser implementado a partir da variante de Rabin, em [8] que OT para cadeias de bits pode ser implementado a partir de um OT, em [25] que o número de escolhas do receptor pode ser aumentado e em [7, 9] que OT pode ser implementado a partir das variantes denominadas XOT, GOT e UOT com repetições (essas são variantes mais fracas do OT). Nesse trabalho focaremos em 1-2 Oblivious Transfer.

Impagliazzo e Rudich [47] mostraram que uma redução caixa-preta entre OT computacionalmente seguro e funções unidirecionais não existe. Isso mostra que se houver uma redução entre essas primitivas, essa redução será difícil de ser encontrada, pois a vasta maioria das reduções criptográficas é caixa-preta. Mas é possível construir protocolos de OT computacionalmente seguros baseado em hipóteses genéricas como *Enhanced Trapdoor Permutations* [30, 35] e *Dense Trapdoor Permutations* [42]. OTs computacionalmente seguros também podem ser implementados baseados em hipóteses específicas como fatoração [62], Diffie-Hellman [3, 57, 1], *Quadratic or Higher-Order Residuosity*, ou da hipótese extendida de Riemann [48]. Também é possível implementar OT baseado em hipóteses físicas como ruído [20, 21].

1.1.2 Criptosistemas de Chave Pública IND-CCA2 Seguros

A segurança IND-CCA2 (indistinguishability of messages under adaptive chosen ciphertext attacks) [63] é a noção de segurança mais forte para criptosistemas de chave pública. Nessa noção de segurança, o adversário tem acesso a um oráculo de deciframento ao qual ele pode solicitar o deciframento de qualquer texto cifrado (exceto o texto cifrado desafio) e mesmo assim, ele não deve obter nenhuma informação sobre a mensagem relativa ao texto cifrado desafio. Essa noção de segurança é muito interessante, pois ela garante que o criptosistema pode ser composto arbitrariamente com quaisquer outros protocolos, sem que isso afete a sua segurança [44, 10]. Várias hipóteses computacionais já foram utilizadas para obter criptosistemas atendendo a essa noção de segurança e o estudo de construções genéricas de criptosistemas atingindo essa definição de segurança a partir de primitivas mais fracas é um tópico interessante. Dado uma permutação do tipo conhecido como *Enhanced One-way Trapdoor Permutations* é possível obter a segurança IND-CCA2 a partir de qualquer criptosistema semanticamente seguro [58, 26, 66, 53]. Construções eficientes são conhecidas baseadas em hipóteses relacionadas a teoria dos números [17] e em criptosistemas baseados

em identidades [11].

Recentemente Rosen e Segev apresentaram uma hipótese computacional simples e elegante para obter criptosistema de chave pública IND-CCA2 seguros: produtos correlacionados [65]. Eles demonstrarão algumas construções de produtos correlacionados baseado na existência das funções denominadas *lossy trapdoor functions* [61] que por sua vez podem ser baseadas no problema decisional de Diffie-Hellman e no problema conhecido como *Paillier's decisional residuosity problem* [61].

1.2 RESULTADOS OBTIDOS

OT Baseado nas Hipóteses de McEliece. No capítulo 3, que é parte de um trabalho [28] realizado conjuntamente com Jeroen van de Graaf, Jörn Müller-Quade e Anderson C. A. Nascimento, focamos em 1-2 Oblivious Transfer (OT) e construimos um protocolo de OT baseado nas duas hipóteses usadas no criptosistema de McEliece:

- Dificuldade de decodificar um código aleatório linear (que é NP-completo [4])
- Indistinguibilidade da chave pública do criptosistema de McEliece [54] de uma matriz aleatória.

É importante notar que não existe uma redução caixa-preta entre criptosistemas de chave pública e OT [33]. No entanto, explorando algumas propriedades algébricas dos textos cifrados gerados pelo criptosistema de McEliece podemos superar o resultado negativo de [33].

Nós apresentamos duas construções diferentes (com complexidades semelhantes): uma baseada na técnica denominada *cut-and-choose* e a outra baseado em uma generalização de um protocolo de comprometimento criado por Bennett e Rudich. Por fim apresentamos um protocolo de OT baseado no criptosistema de Niederreiter [59] (o dual do criptosistema de McEliece).

Como demonstrado em [69], fatorar inteiros e computar logaritmos discretos é fácil para computadores quânticos. Portanto, se a construção de um computador quântico se tornar viável, as hipóteses sobre a dificuldade de resolver esses problemas serão quebradas. Até onde sabemos, esse é o primeiro protocolo de OT baseado somente nas hipóteses de McEliece e, concorentemente com [52], o primeiro protocolo de OT computacionalmente seguro para o qual não se conhece um algoritmo quântico que quebre a segurança. No entanto, para obter um protocolo de complexidade equivalente, Kobara et al.[52] utilizam hipóteses adicionais: a hipótese do oráculo aleatório e hipótese denominada *Permuted Kernels*. Além disso, eles

utilizam o esquema de provas de conhecimento nulo do Shamir [68], o que é evitado em nossa construção.

Nós consideramos somente adversários estáticos. Isto é, assumimos que Alice ou Bob é corrompido antes do começo da execução do protocolo.

Criptosistema de Chave Pública IND-CCA2 Seguro Baseado nas Hipóteses de McEliece. No capítulo 4, que é parte de um trabalho [27] realizado conjuntamente com Jörn Müller-Quade e Anderson C. A. Nascimento, mostramos que as idéais de Rosen e Segev podem ser aplicadas para obter o primeiro criptosistema de chave pública IND-CCA2 seguro baseado nas hipóteses de McEliece. Baseado nas definições de produtos correlacionados [65], nós definimos um novo tipo de criptosistema de chave pública denominado k-repetido e mostramos que a construção proposta por [65] pode ser aplicada nesse novo cenário. Após isso, provamos que uma versão do criptosistema de McEliece randomizado [46] atende aos requisitos de segurança de criptosistemas k-repetidos e assim podemos utilizar-lo para construir um criptosistema IND-CCA2 seguro sem utilizar oráculos aleatórios. O criptosistema resultante cifra vários bits, ao contrário de [65] que cifra somente 1 bit. As chaves públicas e privadas, assim como o texto cifrado, são expandidos por um fator k quando comparados a versão original do McEliece. Além disso, nosso resultado implica uma nova construção de produtos correlacionados, baseada nas hipóteses de McEliece.

Em um trabalho simultâneo e independente, Goldwasser e Vaikuntanathan apresentarão um novo criptosistema de chave pública IND-CCA2 seguro baseado em reticulados usando a construção de Rosen e Segev. O esquema deles assume que o problema denominado *Learning with Errors* (LWE) é difícil [64].

Reduzindo o Tamanho do Texto Cifrado em Criptosistemas do Tipo DDN. No capítulo 5, que é parte de um trabalho [29] conjunto com Goichiro Hanaoka, Hideki Imai e Anderson C. A. Nascimento, mostramos um método para reduzir o tamanho do texto cifrado do criptosistema de Dolev-Dwork-Naor (DDN) [26] pela metade. Nosso esquema também reduz o custo computacional do criptosistema.

Apesar de termos apresentado a prova de segurança somente para o esquema original DDN, nossa idéia pode ser facilmente aplicada em outras construções IND-CCA2 seguras que utilizam vários pares de chaves públicas/privadas. Nossa técnica pode, por exemplo, ser aplicada para reduzir o tamanho do texto cifrado e o custo computacional nos seguintes esquemas: Rosen-Segev [65], Pass-Shelat-Vaikuntanathan [60], Hanaoka-Imai-Ogawa-Watanabe [43] e o nosso esquema descrito no capítulo 4.

Reduzir o tamanho do texto cifrado do DDN é uma contribuição teórica, já que esquemas utilizados na prática não são em geral baseados em reduções genéricas. No entanto, no caso de esquemas pós-quânticos,

todos os protocolos IND-CCA2 seguros e sem oráculo aleatório que são conhecidos são baseados em técnicas de redução genérica [65, 60, 27]. Portanto, reduzir a complexidade computacional e de comunicação desse tipo de esquema é realmente uma questão importante para obter criptosistemas de chave pública IND-CCA2 seguros mais práticos no caso de um eventual progresso dos computadores quânticos.

1.3 ORGANIZAÇÃO DO TRABALHO

Primeiramente apresentamos no capítulo 2 alguns conceitos criptográficos que serão utilizados no restante desse trabalho. O capítulo 3 apresenta o nosso protocolo de OT baseado nas hipóteses de McEliece. No capítulo 4 mostramos nosso criptosistema de chave pública IND-CCA2 seguro baseado nas mesmas hipóteses. Já no capítulo 5 apresentamos a nossa modificação para reduzir o tamanho do texto cifrado nos criptosistemas do tipo DDN. Por fim, no capítulo 6, discutimos as conclusões desse trabalho e alguns problemas que continuam em aberto.

2 PRELIMINARES

2.1 NOTAÇÃO

Se x for uma cadeia de bits, denotamos por $|x|$ o seu tamanho, enquanto $|S|$ representa a cardinalidade de um conjunto S. Se $n \in \mathbb{N}$, então 1^n denota a cadeia de bits com n uns. $s \leftarrow S$ denota a operação de escolher um elemento s do conjunto S de forma uniformemente aleatória. $w \leftarrow \mathcal{A}(x, y, \ldots)$ representa o ato de executar o algoritmo $\mathcal{A}$ com entradas $x, y, \ldots$ e produzindo a saída w. Escrevemos $w \leftarrow \mathcal{A}^{\mathcal{O}}(x, y, \ldots)$ para representar um algoritmo $\mathcal{A}$ com acesso a um oráculo $\mathcal{O}$. $\mathcal{U}_n$ é um oráculo que retorna um elemento aleatório de $\{0,1\}^n$. Se a e b forem duas cadeias de bits, denotamos por $\langle a, b\rangle$ o produto interno deles módulo 2 e por $a \oplus b$ o ou-exlcusivo bit-a-bit deles. Se a e b forem duas cadeias de bits ou duas matrizes, denotamos por $a|b$ sua concatenação. A transposta da matriz M é denotada por M^T.

Denotamos por Pr[E] a probabilidade de que o evento E ocorra. Uma função $f(n) : \mathbb{N} \rightarrow \mathbb{R}$ é dita desprezível se para todo polinômio $p(n)$ existir um número n_L tal que para todo $n > n_L$

$$|f(n)| < \frac{1}{p(n)}.$$

Dizemos que um evento $E(n)$ (indexado por um parâmetro n) tem probabilidade altíssima de ocorrer se Pr[$\overline{E(n)}$] for desprezível. Duas sequências $\{X_n\}_{n\in\mathbb{N}}$ e $\{Y_n\}_{n\in\mathbb{N}}$ de variáveis aleatórias são ditas *computacionalmente indistinguíveis*, denotado por $X \overset{c}{\approx} Y$, se para todo algoritmo não-uniforme, probabilístico e de tempo polinomial D existir uma função desprezível $\epsilon(\cdot)$ tal que para todo $n \in \mathbb{N}$,

$$|Pr[D(X_n) = 1] - Pr[D(Y_n) = 1]| < \epsilon(n).$$

2.2 PROTOCOLOS DE OBLIVIOUS TRANSFER

Sejam a emissora Alice e o receptor Bob modelados por máquinas de Turing probabilísticas de tempo polinomial A e B. Eles tem como entrada o parâmetro de segurança n.

Denotamos por $Vista_{\widetilde{A}}(\widetilde{A}(z), B(c))$ e $Vista_{\widetilde{B}}(A(b_0, b_1), \widetilde{B}(z))$ as *vistas* de Alice e Bob desonestos, respectivamente, que representam suas entradas z, os resultados das computações locais e todas as men-

sagens trocadas entre eles. Nossa definição de segurança é baseada na de [48], a qual adaptamos para protocolos com mais de duas mensagens.

Definição 2.2.1 *Dizemos que um protocolo* $[A, B](b_0, b_1; c)$ *implementa seguramente Oblivious Transfer se no final da sua execução ele atender aos seguintes requisitos:*

- Corretude: *quando ambas as partes são honestas, Bob produz a saída* b_c *e Alice não tem nenhuma saída.*
- Segurança para Alice: *Para todo adversário probabilístico de tempo polinomial* $\tilde{B}$*, todo entrada* z*, e uma fita de aleatoriedade (suficientemente grande)* R_B*, deve haver um bit de escolha* c *tal que para* $b_c \in \{0, 1\}$*, a distribuição (sobre a aleatoriedade usada por Alice) das execuções usando a aleatoriedade* R_B *com Alice tendo como entradas* b_c *and* $b_{\overline{c}} = 0$ *deve ser computacionalmente indistinguível das execuções usando a aleatoriedade* R_B *com Alice tendo como entradas* b_c *and* $b_{\overline{c}} = 1$
- Segurança para Bob: *Para todo adversário probabilístico de tempo polinomial* $\widetilde{A}$*, qualquer parâmetro de segurança* n *e qualquer entrada* z *de tamanho polinomial em* n*, a visão que* $\widetilde{A}(z)$ *obtêm quando a entrada de Bob é* $c = 0$ *é computacionalmente indistinguível da obtida quando a entrada de Bob é* $c = 1$*, denotado por:*

$$Vista_{\widetilde{A}}(\widetilde{A}(z), B(0))|_z \stackrel{c}{\approx} Vista_{\widetilde{A}}(\widetilde{A}(z), B(1))|_z.$$

Um protocolo é dito seguro contra partes honestas-mas-curiosas, se as definições anteriores forem verdadeiras no caso que Alice e Bob seguem os procedimentos do protocolo. Um protocolo de Oblivious Transfer é incondicionalmente seguro contra uma parte se as propriedades de segurança forem atendidas mesmo quando essa parte não for limitada computacionalmente.

2.3 PROTOCOLOS DE COMPROMETIMENTO DE BITS

Utilizamos protocolos de comprometimento de bits em nossas construções. Um protocolo de comprometimento de bits (isto é, de cadeias de bits) consiste de duas fases. Na primeira fase, chamada de comprometimento, a emissora Alice envia uma evidência para o receptor Bob do valor com o qual ela se comprometeu b. Por um lado, um requisito do protocolo de comprometimento é que Bob não possa obter

nenhuma informação sobre o valor b antes que Alice revele esse valor para ele na segunda fase, chamada de fase de revelação. Por outro lado, não deve ser possível que Alice convença Bob que ela havia se comprometido com um valor diferente de b. Usamos uma notação semelhante a utilizada para Oblivious Transfer e denotamos por $Vista_{\widetilde{A}}(\widetilde{A}(z), B(a))$ e $Vista_{\widetilde{B}}(A(b), \widetilde{B}(z))$ as *vistas* de Alice e Bob desonestos, respectivamente, que representam suas entradas z, os resultados das computações locais e todas as mensagens trocadas entre eles. Nossa definição é baseada em [56].

Definição 2.3.1 *Dizemos que um protocolo $[A, B](b)$ implementa seguramente um comprometimento de bits se ao final da execução entre Alice e Bob (que são máquinas de Turing probabilísticas de tempo polinomial tendo como entrada o parâmetro de segurança n) os seguintes requisitos forem atendidos:*

- Corretude: *Quando as duas partes são honestas, Bob aceita b ao final da segunda fase.*
- Sigilo (ou Ocultação): *Para todo adversário probabilístico de tempo polinomial $\widetilde{B}$, qualquer parâmetro de segurança n, qualquer entrada z de tamanho polinomial em n e qualquer $k \in \mathbb{N}$ devemos ter que no período entre o final da fase comprometimento e o início da fase de revelação, a vista de $\widetilde{B}(z)$ quando a entrada de Alice é $b \in \{0,1\}^k$ deve ser computacionalmente indistinguível da vista quando a entrada de Alice é qualquer outro valor $b' \in \{0,1\}^k$, $b' \neq b$:*

$$Vista_{\widetilde{B}}(A(b), \widetilde{B}(z))|_z \stackrel{c}{\approx} Vista_{\widetilde{B}}(A(b'), \widetilde{B}(z))|_z$$

- Desambigüidade (ou Amarração): *Para todo adversário probabilístico de tempo polinomial $\widetilde{A}$, qualquer parâmetro de segurança n, qualquer entrada z de tamanho polinomial em n e qualquer $k \in \mathbb{N}$ deve existir um $b \in \{0,1\}^k$ que possa ser computado por Alice ao final da fase de comprometimento e tal que a probabilidade de que $\widetilde{A}(b')$, $b' \neq b$ seja aceito por Bob ao final da fase de revelação seja desprezível em n.*

Um protocolo de comprometimento de bits é incondicionalmente seguro contra uma parte se as propriedades de segurança forem atendidas mesmo quando essa parte não for limitada computacionalmente.

2.4 CRIPTOSISTEMAS DE CHAVE PÚBLICA

Um Criptosistema de Chave Pública (PKE) é definido da seguinte maneira:

Definição 2.4.1 *(Criptosistema de Chave Pública). Um criptosistema de chave pública consiste de três algoritmos (*Gen*,* Enc*,* Dec*) tais que:*

- Gen *é um algoritmo probabilístico de geração de chaves que executa em tempo polinomial. Esse algoritmo tem como entrada o parâmetro de segurança* 1^n *e produz como saídas a chave pública* pk *e chave secreta* sk*. A chave pública especifica o espaço de mensagens* $\mathcal{M}$ *e o espaço de textos cifrados* $\mathcal{C}$*.*

- Enc *é um algoritmo (que pode ser probabilístico) de ciframento de mensagens que executa em tempo polinomial. Esse algoritmo recebe como entrada a chave pública* pk *e uma mensagem* $\mathsf{m} \in \mathcal{M}$ *e produz como saída um texto cifrado* $\mathsf{c} \in \mathcal{C}$*.*

- Dec *é um algoritmo determinístico de deciframento em tempo polinomial. Esse algoritmo recebe como entrada a chave secreta* sk *e um texto cifrado* c *e produz como saída uma mensagem* $\mathsf{m} \in \mathcal{M}$ *ou um símbolo de erro* $\perp$*.*

- *(Corretude) Para qualquer par de chaves pública e privada geradas por* Gen *e qualquer mensagem* $\mathsf{m} \in \mathcal{M}$ *temos que* $\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{m})) \neq \mathsf{m}$ *tem probabilidade desprezível sobre as aleatoriedades utilizadas por* Gen *e* Enc*.*

Abaixo definiremos duas noções de segurança:

- Segurança IND-CPA (indistinguishability against chosen-plaintext attacks) [39]: Também conhecida como segurança semântica. Basicamente essa definição de segurança garante que um adversário com acesso aos textos cifrados não consegue extrair nenhuma informação sobre as mensagens cifradas neles.

- Segurança IND-CCA2 (indistinguibilidade against adaptive chosen-ciphertext attacks) [63]: Essa definição de segurança basicamente garante que um adversário não consegue obter nenhuma informação relativa a qual mensagem corresponde a um determinado texto cifrado mesmo que o adversário tenha acesso a um oráculo de deciframento ao qual ele possa solicitar o deciframento de qualquer outro texto cifrado. Essa é a definição de segurança mais forte existente para criptosistemas de chave pública e implica que o criptosistema pode ser composto com quaisquer outros protocolos sem que isso afete a sua segurança [44, 10].

As nossas definições baseadas em jogos seguem o modelo de [45].

Definição 2.4.2 *(Segurança IND-CPA). Dado um adversário de dois estágios $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ contra o criptosistema de chave pública* PKE*, associamos a ele o seguinte experimento* $\mathsf{Exp}^{cpa}_{\mathsf{PKE},\mathcal{A}}(n)$*:*

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$

$(\mathsf{m}_0, \mathsf{m}_1, estado) \leftarrow \mathcal{A}_1(\mathsf{pk})$ *tal que* $|\mathsf{m}_0| = |\mathsf{m}_1|$

$b \leftarrow \{0, 1\}$

$\mathsf{c}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_b)$

$b' \leftarrow \mathcal{A}_2(\mathsf{c}^*, estado)$

Se $b = b'$ *retorna* 1*, caso contrário retorna* 0*.*

Definimos a vantagem do adversário $\mathcal{A}$ nesse experimento como:

$$\mathsf{Adv}^{cpa}_{\mathsf{PKE},\mathcal{A}}(n) = |Pr[\mathsf{Exp}^{cpa}_{\mathsf{PKE},\mathcal{A}}(n) = 1] - \tfrac{1}{2}|$$

Dizemos que um criptosistema de chave pública PKE *é IND-CPA seguro se a vantagem de $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ nesse experimento for uma função desprezível de n para todo adversário probabilístico de tempo polinomial $\mathcal{A}$.*

Definição 2.4.3 *(Segurança IND-CCA2). Dado um adversário de dois estágios $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ contra o criptosistema de chave pública* PKE*, associamos a ele o seguinte experimento* $\mathsf{Exp}^{cca2}_{\mathsf{PKE},\mathcal{A}}(n)$*:*

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$

$(\mathsf{m}_0, \mathsf{m}_1, estado) \leftarrow \mathcal{A}_1^{\mathsf{Dec}(\mathsf{sk},\cdot)}(\mathsf{pk})$ *tal que* $|\mathsf{m}_0| = |\mathsf{m}_1|$

$b \leftarrow \{0, 1\}$

$\mathsf{c}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{m}_b)$

$b' \leftarrow \mathcal{A}_2^{\mathsf{Dec}(\mathsf{sk},\cdot)}(\mathsf{c}^*, estado)$

Se $b = b'$ *retorna* 1*, caso contrário retorna* 0*.*

O adversário $\mathcal{A}_2$ não pode solicitar o deciframento de c^* *ao oráculo* $\mathsf{Dec}(\mathsf{sk}, \cdot)$*. Definimos a vantagem do adversário $\mathcal{A}$ nesse experimento como:*

$$\mathsf{Adv}^{cca2}_{\mathsf{PKE},\mathcal{A}}(n) = |Pr[\mathsf{Exp}^{cca2}_{\mathsf{PKE},\mathcal{A}}(n) = 1] - \tfrac{1}{2}|$$

Dizemos que um criptosistema de chave pública PKE *é IND-CCA2 seguro se a vantagem de* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *nesse experimento for uma função desprezível de* n *para todo adversário probabilístico de tempo polinomial* $\mathcal{A}$ *que faça um número polinomial de consultas ao oráculo.*

2.5 CRIPTOSISTEMA DE MCELIECE

Nesta seção definimos o criptosistema de McEliece [55] seguindo a descrição usada por [46]. O criptosistema de McEliece consiste de três algoritmos $(\mathsf{Gen}_{\mathrm{McE}}, \mathsf{Enc}_{\mathrm{McE}}, \mathsf{Dec}_{\mathrm{McE}})$ tais que:

– O algoritmo probabilístico de tempo polinomial para a geração de chaves, $\mathsf{Gen}_{\mathrm{McE}}$, executa da seguinte maneira:

1. Gera uma matriz geradora $\mathbf{G}$ de dimensões $l \times n$ do código de Goppa. É assumido que existe um algoritmo eficiente de correção de erros $\mathsf{Corrigir}$ que sempre pode corrigir até t erros.
2. Gera uma matriz não-singular aleatória $\mathbf{S}$ de dimensões $l \times l$.
3. Gera uma matriz de permutação aleatória $\mathbf{T}$ de dimensões $n \times n$.
4. Estabelece que $\mathbf{P} = \mathbf{SGT}$, $\mathcal{M} = \{0,1\}^l$, $\mathcal{C} = \{0,1\}^n$.
5. Gera as saídas $\mathsf{pk} = (\mathbf{P}, t, \mathcal{M}, \mathcal{C})$ e $\mathsf{sk} = (\mathbf{S}, \mathbf{G}, \mathbf{T})$.

– O algoritmo probabilístico de tempo polinomial de ciframento, $\mathsf{Enc}_{\mathrm{McE}}$, recebe como entradas a chave pública pk e uma mensagem $\mathsf{m} \in \{0,1\}^l$ e gera como saída o texto cifrado $\mathsf{c} = \mathsf{m}\mathbf{P} \oplus \mathsf{e}$, onde $\mathsf{e} \in \{0,1\}^n$ é um vetor aleatório com peso de Hamming t.

– O algoritmo determinístico de tempo polinomial de deciframento, $\mathsf{Dec}_{\mathrm{McE}}$, executa da seguinte maneira:

1. Computa $\mathsf{c}\mathbf{T}^{-1} = (\mathsf{m}\mathbf{S})\mathbf{G} \oplus \mathsf{e}\mathbf{T}^{-1}$, onde $\mathbf{T}^{-1}$ denota a matriz inversa de $\mathbf{T}$.
2. Computa $\mathsf{m}\mathbf{S} = \mathsf{Corrigir}(\mathsf{c}\mathbf{T}^{-1})$.
3. Gera como saída $\mathsf{m} = (\mathsf{m}\mathbf{S})\mathbf{S}^{-1}$.

No nosso criptosistema de chave pública IND-CCA2 seguro, apresentado no capítulo 4, nós utilizamos uma versão levemente modificada do criptosistema de McEliece. Ao invés de criar o vetor de erro

escolhendo ele aleatoriamente do conjunto de vetores com peso de Hamming t, nós o geramos escolhendo cada um de seus bits de acordo com uma distribuição de Bernoulli $\mathcal{B}_\theta$ com parâmetro $\theta = \frac{t}{n} - \epsilon$ para algum $\epsilon > 0$. Devido a Lei dos Grandes Números, a probabilidade do vetor de erro resultante estar fora da capacidade de correção do código é desprezível.

2.6 CRIPTOSISTEMA DE NIEDERREITER

O criptosistema de Niederreiter consiste de três algoritmos $\mathrm{NR} = (\mathsf{Gen}_{\mathrm{NR}}, \mathsf{Enc}_{\mathrm{NR}}, \mathsf{Dec}_{\mathrm{NR}})$ tais que:

- O algoritmo probabilístico de tempo polinomial para a geração de chaves, $\mathsf{Gen}_{\mathrm{NR}}$, executa da seguinte maneira:

  1. Gera uma matriz de paridade $\mathbf{H}$, de dimensões $(n-k) \times n$, de um código binário irredutível de Goppa $\mathcal{G}$ com dimensão máxima k e que possa corrigir eficientemente até t erros.
  2. Gera uma matriz não-singular aleatória $\mathbf{M}$ de dimensões $(n-k) \times (n-k)$.
  3. Gera uma matriz de permutação aleatória $\mathbf{P}$ de dimensões $n \times n$.
  4. Estabelece que $\mathbf{H}' = \mathbf{MHP}$, que $\mathcal{M}$ é formado pelas cadeias de bits de tamanho n e peso de Hamming t e que $\mathcal{C}$ é formado pelas cadeias de bits de tamanho $(n-k)$.
  5. Gera as saídas $\mathsf{pk} = (\mathbf{H}', t, \mathcal{M}, \mathcal{C})$ e $\mathsf{sk} = (\mathbf{M}, \mathbf{P}, \mathbf{D}_\mathcal{G})$ (onde $\mathbf{D}_\mathcal{G}$ é um algoritmo eficiente de decodificação de síndrome para o código $\mathcal{G}$).

- O algoritmo probabilístico de tempo polinomial de ciframento, $\mathsf{Enc}_{\mathrm{NR}}$, recebe como entradas a chave pública pk e uma mensagem $\mathsf{m} \in \{0,1\}^n$ de peso de Hamming t. Ele gera como saída a síndrome $\mathsf{s} = \mathbf{H}'m^T$.

- O algoritmo determinístico de tempo polinomial de deciframento, $\mathsf{Dec}_{\mathrm{NR}}$, tem como entradas o texto cifrado s e a chave secreta sk. Ele executa da seguinte maneira:

  1. Computa $\mathbf{M}^{-1}\mathsf{s} = \mathbf{HP}\mathsf{m}^T$.
  2. Computa $\mathbf{D}_\mathcal{G}(\mathbf{HP}\mathsf{m}^T)$ para obter $\mathbf{P}\mathsf{m}^T$.
  3. Computa $\mathsf{m}^T = \mathbf{P}^{-1}\mathbf{P}\mathsf{m}^T$ e gera como saída m.

2.7 HIPÓTESES DE MCELIECE

Nesta seção apresentamos as hipóteses de McEliece, que serão utilizadas durante esse trabalho.

Assumimos que não existe um algoritmo eficiente que possa distinguir entre a matriz geradora "embaralhada" P do código de Goppa (embaralhada de acordo com a descrição da seção anterior) e uma matriz aleatória de mesmo tamanho. O melhor algoritmo para distinguir essas matrizes atualmente é o de Courtois et al. [16] que é baseado em um algoritmo chamado *support splitting algorithm* [67].

Hipótese 2.7.1 *Não existe algoritmo probabilístico de tempo polinomial que possa distinguir com probabilidade não desprezível entre a chave pública P do criptosistema de McEliece e uma matriz aleatória de mesmo tamanho.*

Essa hipótese foi utilizada em [16] para construir um esquema de assinatura digital.

Também assumimos que o Problema da Decodificação de Síndrome (Syndrome Decoding Problem) é díficil. Esse problema é NP-completo [4], e todos os algoritmos conhecidos atualmente para resolver esse problema são de tempo exponencial. Os melhores algoritmos foram apresentados por Canteaut e Chabaud [12] e recentemente por Bernstein et al. [5].

Hipótese 2.7.2 *O Problema da Decodificação de Síndrome é difícil para todo algoritmo probabilístico de tempo polinomial.*

Esse problema é equivalente ao problema conhecido como Learning Parity with Noise (LPN). Abaixo apresentamos a definição do problema LPN seguindo a descrição de [46].

Definição 2.7.3 *(Problema LPN) Sejam r, a cadeias de bits de tamanho l. Consideramos a distribuição de Bernoulli $\mathcal{B}_\theta$ com parâmetro $\theta \in (0, \frac{1}{2})$. Seja $\mathcal{Q}_{r,\theta}$ a seguinte distribuição:*

$$\{(a, \langle r, a\rangle \oplus v) | a \leftarrow \{0,1\}^l, v \leftarrow \mathcal{B}_\theta\}$$

Para um adversário $\mathcal{A}$ que tenta descobrir a cadeia de bits aleatória r, definimos a sua vantagem como:

$$\mathsf{Adv}_{\mathsf{LPN}_\theta,\mathcal{A}}(l) = \Pr[\mathcal{A}^{\mathcal{Q}_{r,\theta}} = r | r \leftarrow \{0,1\}^l]$$

O problema LPN_θ com parâmetro θ é difícil se a vantagem do adversário $\mathcal{A}$ for desprezível para todo $\mathcal{A}$ probabilístico e de tempo polinomial que faça um número polinomial de consultas ao oráculo.

A segurança dos criptosistemas de McEliece e Niederreiter são equivalentes [23] (a dificuldade de quebrar o criptosistema de Niederreiter é a mesma de quebrar o criptosistema de McEliece com os mesmos parâmetros de segurança).

2.8 COMPROMETIMENTO DE BIT BASEADO NAS HIPÓTESES DE MCELIECE

Em nossos protocolos de Oblivious Transfer necessitamos um esquema de comprometimento de bits que seja baseado nas mesmas hipóteses de segurança. Uma alternativa seria utilizar a versão modificada do esquema de McEliece que é IND-CPA segura [46]. Porém temos uma outra alternativa melhor.

De acordo com um conhecido resultado de Naor [56], um esquema de comprometimento de bits pode ser construido usando um gerador de números pseudo-aleatórios. Tal gerador pode ser construido eficientemente usando o problema da decodificação de síndrome conforme descrito por Fischer e Stern [31]. No esquema do Naor o Sigilo é incondicional e a Desambigüidade é computacionalmente segura. Além disso esse esquema atende ao requisito de corretude. Portanto, utilizando esta construção para obter o comprometimento de bit, estaremos usando somente uma das hipóteses de McEliece. Além disso, para esquemas de comprometimento de cadeias de bits essa construção do Naor é muito eficiente.

2.9 CRIPTOSISTEMA DE CHAVE PÚBLICA ADMISSÍVEL

Abaixo apresentaremos o conceito de Criptosistema de Chave Pública Admissível (admissible PKE) que implica na segurança IND-CPA [46]. Abaixo $\mathsf{Enc}(\mathsf{pk}, \mathsf{m}, \mathsf{r})$ denota um criptosistema de chave pública cifrando uma mensagem m com a chave pública pk e usando a aleatoriedade r.

Definição 2.9.1 *(Criptosistema de Chave Pública Admissível [46]) Um criptosistema de chave pública* $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *com espaço de mensagens* $\mathcal{M}$ *e espaço de aleatoriedade* $\mathcal{R}$ *é dito admissível se existir um par de algoritmos determinísticos de tempo polinomial* Enc_1 *e* Enc_2 *satisfazendo as seguintes propriedades:*

- *Divisibilidade:* Enc_1 *tem como entrada a chave pública* pk *e* $\mathsf{r} \in \mathcal{R}$*, e gera como saída uma cadeia de bits de tamanho* $p(n)$*.* Enc_2 *tem como entrada a chave pública* pk *e* $\mathsf{m} \in \mathcal{M}$*, e gera como saída uma cadeia de bits de tamanho* $p(n)$*.* p *denota um polinômio em* n*. Então para toda chave pública* pk *gerada por* Gen*,* $\mathsf{r} \in \mathcal{R}$ *e* $\mathsf{m} \in \mathcal{M}$*, temos que* $\mathsf{Enc}_1(\mathsf{pk}, \mathsf{r}) \oplus \mathsf{Enc}_2(\mathsf{pk}, \mathsf{m}) = \mathsf{Enc}(\mathsf{pk}, \mathsf{m}, \mathsf{r})$.

– *Pseudo-aleatoriedade: Considere um adversário probabilístico de tempo polinomial $\mathcal{A}$ contra* PKE*, nós associamos a ele o seguinte experimento* $\mathsf{Exp}^{ind}_{\mathsf{PKE},\mathcal{A}}(n)$*:*

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$

$\mathsf{s}_0 \leftarrow \mathcal{U}_{p(n)}$

$\mathsf{r} \in \mathcal{R}$

$\mathsf{s}_1 \leftarrow \mathsf{Enc}_1(\mathsf{pk}, \mathsf{r})$

$b \leftarrow \{0, 1\}$

$b' \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{s}_b)$

Se $b = b'$ *retorna* 1*, caso contrário retorna* 0*.*

Definimos a vantagem de $\mathcal{A}$ nesse experimento como:

$$\mathsf{Adv}^{ind}_{\mathsf{PKE},\mathcal{A}}(n) = |Pr[\mathsf{Exp}^{ind}_{\mathsf{PKE},\mathcal{A}}(n) = 1] - \tfrac{1}{2}|$$

Então para todo adversário probabilístico de tempo polinomial $\mathcal{A}$, a vantagem de $\mathcal{A}$ nesse experimento deve ser uma função desprezível de n.

2.10 ASSINATURAS DIGITAIS

Abaixo definimos o conceito de Assinatura Digital (Signature Scheme, doravante denotado por SS) e explicamos a noção de segurança chamada One-time Strong Unforgeability (OTSU).

Definição 2.10.1 *(Assinatura Digital). Uma assinatura digital consiste em três algoritmos (*Gen*,* Sign*,* Ver*) tais que:*

– Gen *é um algoritmo probabilístico de geração de chaves que executa em tempo polinomial. Ele recebe como entrada o parâmetro de segurança 1^n e gera como saída uma chave de verificação* vk *e uma chave de assinatura* dsk*. A chave de verificação especifica o espaço de mensagens $\mathcal{M}$ e o espaço de assinaturas $\mathcal{S}$.*

– Sign *é um algoritmo (que pode ser probabilístico) de assinatura que executa em tempo polinomial. Ele recebe como entrada a chave de assinatura* dsk *e uma mensagem* $\mathsf{m} \in \mathcal{M}$*, e gera como saída uma assinatura* $\sigma \in \mathcal{S}$*.*

– Ver *é um algoritmo determinístico de verificação que executa em tempo polinomial. Ele recebe como entrada a chave de verificação* vk*, uma mensagem* $\mathsf{m} \in \mathcal{M}$ *e uma assinatura* $\sigma \in \mathcal{S}$*, e gera como saída um bit indicando se* σ *é uma assinatura válida para* m *ou não (isto é, o algoritmo gera como saída* 1 *se a assinatura for válida e* 0 *em outro caso).*

– *Para qualquer par de chaves de assinatura e de verificação gerados por* Gen*, qualquer mensagem* $\mathsf{m} \in \mathcal{M}$ *devemos ter* $\mathsf{Ver}(\mathsf{vk}, \mathsf{m}, \mathsf{Sign}(\mathsf{dsk}, \mathsf{m})) \neq 1$ *somente com probabilidade desprezível sobre as aleatoriedades utilizadas por* Gen *e* Sign*.*

Definição 2.10.2 *(Segurança OTSU). Dado um adversário de dois estágios* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *contra o esquema de assinatura digital* SS*, associamos a ele o seguinte experimento* $\mathsf{Exp}^{otsu}_{\mathsf{SS},\mathcal{A}}(n)$*:*

$(\mathsf{vk}, \mathsf{dsk}) \leftarrow \mathsf{Gen}(1^n)$

$(\mathsf{m}, estado) \leftarrow \mathcal{A}_1(\mathsf{vk})$

$\sigma \leftarrow \mathsf{Sign}(\mathsf{dsk}, \mathsf{m})$

$(\mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}_2(\mathsf{m}, \sigma, estado)$

Se $\mathsf{Ver}(\mathsf{vk}, \mathsf{m}^*, \sigma^*) = 1$ *e* $(\mathsf{m}^*, \sigma^*) \neq (\mathsf{m}, \sigma)$ *retorna* 1*, senão retorna* 0*.*

Dizemos que um esquema é seguro de acordo com a noção de segurança denominada OTSU se a probabilidade de $\mathsf{Exp}^{otsu}_{\mathsf{SS},\mathcal{A}}(n)$ *retornar* 1 *for uma função desprezível de* n *para todo adversário probabilístico de tempo polinomial* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*.*

2.11 PROVAS DE CONHECIMENTO NULO NÃO-INTERATIVAS

Uma Prova de Conhecimento Nulo é uma técnica utilizada por uma parte $\mathcal{P}$ para provar para outra parte $\mathcal{V}$ que um fato é verdadeiro, sem revelar para $\mathcal{V}$ nada além da veracidade do fato. Agora definiremos a noção de Provas de Conhecimento Nulo Não-Interativas com Segurança Adaptativa. Essa definição é de [49].

Definição 2.11.1 *(Provas de Conhecimento Nulo Não-Interativas com Segurança Adaptativa). Um par de algoritmos probabilísticos de tempo polinomial* $(\mathcal{P}, \mathcal{V})$ *é um sistema de provas de conhecimento nulo não-interativo e adaptativo para a linguagem* $L \in \mathrm{NP}$ *se existir um polinômio* poly *tal que:*

Completude: *Para todo* $x \in L \cap \{0,1\}^n$ *e todo testemunho* w *para* x*,*

$$\Pr[r \leftarrow \{0,1\}^{\mathrm{poly}(n)}; \pi \leftarrow \mathcal{P}(r, x, w) : \mathcal{V}(r, x, \pi) = 1] = 1$$

Corretude: *Para todo algoritmo (possivelmente ilimitado)* $\mathcal{P}^*$*, a seguinte expressão deve ser desprezível em* n*:*

$$\Pr[r \leftarrow \{0,1\}^{\mathrm{poly}(n)}; (x, \pi) \leftarrow \mathcal{P}^*(r) : \mathcal{V}(r, x, \pi) = 1 \wedge x \in \{0,1\}^n \setminus L].$$

Conhecimento Nulo: *Sejam* $(\mathcal{S}_1, \mathcal{S}_2)$ *e* $(\mathcal{A}_1, \mathcal{A}_2)$ *pares de algoritmos em dois estágios. Considere os seguintes experimentos onde* $x \in L \cap \{0,1\}^n$*:*

Game $\mathrm{ZK}_{\mathrm{real}}$	Game $\mathrm{ZK}_{\mathrm{sim}}$
$r \leftarrow \{0,1\}^{\mathrm{poly}(n)}$	$(r, estado') \leftarrow \mathcal{S}_1(1^n)$
$(x, w, estado) \leftarrow \mathcal{A}_1(r)$	$(x, w, estado) \leftarrow \mathcal{A}_1(r)$
$\pi \leftarrow \mathcal{P}(r, x, w)$	$\pi \leftarrow \mathcal{S}_2(x, estado')$
$b \leftarrow \mathcal{A}_2(r, x, \pi, estado)$	$b \leftarrow \mathcal{A}_2(r, x, \pi, estado)$

Deve existir um simulador probabilístico e de tempo polinomial $(\mathcal{S}_1, \mathcal{S}_2)$ *tal que para qualquer algoritmo probabilístico e de tempo polinomial* $(\mathcal{A}_1, \mathcal{A}_2)$ *a seguinte expressão deve ser desprezível em* n*:*

$$|\Pr_{\mathrm{ZK}_{\mathrm{real}}}[\mathcal{A}_2 \text{ gera } b = 0] - \Pr_{\mathrm{ZK}_{\mathrm{sim}}}[\mathcal{A}_2 \text{ gera } b = 0]|.$$

2.12 CRIPTOSISTEMA DDN

Nessa seção apresentamos brevemente o criptosistema DDN que é construido da seguinte maneira. Seja $\Pi = (\mathsf{CPA.Gen}, \mathsf{CPA.Enc}, \mathsf{CPA.Dec})$ um criptosistema de chave pública IND-CPA seguro e $\Sigma = (\mathsf{SS.Gen}, \mathsf{SS.Sign}, \mathsf{SS.Ver})$ um esquema de assinatura digital OTSU seguro. Por simplicidade assumimos

que o tamanho da assinatura de Σ é n onde n é o parâmetro de segurança. Então construimos outro criptosistema de chave pública da seguinte forma.

Geração de Chaves: Para o parâmetro de segurança n, rodamos $2n$ vezes $\mathsf{CPA.Gen}(1^n)$ para obter

$$\mathsf{sk} = (\mathsf{sk}_{0,1}, \ldots, \mathsf{sk}_{0,n}, \mathsf{sk}_{1,1}, \ldots, \mathsf{sk}_{1,n})$$

e

$$\mathsf{pk}' = (\mathsf{pk}_{0,1}, \ldots, \mathsf{pk}_{0,n}, \mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,n}).$$

Escolhemos um número aleatório r (que será usado pelo emissor para gerar uma prova de conhecimento nulo não-interativa). A chave secreta é sk e a chave pública é $\mathsf{pk} = (\mathsf{pk}', r)$.

Ciframento: Para cifrar uma mensagem m, rodamos $\mathsf{SS.Gen}(1^m)$ para obter a chave de verificação vk e chave de assinatura dsk. Seja vk_i o i-ésimo bit de vk. Então executamos $\mathsf{c}_i \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}_{vk_i,i}, \mathsf{m})$ para $1 \leq i \leq n$. Também geramos uma prova de conhecimento nulo não-interativa π que garanta que as mensagens cifradas em todos c_i são idênticas.[1] Finalmente executamos $\sigma \leftarrow \mathsf{SS.Sign}((\mathsf{c}_1, \ldots, \mathsf{c}_n, \pi), \mathsf{dsk})$. O texto cifrado é $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_n, \pi, \mathsf{vk}, \sigma)$.

Deciframento: Para decifrar um texto cifrado $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_n, \pi, \mathsf{vk}, \sigma)$ rodamos $\mathsf{SS.Ver}((\mathsf{c}_1, \ldots, \mathsf{c}_n, \pi), \sigma, \mathsf{vk})$ para checar a validade da assinatura. Também testamos a validade da prova π. Se alguma das provas for inválida, geramos a saída $\perp$. Caso contrário, executamos $\mathsf{m} \leftarrow \mathsf{CPA.Dec}(\mathsf{c}_1, \mathsf{sk}_{vk_1,1})$.

[1] Como isso é uma linguagem NP, podemos gerar uma prova de conhecimento nulo não-interativa.

3 OBLIVIOUS TRANSFER BASEADO NAS HIPÓTESES DE MCELIECE

Nesse capítulo focamos no 1-2 Oblivious Transfer (OT) e construimos um OT baseado nas duas hipóteses de McEliece (Seção 2.7): (1) dificuldade do Problema da Decodificação de Síndrome (que é NP-completo [4] e equivalente ao problema LPN [64]); (2) indistinguibilidade da matriz "embaralha" do criptosistema de McEliece [54] de uma matriz aleatória. Consideramos somente adversários estáticos, isto é, assumimos que as partes são corrompidas antes do início da execução do protocolo. Esse capítulo é parte de um trabalho [28] realizado conjuntamente com Jeroen van de Graaf, Jörn Müller-Quade (Universität Karlsruhe) e Anderson C. A. Nascimento.

Até onde sabemos esse é o primeiro protocolo de Oblivious Transfer baseado somente nas hipóteses de McEliece. Além disso esse protocolo é, juntamente com [52], o primeiro protocolo de OT computacionalmente seguro para o qual não se conhece um algoritmo quântico que quebre a segurança do protocolo. No entanto para obter um protocolo de complexidade equivalente Kobara et al. [52] utiliza hipóteses adicionais: a hipótese do oráculo aleatório e a hipótese conhecida como Permuted Kernels. Além disso eles usam o esquema de provas de conhecimento nulo do Shamir [68], o que não é necessário em nossa construção. Nosso procotolo é incondicionalmente seguro para o Bob e computacionalmente seguro para Alice.

3.1 PROTOCOLO DE OT SEGURO CONTRA ADVERSÁRIOS PASSIVOS

Como primeiro passo apresentamos um protocolo de OT seguro contra adversários honestos-mas-curiosos (também chamados de passivos). Esse tipo de adversário segue as instruções do protocolo corretamente, mas tenta obter informações adicionais a partir dos dados recebidos durante a execução do protocolo. Tal protocolo será modificado nas seções posteriores para obter um protocolo de OT seguro contra adversários maliciosos.

Primeiro esboçamos a intuição por trás desse protocolo. Nossa construção é baseada no paradigma apresentado em [3]. Bob envia para Alice um objeto que é ou uma chave pública do criptosistema de McEliece ou uma chave randomizada de mesmo tamanho para a qual o problema da decodificação de síndrome é difícil. Para tornar aleatória a chave pública, usamos um ou-exclusivo bit-a-bit com uma matriz

aleatória. Alice computa o ou-exclusivo bit-a-bit do objeto recebido com a mesma matriz aleatória para assim obter a segunda "chave". Ela cifra b_0 e b_1 com as duas chaves, respectivamente, e envia os textos cifrados para Bob. O protocolo é seguro para Bob porque Alice não pode distinguir uma chave pública de uma matriz aleatória. O protocolo é seguro para Alice porque Bob não consegue decifrar um texto cifrado com uma matriz aleatória. E por fim, o protocolo é correto pois Bob sempre consegue decifrar o bit que foi cifrado usando a chave pública.

A entrada de Alice são os bits b_0 e b_1 e a entrada de Bob é o bit c. Denotamos o peso de Hamming de um vetor z por $w_H(z)$.

Protocolo 3.1.1

1. Alice escolhe uma matriz binária aleatória Q de tamanho $k \times n$ e a envia para Bob.

2. Bob gera a chave secreta (S, G, T) seguindo os procedimentos do criptosistema de McEliece, define $P_c = SGT$ e $P_{\overline{c}} = P_c \oplus Q$ e envia P_0, t para Alice.

3. Alice computa $P_1 = P_0 \oplus Q$ e então cifra duas cadeias de bits aleatórias $r_0, r_1 \leftarrow \{0,1\}^k$ usando P_0 e P_1 respectivamente. Isto é, para $i = 0, 1 : y_i = r_i P_i \oplus e_i$, onde $e_i \in \{0,1\}^n$, $w_H(e_i) = t$. Ela também computa para $i = 0, 1$:$h_i \leftarrow \{0,1\}^k$ e então cifra b_0 e b_1 da seguinte forma: para $i = 0, 1 : \hat{b}_i = b_i \oplus \langle r_i, h_i \rangle$. Alice envia para $i = 0, 1 : y_i, h_i, \hat{b}_i$ para Bob.

4. Bob decifra r_c e computa $b_c = \hat{b}_c \oplus \langle r_c, h_c \rangle$.

O próximo teorema estabelece formalmente a segurança do protocolo acima.

Teorema 3.1.2 *Assumindo as hipóteses 2.7.1 e 2.7.2, o protocolo 3.1.1 é correto e seguro tanto para Alice como para Bob passivos de acordo com a definição 2.2.1.*

Dado que nos ataques passivos os participantes sempre seguem o protocolo, argumentamos que as propriedades listadas na definição 2.2.1 são satisfeitas.

Corretude: Isso segue da observação de que Bob sempre recebe um ciframento válido de r_c e portanto é capaz de calcular b_c no passo 4 do protocolo.

Segurança para Alice: Seja $\widetilde{B}$ qualquer receptor probabilístico de tempo polinomial. Seja c o bit tal que $\hat{b}_{\overline{c}} = b_{\overline{c}} \oplus \langle r_{\overline{c}}, h_{\overline{c}} \rangle$ e $y_{\overline{c}} = r_{\overline{c}}(P_c \oplus Q) \oplus e_{\overline{c}}$. Note que Q é escolhido de forma aleatória e independentemente de P_c, logo do ponto de vista de $\widetilde{B}$ aprender $r_{\overline{c}}$ é equivalente a decodificar um código linear aleatório

com matriz geradora $P_c \oplus Q$. Esse problema é difícil [4]. Foi provado em [36] que $\langle r, h \rangle$ é predicado núcleo-duro (Hardcore Predicate) para qualquer função unidirecional f dado $f(r)$ e h. Portanto, pela hipótese 2.7.2, a distribuição (sobre a aleatoriedade usada por Alice) das execuções usando a aleatoriedade R_B com Alice tendo como entradas b_c and $b_{\overline{c}} = 0$ é computacionalmente indistinguível das execuções usando a aleatoriedade R_B com Alice tendo como entradas b_c and $b_{\overline{c}} = 1$

Segurança para Bob: Isso segue diretamente da hipótese 2.7.1. Alice honesta-mas-curiosa não pode distinguir entre $P = SGT$ e uma matriz aleatória de tamanho $k \times n$. Portanto ela também não pode distinguir entre $P_c = SGT$ e $P_{\overline{c}} = SGT \oplus Q$ para qualquer $c \in \{0, 1\}$. Isso resulta na indistinguibilidade computacional das execuções do protocolo para Alice.

Infelizmente o protocolo 3.1.1 não é seguro contra ataques ativos. Um problema é que dada a matriz aleatória Q, Bob pode obter duas matrizes P', P'' tais que $P' \oplus P'' = Q$ e que ambas sejam matrizes geradoras de códigos com propriedades de decodificação razoavelmente boas. Nesse caso Bob conseguiria decodificar parcialmente ambos b_0 e b_1.

3.2 PROTOCOLO DE OT SEGURO CONTRA ADVERSÁRIOS MALICIOSOS

Para transformar um protocolo seguro contra adversários passivos em um protocolo seguro contra adversários maliciosos podemos utilizar um compilador genérico como o de [35]. No entanto nós apresentaremos um solução direta e mais eficiente. Tal solução é realizada em três passos:

1. Implementamos um OT randomizado no qual Bob é forçado a escolher a chave pública antes e independentemente de Q (caso ele não faça isso Alice detectará com probabilidade $\frac{1}{2}$);
2. Convertemos o OT randomizado em um OT para entradas específicas com as mesmas características de segurança,
3. Reduzimos a probabilidade de que Bob obtenha informação simultaneamente sobre b_0 e b_1.

3.2.1 OT Randomizado com Alta Probabilidade de B Trapaçear

Primeiro implementamos um protocolo que gera dois bits de saída aleatórios a_0, a_1 para Alice e um bit de saída aleatório d e a_d para Bob. Nesse protocolo Alice detecta com probabilidade ao menos $\frac{1}{2} - \epsilon$ se um Bob malicioso escolher a chave pública de forma dependente de Q.

Bob gera duas chaves diferentes do criptosistema de McEliece usando os mesmos procedimentos do protocolo 3.1.1 e usando dois bits aleatórios c_0, c_1. Ele se compromete com P_{0,c_0} e P_{1,c_1}. Então Bob recebe duas matrizes aleatórias Q_0 e Q_1 de Alice, computa $P_{0,\overline{c_0}} = P_{0,c_0} \oplus Q_0$ e $P_{1,\overline{c_1}} = P_{1,c_1} \oplus Q_1$ e envia $P_{0,0}, P_{1,0}, t$ para Alice. Alice escolhe um dos comprometimentos para ser revelado. Bob revela esse comprometimento e Alice verifica se a informação revelada é consistente com os procedimentos honestos do protocolo, se não for Alice encerra sua execução. Por fim, Alice cifra a_0 e a_1 usando as matrizes associadas ao comprometimento que não foi revelado.

Protocolo 3.2.1

1. Bob gera duas chaves secretas de McEliece (S_0, G_0, T_0) e (S_1, G_1, T_1). Ele escolhe $c_0, c_1 \leftarrow \{0,1\}$ e define $P_{0,c_0} = S_0 G_0 T_0$ e $P_{1,c_1} = S_1 G_1 T_1$. Ele se compromete com P_{0,c_0} e P_{1,c_1}.

2. Alice escolhe matrizes aleatórias Q_0 e Q_1 e as envia para Bob.

3. Bob computa $P_{0,\overline{c_0}} = P_{0,c_0} \oplus Q_0$ e $P_{1,\overline{c_1}} = P_{1,c_1} \oplus Q_1$. Ele envia $P_{0,0}, P_{1,0}, t$ para Alice.

4. Alice computa $P_{0,1} = P_{0,0} \oplus Q_0$ e $P_{1,1} = P_{1,0} \oplus Q_1$. Ela escolha o desafio $j \leftarrow \{0,1\}$ e o envia para Bob.

5. Bob revela o comprometimento que fez para $P_{\overline{j},c_{\overline{j}}}$ e define $d = c_j$.

6. Alice checa se $P_{\overline{j},c_{\overline{j}}}$ é igual $P_{\overline{j},0}$ ou $P_{\overline{j},1}$, caso contrário ela encerra o protocolo.

7. Alice cifra duas cadeias de bits aleatórias $r_0, r_1 \leftarrow \{0,1\}^k$ com $P_{j,0}$ e $P_{j,1}$ respectivamente. Isto é, para $i = 0, 1 : y_i = r_i P_{j,i} \oplus e_i$, onde $e_i \in \{0,1\}^n$, $w_H(e_i) = t$. Ela também computa para $i = 0, 1$: $h_i \leftarrow \{0,1\}^k$ e cifra $a_0, a_1 \leftarrow \{0,1\}$ da seguinte forma: para $i = 0, 1 : \hat{a}_i = a_i \oplus \langle r_i, h_i \rangle$. Por fim ela envia para $i = 0, 1 : y_i, h_i, \hat{a}_i$ para Bob.

8. Bob decifra r_d e computa $a_d = \hat{a}_d \oplus \langle r_d, h_d \rangle$. Se Bob encontrar um erro durante a decodificação de r_d, ele gera a saída $a_d = 0$.

Teorema 3.2.2 *Assumindo que o esquema de comprometimento de bits utilizado é seguro e que as hipóteses 2.7.1 e 2.7.2 são válidas, o esquema de OT Randomizado é correto e seguro para Bob contra adversários ativos de acordo com a definição 2.2.1. Além disso a probabilidade de que um Bob malicioso obtenha informação sobre ambos a_0 e a_1 é no máximo $\frac{1}{2} + \epsilon(n)$, onde $\epsilon(n)$ é uma função desprezível.*

Corretude: Um Bob honesto sempre passa na teste de verificação do passo 6 e assim recebe um ciframento válido de r_d, o que lhe permite computar a_d.

Segurança para Alice: Para obter informação simultaneamente sobre a_0 e a_1, Bob deve aprender ambos r_0 e r_1. Os ciframentos de r_0 er_1 só dependem de $P_{j,0}$ e $P_{j,1}$ respectivamente.

Se Bob enviar ambos $P_{0,0}$ e $P_{1,0}$ escolhidos de acordo com o protocolo, então a probabilidade de que ele aprenda ambas entradas de Alice é a mesma do protocolo contra adversários passivos e portanto desprezível. Se Bob escolher ambos $P_{0,0}$ e $P_{1,0}$ de forma maliciosa, então com probabilidade altíssima Alice irá interromper a execução do protocolo no passo 6 e logo Bob não aprenderá nem r_0 nem r_1.

A melhor estratégia para Bob é escolher honestamente uma das matrizes e escolher a outra de forma maliciosa. Dessa forma ele pode trapaçear e decodificar r_0 e r_1 se Alice solicitar a ele que revele o comprometimento relativo a matriz escolhida honestamente. No entanto com probabilidade $\frac{1}{2}$, Alice solicitará a Bob que revele o comprometimento relativo a matriz escolhida maliciosamente. Neste caso Bob só conseguirá passar no teste do passo 6 com probabilidade desprezível, pois o esquema de comprometimento de bits é seguro por hipótese. Portanto a probabilidade de que um Bob malicioso aprenda ambos a_0 e a_1 é no máximo $\frac{1}{2} + \epsilon(n)$, onde $\epsilon(n)$ é uma função desprezível.

Segurança para Bob: O comprometimento realizado para $P_{j,c_j} = P_{j,d}$ não é revelado, portanto a segurança de Bob segue da hipótese 2.7.1 como no protocolo 3.1.1.

Desde o esquema de comprometimento de bits utilizado seja seguro, as possíveis diferenças para o cenário com adversários passivos são as seguintes:

- Alice pode trapaçear enviando uma matriz Q escolhida de alguma forma especial. No entanto, pela hipótese 2.7.1, ele não pode distinguir P_{j,c_j} de uma matriz aleatória, e portanto a escolha de Q não afetará a habilidade dela de aprender d;

- Para algum $i \in \{0, 1\}$, Alice pode usar uma matriz diferente de $P_{j,i}$ para cifrar r_i no passo 7, tendo a esperança de que $i = d$ e assim Bob tenha um problema na decodificação e se queixe (relevando dessa forma a sua escolha). No entanto a última instrução do protocolo, passo 8, bloquea esse tipo de ataque fazendo com que Bob gere a saída "0" nesse caso. Portanto enviar uma síndrome "errada" é equivalente a situação em que Alice fixa sua entrada $a_i = 0$.

Portanto o protocolo é seguro contra Alice maliciosa.

3.2.2 OT para Entradas Específicas

Agora usamos o método de [2] para transformar o protocolo de OT Randomizado da seção anterior em um protocolo ordinário de OT (isto é, para entradas especificadas pelas partes) com as mesmas características de segurança.

Protocolo 3.2.3

1. Bob e Alice executam o protocolo 3.2.1. Alice recebe a_0, a_1 e Bob recebe d, a_d.
2. Bob escolhe c, calcula $e = c \oplus d$ e envia e para Alice.
3. Alice escolhe $b_0, b_1 \in \{0,1\}$, computa $f_0 = b_0 \oplus a_e$ e $f_1 = b_1 \oplus a_{\overline{e}}$ e envia f_0, f_1 para Bob.
4. Bob computa $b_c = f_c \oplus a_d$.

Teorema 3.2.4 *O protocolo 3.2.3 implementa OT com as mesmas características de segurança do protocolo 3.2.1.*

Corretude: $f_c = b_c \oplus a_{c \oplus e} = b_c \oplus a_d$, portanto um Bob honesto pode recuperar b_c já que ele conhece a_d.

Segurança para Alice: $f_{\overline{c}} = b_{\overline{c}} \oplus a_{\overline{c \oplus e}} = b_{\overline{c}} \oplus a_{\overline{d}}$, portanto Bob pode recuperar ambos b_0 e b_1 somente se ele souber ambos a_0 e a_1.

Segurança para Bob: Alice deve descobrir d para poder calcular c, logo a segurança para Bob segue do protocolo 3.2.1.

3.2.3 Reduzindo a probabilidade de Bob Trapaçear

Finalmente nós utilizamos a redução de [22] para minimizar a probabilidade que um Bob malicioso aprenda ambas as entradas de Alice. Nessa redução o protocolo 3.2.3 é executado s vezes em paralelo, onde s é um parâmetro de segurança. As entradas de cada execução são escolhidas de tal maneira que Bob terá que aprender ambos os bits em todas as execuções para que ele possa calcular ambas as entradas de Alice no protocolo 3.2.5.

Protocolo 3.2.5

1. Alice escolhe seus dois bits de entrada para o protocolo de OT, $b_0, b_1 \in \{0,1\}$. Ela também escolhe bits aleatórios $b_{0,1}, \ldots, b_{0,s}, b_{1,1}, \ldots, b_{1,s}$ tais que $b_0 = b_{0,1} \oplus \ldots \oplus b_{0,s}$ e $b_1 = b_{1,1} \oplus \ldots \oplus b_{1,s}$.

2. Bob escolhe $c \in \{0, 1\}$.

3. O protocolo 3.2.3 é executado s vezes, com entradas $b_{0,i}, b_{1,i}$ de Alice e entrada $c_i = c$ de Bob para $i = 1 \ldots s$.

4. Bob computa $b_c = b_{c,1} \oplus b_{c,2} \oplus \ldots \oplus b_{c,s}$.

Teorema 3.2.6 *Assumindo que o protocolo de comprometimento de bits utilizado no protocolo 3.2.1 é seguro e que as hipóteses 2.7.1 e 2.7.2 são válidas, o protocolo 3.2.5 é correto e seguro para Alice e Bob contra adversário maliciosos de acordo com a definição 2.2.1.*

Corretude: Um Bob honesto aprende todos $b_{c,i}$ para $i = 1 \ldots s$ nas s execuções do protocolo 3.2.3 e assim ele pode computar b_c.

Segurança para Alice: Bob deve descobrir ambos os bits em todas as execuções do protocolo 3.2.3 para que ele possa aprender simultaneamente b_0 e b_1. A probabilidade de que um Bob malicioso aprenda ambos os bits em uma execução do protocolo 3.2.3 é no máximo $\frac{1}{2} + \epsilon(n)$, onde $\epsilon(n)$ é uma função desprezível. Existe um n_0 tal que $\epsilon(n) < \frac{1}{4}$ para todo $n > n_0$. Nós podemos escolher $n > n_0$, e então $\beta = \frac{1}{2} + \epsilon(n) < \frac{3}{4}$ e a probabilidade de que um Bob malicioso aprenda ambos b_0 e b_1 é menor do que $(\frac{3}{4})^s$, o que é uma função desprezível de s. Portanto o protocolo é seguro para Alice.

Segurança para Bob: Alice descobre c somente se ela descobrir algum c_i, mas a probabilidade desse evento ocorrer é desprezível, pois a probabilidade de que ela descubra um c_i específico na respectiva execução do protocolo 3.2.3 é desprezível e o número de execuções do protocolo 3.2.3 é somente polinomial.

3.2.4 Protocolo de OT Utilizando BCX

Ao invés de utilizarmos os protocolos 3.2.1, 3.2.3 e 3.2.5 para construir um protocolo de OT seguro contra adversários maliciosos, podemos utilizar uma modificação do protocolo BCX (Bit Commitments with XOR) e obter uma outra construção de protocolo de OT. A complexidade dessa nova construção é similar a anterior. No entanto acreditamos que a nossa generalização do BCX pode ser de interesse independente e a apresentamos aqui.

O protocolo BCX é atribuido a Bennett e Rudich e é descrito em [19]. O esquema pode ser baseado em qualquer protocolo de comprometimento de bits e procede assim: Alice se compromete com um bit b se comprometendo com pares de bits b_{jL} e b_{jR} tais que $b = b_{jL} \oplus b_{jR}$. A vantagem do BCX quando comparado ao comprometimentos de bits tradicionais é que ele permite provar relações lineares entre os

valores para com os quais uma parte se comprometeu sem que seja necessário revelar esses valores. Nós modificamos um pouco o BCX para que ele funcione com inteiros módulo q, como descrito abaixo (v é o parâmetro de segurança do procedimento para provar as relações):

Comprometimento: Para se comprometer com algum valor $b \in \mathbb{Z}_q$, Bob escolhe $b_{jL} \leftarrow \mathbb{Z}_q$ e fixa $b_{jR} = b - b_{jL} (\text{mod } q)$ para $j = 1, \ldots, v$. Bob se compromete com b_{jL} e b_{jR} para $j = 1, \ldots, v$.

Revelação: Para revelar um valor b, Bob revela todos os $2v$ comprometimentos. Alice checa se $b = b_{jL} + b_{jR} (\text{mod } q)$ para todo j e aceita o valor b somente se essa condição for satisfeita.

Provando Relações: Para provar uma relação linear entre u valores $b^1, \ldots, b^u$ sem revelar esses valores, a partes fazem o seguinte:

1. Alice especifica u permutações (cada uma de v elementos).
2. Para $l = 1, \ldots, u$, Bob embaralha os v pares de comprometimentos relativos a b^l usando a permutação especificada por Alice.
3. Para $j = 1, \ldots, v$, Bob avalia as relações lineares usando os valores $(b^1_{jL}, \ldots, b^u_{jL})$ e usando os valores $(b^1_{jR}, \ldots, b^u_{jR})$ e denota os resultados por Rel_{jL} e Rel_{jR} respectivamente.
4. Bob envia Rel_{jL} e Rel_{jR} para Alice para $j = 1, \ldots, v$.
5. Alice checa se a soma Rel_{jL} and Rel_{jR} é a mesma para todos j.
6. Para cada $j = 1, \ldots, v$, Alice solicita que Bob revele ou $(b^1_{jL}, \ldots, b^u_{jL})$ ou $(b^1_{jR}, \ldots, b^u_{jR})$ e checa se o resultado enviado anteriormente por Bob é a avaliação correta da relação usando esses valores. Se não for a correta, Alice encerra a execução do protocolo. Caso contrário, Alice aceita que o resultado de aplicar a relação linear nos valores $(b^1, \ldots, b^u)$ é $Rel_{1L} + Rel_{1R} (\text{mod } q)$.

Copiando os Comprometimentos: Cada comprometimento pode ser usado somente uma vez para provar relações lineares. Se Bob estiver comprometido com um valor b, ele pode obter uma nova cópia desse comprometimento da seguinte forma:

1. Bob cria $3v$ pares de comprometimentos tais que a soma módulo q de cada par seja b.
2. Alice divide esses $3v$ pares em 3 subconjuntos de cardinalidade v que ela denota por b^0, b^1, b^2. Ela solicita que Bob prove que $b^0 = b$. Se ele tiver sucesso, b e b^0 não podem mais ser utilizados, mas Alice se convence que $b^1 = b^2 = b$. Portanto Bob agora tem duas instâncias válidas de comprometimentos para o valor b: b^1 e b^2.

Note que um Bob desonesto que tente provar uma relação falsa entre valores será flagrado com probabilidade altíssima no parâmetro de segurança v. Note também que somente um dos comprometimentos de cada par é revelado no procedimento de provar relações lineares, portanto os valores $b^1, \ldots, b^u$ não são revelados pois eles são a soma dos valores de um mesmo par.

Usando as idéias acima podemos construir um protocolo de OT baseado no criptosistema de McEliece e seguro contra adversários maliciosos. Tal protocolo é descrito abaixo. Interpretamos cada matriz de dimensão $k \times n$ como uma cadeia de bits de tamanho nk e usamos a variante dos comprometimentos de Bennett-Rudich acima para computar as operações módulo $q = 2^{nk}$.

Protocolo 3.2.7

1. Bob gera uma chave secreta (S, G, T) seguindo os procedimentos do algoritmo de McEliece e se compromete com a cadeia de bits que representa a chave pública SGT usando o BCX. Denotamos os comprometimentos por $d = (d_{1L}, d_{1R}, \ldots, d_{vL}, d_{vR})$ e definimos $d_j = (d_{jL}, d_{jR})$.

2. Alice escolhe uma matriz aleatória Q de dimensões $k \times n$ e a envia para Bob.

3. Bob define $P_c = SGT$ e $P_{\overline{c}} = P_c \oplus Q$ e se compromete com as cadeias de bits que representam as matrizes P_0 e P_1. Denotamos os comprometimentos por $f = (f_{1L}, f_{1R}, \ldots, f_{vL}, f_{vR})$ e $g = (g_{1L}, g_{1R}, \ldots, g_{vL}, g_{vR})$ respectivamente. Bob envia P_0, t para Alice. Para $j = 1, \ldots, v$, ele também envia os pares $f_j = (f_{jL}, f_{jR})$ e $g_j = (g_{jL}, g_{jR})$ para Alice, escolhendo aleatoriamente se f_j ou g_j é enviado primeiro.

4. Alice computa $P_1 = P_0 \oplus Q$, escolhe aleatoriamente um desafio V de v bits e uma permutação Π de v elementos. Alice envia V e Π para Bob.

5. Denotamos por V_j o j-ésimo bit de V. Para $j = 1, \ldots, v$, Bob faz o seguinte:

   Se $V_j = 0$, então Bob revela os comprometimentos f_j e g_j. Alice checa se um deles equivale a matriz P_0 e o outro a matriz P_1. Em caso negativo, ela encerra a execução do protocolo.

   Se $V_j = 1$, então Bob prova usando o BCX (isto é, enviando as somas da direita e da esquerda e revelando os valores dos comprometimentos da direita ou da esquerda conforme Alice escolher) que $d_{\Pi(j)} = f_j$ ou que $d_{\Pi(j)} = g_j$. Se Bob não conseguir provar isso, Alice encerra a execução do protocolo.

6. Alice cifra duas valores aleatórios $r_0, r_1 \leftarrow \{0,1\}^k$ usando P_0 e P_1, respectivamente. Isto é, para $i = 0, 1 : y_i = r_i P_i \oplus e_i$, onde $e_i \in \{0,1\}^n$, $w_H(e_i) = t$. Ela também computa para $i = 0, 1 : h_i \leftarrow$

$\{0,1\}^k$ e então cifra b_0 e b_1 da seguinte maneira: para $i = 0, 1 : \hat{b}_i = b_i \oplus \langle r_i, h_i \rangle$. Por fim, ela envia para $i = 0, 1 : y_i, h_i, \hat{b}_i$ para Bob.

7. Bob decifra r_c e computa $b_c = \hat{b}_c \oplus \langle r_c, h_c \rangle$. Se Bob encontrar um erro durante a decodificação de r_c, ele gera a saída $b_c = 0$.

Teorema 3.2.8 *Assumindo que o esquema de comprometimento de bits é seguro e que as hipóteses 2.7.1 e 2.7.2 são válidas, o protocolo 3.2.7 implementa um OT que é correto e seguro para Alice e Bob contra adversários maliciosos conforme a definição 2.2.1.*

Corretude: Um Bob honesto sempre passa no teste do passo 5 e recebe um ciframento válido de r_c, que lhe permite calcular b_c.

Segurança para Alice: Para obter informação simultaneamente sobre b_0 e b_1, Bob deve aprender r_0 e r_1. Os ciframentos de r_0 e r_1 só dependem de P_0 e P_1 respectivamente.

Se um Bob desonesto não escolher P_0 e P_1 de acordo com os procedimentos do protocolo, ele só terá sucesso no teste do passo 5 com probabilidade desprezível no parâmetro de segurança v. Se Bob seguir os procedimentos do protocolo para escolher P_0 e P_1, a segurança para Alice segue do caso passivo.

Segurança para Bob: Se Alice escolher $V_j = 0$ no desafio, ele descobre f_j e g_j e pode checar se elas são iguais a P_0 e P_1. No entanto, como f_j e g_j são enviados em uma ordem aleatória, ela não aprende nada sobre o valor c. Se Alice escolher $V_j = 1$ no desafio, ela descobre que $d_{\Pi(j)} = f_j$ ou $d_{\Pi(j)} = g_j$ e também qual dos dois casos ocorreu. Mas como f_j e g_j são enviados em uma ordem aleatória e não são revelados, ela não pode comparar a que é igual a $d_{\Pi(j)}$ com as matrizes P_0 e P_1 para descobrir c. Portanto a segurança para Bob segue da hipótese 2.7.1 como no protocolo 3.1.1.

Desde o esquema de comprometimento de bits seja seguro, as possíveis diferenças em relação ao caso passivo são as seguintes:

- Alice pode trapaçear enviando uma matriz Q escolhida de forma especial. No entanto, pela hipótese 2.7.1, ela não pode distinguir P_c de uma matriz aleatória, e portanto a escolha de Q não afeta a habilidade dela de aprender c.

- Para algum $i \in \{0,1\}$, Alice pode usar uma matriz diferente de P_i para cifrar r_i no passo 6, tendo a esperança de que $i = c$ e assim Bob tenha um problema na decodificação e se queixe (relevando dessa forma a sua escolha). No entanto a última instrução do protocolo, passo 7, bloquea esse tipo

de ataque fazendo com que Bob gere a saída "0" nesse caso. Portanto enviar uma síndrome "errada" é equivalente a situação em que Alice fixa sua entrada $b_i = 0$.

Portanto o protocolo é seguro contra Alice.

3.3 OT BASEADO NO CRIPTOSISTEMA DE NIEDERREITER

Abaixo descrevemos uma variante do protocolo de OT que utiliza o criptosistema de Niederreiter. O protocolo contra adversários passivos pode ser implementado da seguinte forma.

Protocolo 3.3.1

1. Alice escolhe uma matriz aleatória Q de dimensões $(n-k) \times n$ e a envia para Bob.
2. Bob gera a chave secreta $(\mathbf{M}, \mathbf{P}, \mathbf{D}_{\mathcal{G}})$ seguindo os procedimentos do algoritmo de Niederreiter, define $P_c = MHP$ e $P_{\overline{c}} = P_c \oplus Q$ e envia P_0, t para Alice.
3. Alice computa $P_1 = P_0 \oplus Q$, cifra dois valores aleatórios $r_0, r_1 \leftarrow \{0,1\}^n$ de peso de Hamming t usando P_0 e P_1 respectivamente. Isto é, para $i = 0,1 : s_i = P_i r_i^T$. Ela também escolhe para $i = 0, 1$: $h_i \leftarrow \{0,1\}^n$ e então cifra b_0 e b_1 da seguinte forma: para $i = 0,1 : \hat{b}_i = b_i \oplus \langle r_i, h_i \rangle$. Por fim, Alice envia para $i = 0,1 : s_i, h_i, \hat{b}_i$ para Bob.
4. Bob decifra r_c e computa $b_c = \hat{b}_c \oplus \langle r_c, h_c \rangle$.

Teorema 3.3.2 *Assumindo que as hipóteses 2.7.1 e 2.7.2 são válidas, o protocol 3.3.1 é correto e seguro para Alice e Bob contra adversários passivos de acordo com a definição 2.2.1.*

Dado que no caso passivo as partes seguem o protocolo, nós argumentamos que as propriedades listadas na definição 2.2.1 são atendidas.

Corretude: Como no protocolo que utiliza o criptosistema de McEliece, Bob sempre pode decifrar r_c e portanto computar b_c no passo 4.

Segurança para Alice:

Seja $\widetilde{B}$ um receptor desonesto probabilístico e de tempo polinomial.

Seja c o bit tal que $\hat{b}_{\overline{c}} = b_{\overline{c}} \oplus \langle r_{\overline{c}}, h_{\overline{c}} \rangle$ e $s_{\overline{c}} = (P_c \oplus Q) r_{\overline{c}}^T$. Note que Q é escolhido de forma aleatória e independente de P_c. Portanto do ponto de vista de $\widetilde{B}$, aprender $r_{\overline{c}}$ é equivalente a decodificar um código

linear aleatório com matriz de paridade $P_c \oplus Q$. Pelo mesmo argumento da versão utilizando McEliece, $\langle r, h \rangle$ é um predicado núcleo-duro (Hardcore Predicate) para esta função e a segurança para Alice é obtida baseada na hipótese 2.7.2.

Segurança para Bob: Isto segue diretamente da hipótese 2.7.1 e da equivalência da segurança dos criptosistema de McEliece e de Niederreiter [23]. Alice honesta-mas-curiosa não consegue distinguir entre MHP e uma matriz aleatória de mesmas dimensões. Portanto, ela também não consegue distinguir entre $P_c = MHP$ e $P_{\overline{c}} = MHP \oplus Q$ para qualquer $c \in \{0, 1\}$. Isso resulta na indistinguibilidade computacional das execuções do protocolo para Alice.

Para obter um protocolo seguro contra partes maliciosas, podemos utilizar os mesmos métodos usados na versão do protocolo que utiliza o criptosistema de McEliece.

4 CRIPTOSISTEMA DE CHAVE PÚBLICA IND-CCA2 SEGURO BASEADO NAS HIPÓTESES DE MCELIECE SEM ORÁCULO ALEATÓRIO

Nesse capítulo apresentamos um criptosistema de chave pública IND-CCA2 seguro baseado nas hipóteses de McEliece e sem uso de oráculos aleatórios. Tal criptosistema foi proposto em um trabalho [27] realizado conjuntamente com Jörn Müller-Quade e Anderson C. A. Nascimento. Esse criptosistema representa uma alternativa caso os computadores quânticos se tornem práticos, pois, ao contrário do problema da fatoração e do logaritmo discreto, não se conhece nenhum algoritmo quântico eficiente que quebre as hipóteses de McEliece.

4.1 CRIPTOSISTEMAS DE CHAVE PÚBLICA K-REPETIDOS

4.1.1 Definições

Nesta seção nós definimos o conceito de Criptosistema de Chave Pública k-repetido.

Definição 4.1.1 *(Criptosistema de Chave Pública k-repetido). Para um criptosistema de chave pública* PKE (Gen, Enc, Dec)*, nós definimos o criptosistema de chave pública k-repetido* (PKE_k) *como um conjunto de três algoritmos* (Gen_k, Enc_k, Dec_k) *tais que:*

- Gen_k *é um algoritmo probabilístico de geração de chaves que executa em tempo polinomial. Esse algoritmo tem como entrada o parâmetro de segurança 1^n e executa k vezes o algoritmo de geração de chaves do criptosistema* PKE *obtendo as chaves públicas* $(\mathsf{pk}_1, \ldots, \mathsf{pk}_k)$ *e chaves secretas* $(\mathsf{sk}_1, \ldots, \mathsf{sk}_k)$*. Então* Gen_k *gera como saída a chave pública* $pk = (\mathsf{pk}_1, \ldots, \mathsf{pk}_k)$ *e a chave secreta* $sk = (\mathsf{sk}_1, \ldots, \mathsf{sk}_k)$.

- Enc_k *é um algoritmo (que pode ser probabilístico) de ciframento de mensagens que executa em tempo polinomial. Esse algoritmo recebe como entrada a chave pública* $pk = (\mathsf{pk}_1, \ldots, \mathsf{pk}_k)$ *e uma mensagem* $\mathsf{m} \in \mathcal{M}$ *e produz como saída um texto cifrado* $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_k) = (\mathsf{Enc}(\mathsf{pk}_1, \mathsf{m}), \ldots, \mathsf{Enc}(\mathsf{pk}_k, \mathsf{m}))$.

– Dec_k *é um algoritmo determinístico de decifragem em tempo polinomial. Esse algoritmo recebe como entrada a chave secreta* $sk = (\mathsf{sk}_1, \ldots, \mathsf{sk}_k)$ *e um texto cifrado* $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_k)$*. Ele produz como saída a mensagem* m *se* $\mathsf{Dec}(\mathsf{sk}_1, \mathsf{c}_1), \ldots, \mathsf{Dec}(\mathsf{sk}_k, \mathsf{c}_k)$ *forem todos iguais a algum* $\mathsf{m} \in \mathcal{M}$*. Caso contrário, ele produz como saída um símbolo de erro* $\perp$*.*

– *(Corretude) Para qualquer* k *pares de chaves públicas e privadas geradas por* Gen_k *e qualquer mensagem* $\mathsf{m} \in \mathcal{M}$ *temos que* $\mathsf{Dec}_k(\mathsf{sk}, \mathsf{Enc}_k(\mathsf{pk}, \mathsf{m})) \neq \mathsf{m}$ *tem probabilidade desprezível sobre as aleatoriedades utilizadas por* Gen_k *e* Enc_k*.*

Nós também definimos os requisitos de segurança que o criptosistema de chave pública k-repetido utilizado nas próximas seções deve atender.

Definição 4.1.2 *(Segurança IND-CPA para Criptosistemas de Chave Pública* k*-repetidos). Dizemos que* PKE_k *(construído a partir de um* PKE *IND-CPA seguro) é seguro se* PKE_k *for IND-CPA seguro.*

Definição 4.1.3 *(Verificação em Criptosistemas de Chave Pública* k*-repetidos). Dizemos que um* PKE_k *é verificável se dado um texto cifrado* $\mathsf{c} \in \mathcal{C}$*, a chave pública* $pk = (\mathsf{pk}_1, \ldots, \mathsf{pk}_k)$ *e qualquer* sk_i *para* $i \in \{1, \ldots, k\}$*, for possível verificar se* c *é um texto cifrado válido.*

4.1.2 Segurança IND-CCA2 a partir de um Criptosistema de Chave Pública k-repetido

Nesta subseção descrevemos o criptosistema de chave pública IND-CCA2 seguro (PKE_{cca2}) e provamos a sua segurança. Para isso assumimos a existência de um esquema de assinatura digital que seja OTSU seguro e de um PKE_k que seja seguro e verificável.

Geração de Chaves: Gen_{cca2} é um algoritmo probabilístico de geração de chaves que executa em tempo polinomial. Esse algoritmo tem como entrada o parâmetro de segurança 1^n. Gen_{cca2} procede da seguinte maneira:

1. Executa $2k$ vezes o algoritmo de geração de chaves de PKE obtendo as chaves públicas $(\mathsf{pk}_1^0, \mathsf{pk}_1^1, \ldots, \mathsf{pk}_k^0, \mathsf{pk}_k^1)$ e as chaves secretas $(\mathsf{sk}_1^0, \mathsf{sk}_1^1, \ldots, \mathsf{sk}_k^0, \mathsf{sk}_k^1)$.
2. Executa o algoritmo de geração de chaves do esquema de assinatura digital obtendo uma chave de assinatura dsk^* e uma chave de verificação vk^*. Denotamos por vk_i^* o i-ésimo bit de vk^*.
3. Gera como saídas a chave pública $pk = (\mathsf{pk}_1^0, \mathsf{pk}_1^1, \ldots, \mathsf{pk}_k^0, \mathsf{pk}_k^1)$ e a chave secreta $sk = (\mathsf{vk}^*, \mathsf{sk}_1^{1-\mathsf{vk}_1^*}, \ldots, \mathsf{sk}_k^{1-\mathsf{vk}_k^*})$.

Ciframento: Enc_{cca2} é um algoritmo (que pode ser probabilístico) de ciframento de mensagens que executa em tempo polinomial. Esse algoritmo recebe como entrada a chave pública $pk = (\mathsf{pk}_1^0, \mathsf{pk}_1^1, \ldots, \mathsf{pk}_k^0, \mathsf{pk}_k^1)$ e uma mensagem $\mathsf{m} \in \mathcal{M}$ e procede da seguinte maneira:

1. Executa o algoritmo de geração de chaves do esquema de assinatura digital obtendo uma chave de assinatura dsk e uma chave de verificação vk. Denotamos por vk_i o i-ésimo bit de vk.
2. Computa $\mathsf{c}_i = \mathsf{Enc}(\mathsf{pk}_i^{\mathsf{vk}_i}, \mathsf{m})$ para $i \in \{1, \ldots, k\}$.
3. Computa a assinatura $\sigma = \mathsf{Sign}(\mathsf{dsk}, (\mathsf{c}_1, \ldots, \mathsf{c}_k))$.
4. Gera como saída o texto cifrado $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_k, \mathsf{vk}, \sigma)$.

Deciframento: Dec_{cca2} é um algoritmo determinístico de deciframento em tempo polinomial. Esse algoritmo recebe como entrada a chave secreta $sk = (\mathsf{vk}^*, \mathsf{sk}_1^{1-\mathsf{vk}_1^*}, \ldots, \mathsf{sk}_k^{1-\mathsf{vk}_k^*})$ e um texto cifrado $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_k, \mathsf{vk}, \sigma)$ e procede da seguinte maneira:

1. Se $\mathsf{vk} = \mathsf{vk}^*$ ou $\mathsf{Ver}(\mathsf{vk}, (\mathsf{c}_1, \ldots, \mathsf{c}_k), \sigma) = 0$, produz como saída $\perp$ e termina a execução.
2. Para algum $i \in \{1, \ldots, k\}$ tal que $\mathsf{vk}_i \neq \mathsf{vk}_i^*$, computa $\mathsf{m} = \mathsf{Dec}(\mathsf{sk}^{\mathsf{vk}_i}, \mathsf{c}_i)$.
3. Verifica se $\mathsf{c}_i = \mathsf{Enc}(\mathsf{pk}_i^{\mathsf{vk}_i}, \mathsf{m})$ para todo $i \in \{1, \ldots, k\}$. Se essa condição for satisfeita, gera como saída m. Caso contrário, produz como saída $\perp$.

A probabilidade de que $\mathsf{Dec}_{cca2}(\mathsf{sk}, \mathsf{Enc}_{cca2}(\mathsf{pk}, \mathsf{m})) \neq \mathsf{m}$ é a mesma probabilidade de que $\mathsf{vk} = \mathsf{vk}^*$, mas essa probabilidade é desprezível pois o esquema de assinatura digital é OTSU seguro.

Como em [65], podemos utilizar uma função unidirecional de hash universal (universal one-way hash function) nas chaves de verificação (como em [26]) e usar $k = n^\epsilon$ para uma constante $0 < \epsilon < 1$. Para facilitar a apresentação, não utilizamos esse método na descrição do nosso esquema.

Teorema 4.1.4 *Dado um esquema de Assinatura Digital OTSU seguro* SS *e um criptosistema de chave pública* k*-repetido* (PKE_k) *seguro e verificável, o criptosistema de chave pública* PKE_{cca2} *é IND-CCA2 seguro.*

A nossa prova segue de perto a prova de [65]. Seja $\mathcal{A}$ o adversário contra a segurança IND-CCA2.

Denotamos por Forge o evento de que para alguma consulta de $\mathcal{A}$ ao oráculo de deciframento tenhamos que $\mathsf{Ver}(\mathsf{vk}, (\mathsf{c}_1, \ldots, \mathsf{c}_k), \sigma) = 1$ e $\mathsf{vk} = \mathsf{vk}^*$. A validade do teorema segue dos dois lemas abaixo.

Lema 4.1.5 $\Pr[\mathsf{Forge}]$ *é desprezível.*

Assuma que para um adversário probabilístico de tempo polinomial $\mathcal{A}$ contra PKE_{cca2} a probabilidade do evento Forge seja não-desprezível. Então construimos um adversário $\mathcal{A}'$ que forja uma assinatura com a mesma probabilidade. $\mathcal{A}'$ simula a interação IND-CCA2 para $\mathcal{A}$ da seguinte forma:

Geração de Chaves: $\mathcal{A}'$ invoca o oráculo de geração de chaves do esquema de assinatura digital e obtêm vk^*. Ele também executa $2k$ vezes o algoritmo de geração de chaves de PKE obtendo as chaves públicas $(\mathsf{pk}_1^0, \mathsf{pk}_1^1, \ldots, \mathsf{pk}_k^0, \mathsf{pk}_k^1)$ e as chaves privadas $(\mathsf{sk}_1^0, \mathsf{sk}_1^1, \ldots, \mathsf{sk}_k^0, \mathsf{sk}_k^1)$ e usa vk^* para formar a chave secreta do PKE_{cca2}. Ele envia a chave pública para $\mathcal{A}$.

Consultas ao Oráculo de Deciframento: Sempre que $\mathcal{A}$ fizer uma consulta ao oráculo de deciframento, $\mathcal{A}'$ procede da seguinte forma:

1. Se $\mathsf{vk} = \mathsf{vk}^*$ e $\mathsf{Ver}(\mathsf{vk}, (\mathsf{c}_1, \ldots, \mathsf{c}_k), \sigma) = 1$, $\mathcal{A}'$ produz como saída a assinatura forjada $((\mathsf{c}_1, \ldots, \mathsf{c}_k), \sigma)$ e interrompe sua execução.
2. Caso contrário, $\mathcal{A}'$ decifra o texto cifrado usando os procedimentos de PKE_{cca2}.

Oráculo de Desafio: Quando $\mathcal{A}$ fizer a consulta ao oráculo de desafio com duas mensagens $\mathsf{m}_0, \mathsf{m}_1 \in \mathcal{M}$ tais que $|\mathsf{m}_0| = |\mathsf{m}_1|$, $\mathcal{A}'$ procede da seguinte maneira:

1. Escolhe aleatoriamente $b \in \{0, 1\}$.
2. Cifra a mensagem m_b utilizando os procedimentos de PKE_{cca2}. Isto é possível porque $\mathcal{A}'$ pode requerer ao seu oráculo de assinatura que ele assine uma mensagem, e então $\mathcal{A}'$ pede que o oráculo assine a mensagem $(\mathsf{c}_1, \ldots, \mathsf{c}_k)$ obtida durante o processo de ciframento.

A simulação é perfeita se o evento Forge não ocorrer, portanto a probabilidade de que $\mathcal{A}'$ quebre a segurança do esquema de assinatura digital é exatamente Pr[Forge]. Como o esquema de assinatura digital é OTSU seguro por hipótese, segue que Pr[Forge] é desprezível para todo adversário probabilístico de tempo polinomial $\mathcal{A}$ contra PKE_{cca2}.

Lema 4.1.6 *Se o evento* Forge *não ocorrer, a vantagem de qualquer adversário probabilístico de tempo polinomial $\mathcal{A}$ contra* PKE_{cca2}*,*

$$|\Pr[\overline{\mathsf{Forge}} \wedge \mathsf{Exp}^{cca2}_{\mathsf{PKE}_{cca2},\mathcal{A}}(n) = 1] - \tfrac{1}{2}|,$$

é desprezível

Assuma que para um adversário probabilístico de tempo polinomial $\mathcal{A}$ contra PKE_{cca2}, nós tenhamos que $|\Pr[\mathsf{Exp}^{cca2}_{\mathsf{PKE}_{cca2},\mathcal{A}}(n) = 1 \wedge \overline{\mathsf{Forge}}] - \frac{1}{2}|$ é não-desprezível. Então construimos um adversário $\mathcal{A}'$ que quebra a segurança IND-CPA de PKE_k. $\mathcal{A}'$ simula a interação IND-CCA2 para $\mathcal{A}$ da seguinte forma:

Geração de Chaves: $\mathcal{A}'$ recebe como entrada a chave pública $(\mathsf{pk}_1, \ldots, \mathsf{pk}_k)$ do PKE_k. $\mathcal{A}'$ procede da seguinte maneira:

1. Executa o algoritmo de geração de chaves do esquema de assinatura digital e obtêm a chave de verificação vk^* e a chave de assinatura dsk^*.
2. Fixa $\mathsf{pk}_i^{\mathsf{vk}_i^*} = \mathsf{pk}_i$ para $i \in \{1, \ldots, k\}$.
3. Executa k vezes o algoritmo de geração de chaves de PKE obtendo as chaves públicas $(\mathsf{pk}_1^{1-\mathsf{vk}_1^*}, \ldots, \mathsf{pk}_k^{1-\mathsf{vk}_k^*})$ e as chaves privadas $(\mathsf{sk}_1^{1-\mathsf{vk}_1^*}, \ldots, \mathsf{sk}_k^{1-\mathsf{vk}_k^*})$.
4. Fixa a chave pública como $pk = (\mathsf{pk}_1^0, \mathsf{pk}_1^1, \ldots, \mathsf{pk}_k^0, \mathsf{pk}_k^1)$ e chave privada como $sk = (\mathsf{vk}^*, \mathsf{sk}_1^{1-\mathsf{vk}_1^*}, \ldots, \mathsf{sk}_k^{1-\mathsf{vk}_k^*})$.
5. Envia a chave pública para $\mathcal{A}$.

Consultas ao Oráculo de Deciframento: Sempre que $\mathcal{A}$ fizer uma consulta ao oráculo de deciframento, $\mathcal{A}'$ procede da seguinte forma:

1. Se o evento Forge ocorrer, então $\mathcal{A}'$ interrompe sua execução.
2. Caso contrário, $\mathcal{A}'$ decifra o texto cifrado usando os procedimentos de PKE_{cca2}.

Oráculo de Desafio: Quando $\mathcal{A}$ fizer a consulta ao oráculo de desafio com duas mensagens $\mathsf{m}_0, \mathsf{m}_1 \in \mathcal{M}$ tais que $|\mathsf{m}_0| = |\mathsf{m}_1|$, $\mathcal{A}'$ procede da seguinte maneira:

1. Envia m_0 e m_1 para o oráculo de desafio do jogo IND-CPA ($\mathcal{A}'$ tem acesso a esse oráculo pois ele ataca a segurança IND-CPA do criptosistema PKE_k) e obtêm como resposta $(\mathsf{c}_1^*, \ldots, \mathsf{c}_k^*)$.
2. Assina a mensagem $(\mathsf{c}_1^*, \ldots, \mathsf{c}_k^*)$ usando dsk^*.
3. Produz como saída o texto cifrado $\mathsf{c}^* = (\mathsf{c}_1^*, \ldots, \mathsf{c}_k^*, \mathsf{vk}^*, \sigma^*)$.

Output: Quando $\mathcal{A}$ gerar a saída b, $\mathcal{A}'$ também gera a saída b.

Se o evento Forge não ocorrer, a vantagem de $\mathcal{A}'$ de quebrar a segurança IND-CPA de PKE_k é a mesma que a vantagem de $\mathcal{A}$ de quebrar a segurança IND-CCA2 de PKE_{cca2}. Como PKE_k é IND-CPA seguro por hipótese, segue que PKE_{cca2} é IND-CCA2 seguro.

4.2 ESQUEMA DE MCELIECE RANDOMIZADO

Em [46] foi provado que o criptosistema obtido modificando o algoritmo de ciframento do criptosistema de McEliece para cifrar $\mathsf{r}|\mathsf{m}$ (para um valor aleatório r) ao invés de somente cifrar a mensagem m, o chamado esquema de McEliece Randomizado (Randomized McEliece Cryptosystem), é IND-CPA seguro.

Nós modificamos o algoritmo de ciframento do criptosistema de McEliece randomizado da seguinte maneira. Ao invés de escolher o vetor de erro aleatoriamente entre as cadeias de bits de tamanho n com peso de Hamming t, escolhemos cada bit do vetor de erro de acordo com a distribuição de Bernoulli $\mathcal{B}_\theta$ com parâmetro $\theta = \frac{t}{n} - \epsilon$ para algum $\epsilon > 0$.

Pela Lei dos Grandes Números, para n grande o suficiente, o peso de Hamming do vetor de erro gerado desta maneira estará entre $t - 2n\epsilon$ e t com probabilidade altíssima. Portanto o criptosistema atende o requisito da corretude. A segurança IND-CPA segue das hipóteses 2.7.1 e 2.7.2, pois ϵ pode ser arbitrariamente pequeno (para n suficientemente grande).

4.2.1 Segurança do Esquema de McEliece Randomizado k-repetido

Provamos nesta seção que a versão modificada do criptosistema de McEliece randomizado k-repetido é segura e verificável. Isto é, provamos que o criptosistema formado cifrando k vezes $\mathsf{r}|\mathsf{m}$ com diferentes chaves públicas e privadas ($\mathsf{PKE}_{k,McE}$) é correto, IND-CPA seguro e que ele permite a verificação da validade do texto cifrado dado as chaves públicas e uma chave secreta.

Pela corretude de cada instância, a probabilidade de que em uma instância $i \in \{1, \ldots, k\}$ o texto cifrado corretamente gerado seja decifrado incorretamente é desprezível. Como k é polinomial, temos, pelo Limite da União das Probabilidades (Inequação de Boole), que a probabilidade de um texto corretamente cifrado de $\mathsf{PKE}_{k,McE}$ ser decifrado incorretamente é desprezível também. Portanto $\mathsf{PKE}_{k,McE}$ atende o requisito da corretude.

Para provarmos que o criptosistema $\mathsf{PKE}_{k,McE}$ é admissível (e portanto IND-CPA seguro [46]), provamos que ele atende a propriedade da pseudo-aleatoriedade (a divisibilidade segue trivialmente). Denotemos por $\mathbf{R}_1, \ldots, \mathbf{R}_k$ matrizes aleatórias de tamanho $l \times n$, por $\mathbf{P}_1, \ldots, \mathbf{P}_k$ matrizes que representam as chaves públicas do criptosistema de McEliece e por $\mathsf{e}_1, \ldots, \mathsf{e}_k$ os vetores de erro. Definimos $l_1 = |r|$ e $l_2 = |m|$. Sejam $\mathbf{R}_{i,1}$ e $\mathbf{R}_{i,2}$ as submatrizes de $\mathbf{R}_i$ de dimensões $l_1 \times n$ e $l_2 \times n$, respectivamente, tais que $\mathbf{R}_i^T = \mathbf{R}_{i,1}^T | \mathbf{R}_{i,2}^T$. Definimos $\mathbf{P}_{i,1}$ e $\mathbf{P}_{i,2}$ de forma similar. Então utilizamos um lema de [50] relacionado

ao problema LPN:

Lema 4.2.1 *Suponha que exista um algoritmo $\mathcal{A}$ que realize q consultas ao oráculo, rode em tempo t, e tal que*

$$|\Pr[\mathcal{A}^{\mathcal{Q}_{r,\theta}} = 1 | r \leftarrow \{0,1\}^{l_1}] - \Pr[\mathcal{A}^{\mathcal{U}_{l_1+1}} = 1]| \geq \delta$$

Então existe um adversário $\mathcal{A}'$ que realiza $q' = O(q\delta^{-2}\log l_1)$ consultas ao oráculo, roda em tempo $t' = O(tl_1\delta^{-2}\log l_1)$, e tal que

$$\mathsf{Adv}_{\mathsf{LPN}_\theta,\mathcal{A}'} \geq \frac{\delta}{4}$$

Fazendo $q = kn$ no lema, temos que $(\mathsf{r}\mathbf{R}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{R}_{k,1} \oplus \mathsf{e}_k)$ é pseudo-aleatório se o problema LPN_θ for difícil.

Agora provamos que substituir as matrizes aleatórias pelas matrizes que representam as chaves públicas do criptosistema de McEliece não afeta a pseudo-aleatoriedade da saída $(\mathsf{r}\mathbf{P}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{P}_{k,1} \oplus \mathsf{e}_k)$.

Lema 4.2.2 $(\mathsf{r}\mathbf{P}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{P}_{k,1} \oplus \mathsf{e}_k)$ *é pseudo-aleatório.*

Suponha que algum adversário probabilístico de tempo polinomial $\mathcal{A}$ tenha uma vantagem não-desprezível de distinguir entre $(\mathsf{r}\mathbf{R}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{R}_{k,1} \oplus \mathsf{e}_k)$ e $(\mathsf{r}\mathbf{P}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{P}_{k,1} \oplus \mathsf{e}_k)$. Denotemos essas duas distribuições por H_0 e H_k respectivamente. Para $i \in \{1, \ldots, k-1\}$, definimos H_i como

$$(\mathsf{r}\mathbf{P}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{P}_{i,1} \oplus \mathsf{e}_i)|(\mathsf{r}\mathbf{R}_{i+1,1} \oplus \mathsf{e}_{i+1})| \ldots |(\mathsf{r}\mathbf{R}_{k,1} \oplus \mathsf{e}_k).$$

Como k é polinomial, usando o argumento híbrido é possível construir um adversário $\mathcal{A}'$ que utiliza $\mathcal{A}$ como uma caixa-preta e tem vantagem não-desprezível de distinguir entre H_{i-1} e H_i para algum $i \in \{1, \ldots, k\}$. Mas isso implica que $\mathcal{A}'$ tem uma vantagem não-desprezível de distinguir entre a chave pública $\mathbf{P}$ do criptosistema de McEliece e uma matriz aleatória de mesmo tamanho. Pela hipótese 2.7.1, não existe tal $\mathcal{A}'$ e portanto não pode existir um adversário $\mathcal{A}$ que tenha vantagem não-desprezível de distinguir entre H_0 e H_k.

Teorema 4.2.3 $\mathsf{PKE}_{k,McE}$ *é IND-CPA seguro.*

Dos lemas 4.2.1 e 4.2.2 temos que $(\mathsf{r}\mathbf{P}_{1,1} \oplus \mathsf{e}_1)| \ldots |(\mathsf{r}\mathbf{P}_{k,1} \oplus \mathsf{e}_k)$ é pseudo-aleatório. Portanto o criptosistema é admissível. A segurança IND-CPA do criptosistema segue do fato de que um criptosistema admissível é também IND-CPA seguro [46].

Teorema 4.2.4 $\mathsf{PKE}_{k,McE}$ *é verificável.*

Para verificar se um texto cifrado $(\mathsf{c}_1, \ldots, \mathsf{c}_k)$ é válido dado as chaves públicas e uma chave secreta do criptosistema de McEliece ($\mathbf{S}_j, \mathbf{G}_j, \mathbf{T}_j$), decifframos c_j obtendo $\mathsf{r}|\mathsf{m}$. Então para todo $i \in \{1, \ldots, k\}$ computamos $\mathsf{c'}_i = (\mathsf{r}|\mathsf{m})\mathbf{P}_i$ e verificamos se a distância de Hamming entre $\mathsf{c'}_i$ e c_i é menor que ou igual a t.

Teorema 4.2.5 *É possível construir um criptosistema de chave pública IND-CCA2 seguro baseado nas hipóteses de McEliece.*

Segue diretamente dos teoremas 4.1.4, 4.2.3 e 4.2.4.

5 REDUZINDO O TAMANHO DO TEXTO CIFRADO EM CRIPTOSISTEMAS DO TIPO DDN

Nesse capítulo apresentamos a nossa técnica para reduzir o tamanho do texto cifrado dos criptosistemas do tipo DDN, sem alterar nenhum dos outros parâmetros do esquema. Esse trabalho foi realizado em conjunto com Goichiro Hanaoka (AIST), Hideki Imai (AIST) e Anderson Nascimento (Universidade de Brasília, UnB). Embora a técnica seja apresentada para o esquema DDN original [26], ela também pode ser utilizada em outras construções que utilizam vários pares de chaves públicas/privadas, tais como: Rosen-Segev [65], Pass-Shelat-Vaikuntanathan [60], Hanaoka-Imai-Ogawa-Watanabe [43] e o nosso esquema descrito no capítulo 4.

5.1 NOSSA CONSTRUÇÃO APERFEIÇOADA

5.1.1 O esquema

Aqui apresentamos nossa versão aperfeiçoada do criptosistema DDN.

Geração de Chaves: Para um parâmetro de segurança n, executamos $2n$ vezes $\mathsf{CPA.Gen}(1^k)$ para obter

$$\mathsf{sk} = (\mathsf{sk}_{00,1}, \ldots, \mathsf{sk}_{00,n/2}, \mathsf{sk}_{01,1}, \ldots, \mathsf{sk}_{01,n/2}, \mathsf{sk}_{10,1}, \ldots, \mathsf{sk}_{10,n/2}, \mathsf{sk}_{11,1}, \ldots, \mathsf{sk}_{11,n/2});$$

$$\mathsf{pk}' = (\mathsf{pk}_{00,1}, \ldots, \mathsf{pk}_{00,n/2}, \mathsf{pk}_{01,1}, \ldots, \mathsf{pk}_{01,n/2}, \mathsf{pk}_{10,1}, \ldots, \mathsf{pk}_{10,n/2}, \mathsf{pk}_{11,1}, \ldots, \mathsf{pk}_{11,n/2}).$$

Escolhemos um número aleatório r. A chave secreta é sk e a chave pública é $\mathsf{pk} = (\mathsf{pk}', r)$.

Ciframento: Para cifrar uma mensagem m executamos $\mathsf{SS.Gen}(1^n)$ para obter a chave de verificação vk e a chave de assinatura dsk. Seja vk_i o i-ésimo bit de vk. Então executamos $\mathsf{c}_i \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}_{vk_{2j-1}vk_{2j},j}, \mathsf{m})$ para $1 \leq j \leq n/2$. Geramos uma prova de conhecimento nulo não-interativa π que garanta que todas as mensagens cifradas nos c_i são idênticas.[1]. E por fim rodamos $\sigma \leftarrow \mathsf{SS.Sign}((\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi), dsk)$. O texto cifrado é $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi, vk, \sigma)$.

Deciframento: Para decifrar um texto cifrado $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi, vk, \sigma)$, executamos $\mathsf{SS.Ver}((\mathsf{c}_1, \ldots,$

[1] Como essa é uma linguagem NP, essa prova de conhecimento nulo não-interativa pode ser gerada.

$\mathsf{c}_{n/2}, \pi), \sigma, vk)$ para checar a validade da assinatura digital. Também testamos a validade de π. Se alguma das provas for inválida, geramos como saída $\perp$. Caso contrário, geramos como saída $\mathsf{m} \leftarrow \mathsf{CPA.Dec}(\mathsf{c}_1, \mathsf{sk}_{vk_1 vk_2,1})$.

5.1.2 Prova de Segurança

Nosso esquema, apresenta acima, é tão seguro quanto o criptosistema DDN original. Ou seja, ele é IND-CCA2 seguro assumindo que Π seja IND-CPA seguro, Σ seja OTSU seguro e que a prova de conhecimento nulo não-interativo seja adaptativamente segura.

Teorema 5.1.1 *O esquema proposto acima é IND-CCA2 seguro assumindo que o criptosistema de chave pública Π é IND-CPA seguro, que o esquema de assinatura digital Σ é OTSU seguro e que o esquema de provas de conhecimento nulo não-interativa é adaptativamente seguro.*

Agora desenvolvemos a prova de segurança do esquema proposto. Usando um algoritmo $\mathcal{A}$ que quebre a segurança IND-CCA2 do esquema proposto, construimos outro algoritmo $\mathcal{B}$ que quebra a segurança IND-CPA do esquema Π'. Abaixo descrevemos Π'.

Geração de Chaves: Para um parâmetro de segurança n, executamos $n/2$ vezes $\mathsf{CPA.Gen}(1^n)$ para obter

$$\tilde{\mathsf{sk}} = (\mathsf{sk}_1, \ldots, \mathsf{sk}_{n/2});$$

$$\tilde{\mathsf{pk}} = (\mathsf{pk}_1, \ldots, \mathsf{sk}_{n/2}).$$

A chave secreta é $\tilde{\mathsf{sk}}$ e a chave pública é $\tilde{\mathsf{pk}}$.

Ciframento: Para cifrar m, executamos $\mathsf{c}_i \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}_j, \mathsf{m})$ para $1 \leq j \leq n/2$.

Deciframento: Para decifrar um texto cifrado $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_{n/2})$ rodamos $\mathsf{m}_j \leftarrow \mathsf{CPA.Dec}(\mathsf{c}_j, \mathsf{sk}_j)$ para $1 \leq j \leq n/2$. Se $\mathsf{m}_1 = \ldots = \mathsf{m}_{n/2}$, geramos a saída m_1. Caso contrário geramos a saída $\perp$.

Devido ao argumento híbrido, segue que o esquema Π' é IND-CPA seguro se Π for IND-CPA seguro.

Para uma chave pública $\tilde{\mathsf{pk}} = (\mathsf{pk}_1, \ldots, \mathsf{pk}_{n/2})$, $\mathcal{B}$ simula o jogo IND-CCA2 para $\mathcal{A}$ da seguinte forma.

Geração de Chaves: $\mathcal{B}$ gera a chave pública $\mathsf{pk} = (\mathsf{pk}', r)$ que será a entrada para $\mathcal{A}$, onde

$$\mathsf{pk}' = (\mathsf{pk}_{00,1}, \ldots, \mathsf{pk}_{00,n/2}, \mathsf{pk}_{01,1}, \ldots, \mathsf{pk}_{01,n/2}, \mathsf{pk}_{10,1}, \ldots, \mathsf{pk}_{10,n/2}, \mathsf{pk}_{11,1}, \ldots, \mathsf{pk}_{11,n/2}),$$

da seguinte forma:

1. Executa $(\mathsf{vk}^\star, \mathsf{dsk}^\star) \leftarrow \mathsf{SS.Gen}(1^n)$. Seja $\mathsf{vk}_i^\star$ o i-ésimo bit de $\mathsf{vk}^\star$.
2. Para $1 \leq j \leq n/2$ e para $b\beta \in \{00, 01, 10, 11\}$,
   (a) Se $b\beta = \mathsf{vk}_{2j-1}^\star \mathsf{vk}_{2j}^\star$, então $pk_{b\beta,j} = pk_j$.
   (b) Se $b\beta \neq \mathsf{vk}_{2j-1}^\star \mathsf{vk}_{2j}^\star$, então executa $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{CPA.Gen}(1^n)$ e define $\mathsf{pk}_{b\beta,j} = \mathsf{pk}$ e $\mathsf{sk}_{b\beta,j} = \mathsf{sk}$.

r também é gerado de forma adequada por $\mathcal{B}$. $\mathcal{B}$ passa a entrada pk para $\mathcal{A}$.

Consultas ao Oráculo de Deciframento: Quando $\mathcal{A}$ submete uma consulta $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi, \mathsf{vk}, \sigma)$ ao oráculo de deciframento, $\mathcal{B}$ verifica a validade de c, e se inválido gera a saída $\perp$. Caso contrário, $\mathcal{B}$ seleciona j tal que $\mathsf{vk}_{2j-1}\mathsf{vk}_{2j} \neq \mathsf{vk}_{2j-1}^\star \mathsf{vk}_{2j}^\star$ (se $\mathsf{vk} \neq \mathsf{vk}^\star$, então existe tal j), computa $\mathsf{m}_j \leftarrow \mathsf{CPA.Dec}(\mathsf{c}_j, \mathsf{sk}_{vk_{2j-1}vk_{2j},j})$ e retorna m_j para $\mathcal{A}$. Se $\mathsf{vk} = \mathsf{vk}^\star$, então $\mathcal{B}$ termina sua execução.

Oráculo de Desafio: Quando $\mathcal{A}$ enviar as duas mensagens m_0 e m_1 tal que $|\mathsf{m}_0| = |\mathsf{m}_1|$ para o oráculo de desafio, $\mathcal{B}$ submete essas mensagens para o seu próprio oráculo de desafio. Seja $(\mathsf{c}_1^\star, \ldots, \mathsf{c}_{n/2}^\star)$ o texto cifrado desafio para Π'. Então $\mathcal{B}$ gera uma prova de conhecimento nulo não-interativa $\pi^\star$ que garanta que o resultado de decifrar $\mathsf{c}_1^\star, \ldots, \mathsf{c}_{n/2}^\star$ é igual (como r foi gerado por $\mathcal{B}$, ele pode usar informação adicional [trapdoor information] para gerar tal $\pi^\star$ sem conhecer as mensagens e nem a aleatoriedade utilizada no algoritmo de ciframento). Por fim, $\mathcal{B}$ computa $\sigma^\star \leftarrow \mathsf{SS.Sign}((\mathsf{c}_1^\star, \ldots, \mathsf{c}_{n/2}^\star, \pi^\star), \mathsf{dsk}^\star)$ e envia o texto cifrado desafio $\mathsf{c}^\star = (\mathsf{c}_1^\star, \ldots, \mathsf{c}_{n/2}^\star, \pi^\star, \mathsf{vk}^\star, \sigma^\star)$ para $\mathcal{A}$

Saída: Quando $\mathcal{A}$ gerar uma saída b, $\mathcal{B}$ gera a mesma saída como sendo o seu palpite.

A simulação da interação IND-CCA2 falha quando $\mathcal{A}$ submete ao oráculo de deciframento um texto cifrado válido tal que $\mathsf{vk} = \mathsf{vk}^\star$ ou quando $\mathcal{A}$ produz uma prova de conhecimento nulo não interativa válida para um texto cifrado no qual c_i e c_j (para algum $i, j \in \{1, \ldots, n/2\}$) são textos cifrados para mensagens diferentes. Mas no entanto a probabilidade de que o primeiro evento ocorra é desprezível, pois Σ é OTSU seguro por hipótese. E a probabilidade de que o segundo evento ocorra também é desprezível, pois o esquema de provas de conhecimento nulo não-interativo é adaptativamente seguro (e portanto atende ao requisito da corretude). Claramente a vantagem de $\mathcal{B}$ no jogo IND-CCA2 é a mesma que a vantagem de $\mathcal{A}$ no jogo IND-CPA (a menos da probabilidade de que $\mathcal{A}$ produza um dos dois eventos acima).

Seja Forge o evento de que para alguma consulta de $\mathcal{A}$ ao oráculo de deciframento tenhamos $\mathsf{Ver}((\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi), \sigma, \mathsf{vk}) = 1$ e $\mathsf{vk} = \mathsf{vk}^*$.

Lema 5.1.2 $\Pr[\mathsf{Forge}]$ *é desprezível.*

Assuma que para um adversário probabilístico de tempo polinomial $\mathcal{A}$ contra o esquema tenhamos Pr[Forge] não-desprezível, então construimos um adversário $\mathcal{B}'$ que forja uma assinatura com a mesma probabilidade. $\mathcal{B}'$ simula a interação IND-CCA2 para $\mathcal{A}$ da seguinte maneira:

Geração de Chaves $\mathcal{B}'$ invoca o algoritmo de geração de chaves do esquema de assinatura digital e obtêm a chave de verificação vk^*. Ele também invoca $2n$ vezes o algoritmo de geração de chave de Π, obtendo as chaves públicas

$$\mathsf{pk}_{00,1}, \ldots, \mathsf{pk}_{00,n/2}, \mathsf{pk}_{01,1}, \ldots, \mathsf{pk}_{01,n/2}, \mathsf{pk}_{10,1}, \ldots, \mathsf{pk}_{10,n/2}, \mathsf{pk}_{11,1}, \ldots, \mathsf{pk}_{11,n/2},$$

que formam pk' e as chaves secretas

$$\mathsf{sk}_{00,1}, \ldots, \mathsf{sk}_{00,n/2}, \mathsf{sk}_{01,1}, \ldots, \mathsf{sk}_{01,n/2}, \mathsf{sk}_{10,1}, \ldots, \mathsf{sk}_{10,n/2}, \mathsf{sk}_{11,1}, \ldots, \mathsf{sk}_{11,n/2},$$

que formam sk. r também é gerado adequadamente por $\mathcal{B}'$. $\mathcal{B}'$ envia a entrada pk para $\mathcal{A}$.

Consultas ao Oráculo de Deciframento: Quando $\mathcal{A}$ faz uma consulta ao oráculo de deciframento, $\mathcal{B}'$ procede da seguinte forma:

1. Se $\mathsf{vk} = \mathsf{vk}^*$ e $\mathsf{SS.Ver}((\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi), \sigma, \mathsf{vk}) = 1$, $\mathcal{B}'$ gera como saída a assinatura forjada $((\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi), \sigma)$ e termina sua execução.
2. Caso contrário, $\mathcal{B}'$ decifra o texto cifrado usando os procedimentos do nosso esquema.

Oráculo de Desafio: Quando $\mathcal{A}$ faz uma consulta ao oráculo de desafio com duas mensagens $m_0, m_1 \in \mathcal{M}$ tal que $|m_0| = |m_1|$, $\mathcal{B}'$ procede da seguinte forma:

1. Escolhe aleatoriamente $b \in \{0, 1\}$.
2. Cifra a mensagem m_b usando os procedimentos do esquema. Isso é possível porque $\mathcal{B}'$ pode solicitar ao seu oráculo de assinatura que assine uma mensagem, então ele solicita que o oráculo assine o valor $(\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}, \pi)$ obtido durante o processo de cifragem.

Se o evento Forge não ocorrer, a simulação é perfeita. Portanto a probabilidade de $\mathcal{B}'$ quebrar a segurança OTSU do esquema de assinatura digital é exatamente Pr[Forge]. Como o esquema de assinatura

é OTSU seguro por hipótese, Pr[Forge] é desprezível para todos adversários probabilísticos e de tempo polinomial contra o nosso esquema.

Seja $\mathsf{V_{ILL}}$ o evento que $\mathcal{A}$ produza uma prova de conhecimento nulo não-interativa para a validade de um texto cifrado no qual c_i e c_j correspondem a mensagens diferentes (para algum $i, j \in \{1, \ldots, n/2\}$).

Lema 5.1.3 $\Pr[\mathsf{V_{ILL}}]$ *é desprezível.*

Assuma que para um adversário probabilístico de tempo polinomial $\mathcal{A}$ contra o esquema tenhamos $\Pr[\mathsf{V_{ILL}}]$ não-desprezível, então construimos um adversário $\mathcal{B}''$ que viola a corretude do esquema de provas de conhecimento nulo não-interativo com a mesma probabilidade. $\mathcal{B}''$ simula a interação IND-CCA2 para $\mathcal{A}$ da seguinte maneira:

Geração de Chaves: $\mathcal{B}''$ recebe r como entrada. Ele executa $2n$ vezes o algoritmo de geração de chaves de Π obtendo as chaves públicas

$$\mathsf{pk}_{00,1}, \ldots, \mathsf{pk}_{00,n/2}, \mathsf{pk}_{01,1}, \ldots, \mathsf{pk}_{01,n/2}, \mathsf{pk}_{10,1}, \ldots, \mathsf{pk}_{10,n/2}, \mathsf{pk}_{11,1}, \ldots, \mathsf{pk}_{11,n/2},$$

que formam pk' e as chaves secretas

$$\mathsf{sk}_{00,1}, \ldots, \mathsf{sk}_{00,n/2}, \mathsf{sk}_{01,1}, \ldots, \mathsf{sk}_{01,n/2}, \mathsf{sk}_{10,1}, \ldots, \mathsf{sk}_{10,n/2}, \mathsf{sk}_{11,1}, \ldots, \mathsf{sk}_{11,n/2},$$

que formam sk. $\mathcal{B}''$ envia a entrada $\mathsf{pk} = (\mathsf{pk}', \mathsf{r})$ para$\mathcal{A}$.

Consultas ao Oráculo de Deciframento: Quando $\mathcal{A}$ faz uma consulta ao oráculo de deciframento, $\mathcal{B}''$ procede da seguinte forma:

1. Se para esse texto cifrado π for válido e houver c_i e c_j que correspondem a mensagens diferentes (para algum $i, j \in \{1, \ldots, n/2\}$), $\mathcal{B}''$ gera a saída $((\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}), \pi)$ e termina sua execução.
2. Caso contrário, $\mathcal{B}''$ decifra o texto cifrado usando os procedimentos do nosso esquema.

Oráculo de Desafio: Quando $\mathcal{A}$ faz uma consulta ao oráculo de desafio com duas mensagens $m_0, m_1 \in \mathcal{M}$ tal que $|m_0| = |m_1|$, $\mathcal{B}''$ procede da seguinte maneira:

1. Escolhe aleatoriamente $b \in \{0, 1\}$.
2. Cifra a mensagem m_b usando os procedimentos do nosso esquema.

Se o evento $\mathsf{V_{ILL}}$ não ocorrer, a simulação é perfeita. Portanto a probabilidade de que $\mathcal{B}''$ quebre a corretude do esquema de provas de conhecimento nulo não-interativo é exatamente $\Pr[\mathsf{V_{ILL}}]$. Como o esquema de provas de conhecimento nulo não-interativo é adaptativamente seguro por hipótese, $\Pr[\mathsf{V_{ILL}}]$ é desprezível para todos adversários probabilísticos e de tempo polinomial contra o nosso esquema.

5.1.3 Performance: Comparação com o Esquema Original DDN

A maior vantagem da nossa construção em relação ao esquema original DDN é que o tamanho do texto cifrado e o custo computacional é reduzido significativamente *sem sacrificar nenhum dos outros aspectos*. Especificamente, no nosso esquema o número de textos cifrados componentes que formam o texto cifrado final (isto é, $(\mathsf{c}_1, \ldots, \mathsf{c}_{n/2})$) é reduzido pela metade em relação ao esquema original sem aumentar o tamanho das chaves. Além disso, como a prova de conhecimento nulo não-interativa para provar que os resultados de decifrar $\mathsf{c}_1, \ldots, \mathsf{c}_{n/2}$ são idênticos pode ser significativamente mais simples do que essa prova para $\mathsf{c}_1, \ldots, \mathsf{c}_n$. Portanto esse componente do texto cifrado também pode ser reduzido. Assim temos que a performance do nosso esquema é equivalente ou superior a do esquema original DDN em *todos os aspectos*. Fazemos abaixo um resumo das propriedades do nosso esquema em comparação com o DDN original.

Tamanho do Texto Cifrado	aproximadamente metade
Tamanho da Chave Pública	igual
Tamanho da Chave Secreta	igual
Custo Computacional do Ciframento	aproximadamente metade
Custo Computacional do Deciframento	aproximadamente o mesmo

5.2 EXPLICAÇÃO INTUITIVA DO NOSSO ARTIFÍCIO

Nesta seção nós damos uma explicação sobre o artifício que utilizamos para obter essa melhoria. Começamos a explicação com a seguinte observação. Olhando atentamente a prova de segurança do esquema DDN original [26], notamos que ela continuaria sendo válida para a seguinte construção generalizada:

Geração de Chaves: Para o parâmetro de segurança n, executamos kN vezes o algoritmo $\mathsf{CPA.Gen}(1^n)$ para obter

$$\mathsf{sk} = (\mathsf{sk}_{1,1}, \ldots, \mathsf{sk}_{1,N}, \mathsf{sk}_{2,1}, \ldots, \mathsf{sk}_{2,N}, \ldots, \mathsf{sk}_{k,1} \ldots, \mathsf{sk}_{k,N});$$

$$\mathsf{pk}' = (\mathsf{pk}_{1,1}, \ldots, \mathsf{pk}_{1,N}, \mathsf{pk}_{2,1}, \ldots, \mathsf{pk}_{2,N}, \ldots, \mathsf{pk}_{k,1} \ldots, \mathsf{pk}_{k,N}).$$

Escolhemos r aleatoriamente. A chave secreta é sk e a chave pública é $\mathsf{pk} = (\mathsf{pk}', \mathsf{r})$.

Ciframento: Para cifrar a mensagem m, executamos $\mathsf{SS.Gen}(1^n)$ para obter a chave de verificação vk e a chave de assinatura dsk. Assumimos que vk pode ser expresso como $(\mathsf{vk}_1, \mathsf{vk}_2, \ldots, \mathsf{vk}_N) \in \{1, \ldots, k\}^N$ tal que todos $i \in \{1, \ldots, N\}$, $\mathsf{vk}_i \in \{1, \ldots, k\}$. Então obtemos $\mathsf{c}_i \leftarrow \mathsf{CPA.Enc}(\mathsf{pk}_{vk_j,j}, \mathsf{m})$ for $1 \leq j \leq N$. Após isso geramos uma prova de conhecimento nulo não-interativa π que garanta que a mensagem cifrada em todos c_i é idêntica. Por fim, executamos $\sigma \leftarrow \mathsf{SS.Sign}((\mathsf{c}_1, \ldots, \mathsf{c}_N, \pi), \mathsf{dsk})$. O texto cifrado é $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_N, \pi, \mathsf{vk}, \sigma)$.

Deciframento: Para decifrar o texto cifrado $\mathsf{c} = (\mathsf{c}_1, \ldots, \mathsf{c}_N, \pi, \mathsf{vk}, \sigma)$, rodamos $\mathsf{SS.Ver}((\mathsf{c}_1, \ldots, \mathsf{c}_N, \pi), \sigma, \mathsf{vk})$ para checar a validade da assinatura. Também testamos a validade de π. Se uma das provas for inválida, geramos a saída $\perp$. Caso contrário, geramos a saída $\mathsf{m} \leftarrow \mathsf{CPA.Dec}(\mathsf{c}_1, \mathsf{sk}_{vk_1,1})$.

Podemos provar que o esquema acima é IND-CCA2 seguro se $k^N \geq 2^n$. Essa prova pode ser obtida trivialmente da prova apresentada na seção anterior.

A partir dessa observação, percebemos que assintoticamente existe uma relação de compromisso entre N e $k \times N$. Isso implica que se reduzirmos o tamanho do texto cifrado (em outras palavras, diminuirmos N), então teremos que aumentar o tamanho da chave (em outras palavras, aumentar kN). Denotemos por $|KEY|$ e $|CTXT|$ as quantidades kN e N respectivamente. Logo temos a seguinte inequação:

$$|KEY| \geq |CTXT| \cdot 2^{n|CTXT|^{-1}}.$$

Essa inequação implica que se $|CTXT|$ diminuir, então $|KEY|$ aumenta *exponencialmente*. Logo a generalização acima não é muito útil no caso geral. No entanto, essa observação assintótica não é válida sempre. Especificamente o lado direito da inequação acima não é monotônica em $|CTXT|$ (o seu mínimo local é em $|CTXT| = n/(\log_2 e)$ e portanto existem definições de parâmetros interessantes para $n/2 \leq |CTXT| \leq n$.

Baseado nessa observação, podemos construir algumas variantes do DDN interessantes. Por exemplo, definindo $|CTXT| = n/2$, teremos que $|KEY|$ permanece igual ao valor para $|CTXT| = n$. Isso significa que podemos comprimir o tamanho do texto cifrado sem aumentar o tamanho da chave (esse foi o exemplo apresentado na seção anterior). Outro exemplo interessante é obtido definindo $|CTXT| \simeq$

$n/(\log_2 e)$, reduzindo assim ambos $|CTXT|$ e $|KEY|$. Outras variações interessantes parecem ser possíveis.

6 CONCLUSÕES

Essa dissertação tem três contribuições: (1) apresentamos um protocolo de Oblivious Transfer baseado nas hipóteses de McEliece, (2) obtemos um criptosistema de chave pública IND-CCA2 seguro baseado nas hipóteses de McEliece sem oráculo aleatório e (3) mostramos uma técnica para reduzir o tamanho dos textos cifrados pela metade em criptosistemas do tipo DDN.

No capítulo 3, descrevemos um 1-2 OT baseado somente nas duas hipóteses de McEliece. Essas hipóteses foram utilizadas para construir um dos primeiros criptosistemas de chave pública e, em sua formulação original, não foram quebradas até hoje, mesmo por algoritmos quânticos. Esse fato nos dá certa confiança sobre a veracidade das hipóteses. Uma das hipóteses de McEliece (dificuldade do Problema da Decodificação de Síndrome) é na verdade NP-completa e se alguém provar que decodificar um código linear aleatório é fácil, isso implicaria que $P = NP$ (uma das questões em aberto mais importantes da ciência da computação).

Até onde sabemos, esse é o primeiro protocolo de OT baseado somente nas hipóteses de McEliece e, concomitantemente com [52], o primeiro protocolo de OT computacionalmente seguro para o qual não existe um ataque quântico eficiente. Nós acreditamos que é importante construir protocolos de OT seguros baseados em outras hipóteses computacionais para as quais não existe um ataque quântico, já que OT é um dos protocolos fundamentais para a computação segura multi-parte.

Um problema em aberto é construir um protocolo de OT seguro contra adversários dinâmicos usando somente as hipóteses de McEliece. Outro problema em aberto é provar a segurança de algum protocolo de OT baseado nestas hipóteses utilizando o paradigma da simulação, especialmente no modelo UC [10]. Nós obtemos a segurança no que é chamado na literatura de modelo “half-simulatable”. Nós acreditamos que obter protocolos de OT UC-seguros baseados em hipóteses para as quais não existem ataque quânticos eficientes é uma questão fundamental dada a importância do OT para a computação segura multi-parte.

No capítulo 4, apresentamos um criptosistema de chave pública IND-CCA2 seguro baseado nas mesmas hipóteses mencionadas acima e sem o uso de oráculos aleatórios. Pelas mesmas razões apresentadas anteriormente, achamos importante obter criptosistemas seguros utilizando hipóteses para as quais não se conhecem ataques quânticos eficientes, de preferência criptosistemas IND-CCA2 seguros (que é a noção mais forte de segurança para criptosistemas de chave pública e implica que ele pode ser composto de forma arbitrária com quaisquer outros protocolos sem que sua segurança seja afetada) como é o nosso caso. Um

problema em aberto é obter um criptosistema mais eficiente utilizando estas hipóteses.

Por fim, no capítulo 5, modificamos o criptosistema DDN e reduzimos o tamanho do texto cifrado para a metade sem alterar o tamanho das chaves e nem a segurança. Nossa melhoria é baseada em um artifício que permite uma relação assintótica de compromisso entre o tamanho do texto cifrado e o tamanho das chaves, mas que para os nossos parâmetros específicos permite uma redução do tamanho do texto cifrado sem alterar os demais parâmetros do criptosistema.

REFERÊNCIAS

[1] AIELLO, W.;ISHAI, Y.;REINGOLD, O. Priced Oblivious Transfer: How to Sell Digital Goods. *EUROCRYPT 2001*. pp. 119–135. 2001.

[2] BEAVER, D. Precomputing Oblivious Transfer. *CRYPTO 1995*. pp. 97–109. 1995.

[3] BELLARE, M.;MICALI, S. Non-Interactive Oblivious Transfer and Applications, *CRYPTO 1989*. pp. 547–557. 1990.

[4] BERLEKAMP, E.;MCELIECE, R.;VAN TILBORG, H. On the Inherent Intractability of Certain Coding Problems. *IEEE Trans. Inf. Theory* Vol. 24. pp. 384–386. 1978.

[5] BERNSTEIN, D.; LANGE, T.; PETERS, C. Attacking and Defending the McEliece cryptosystem. Available at http://eprint.iacr.org/2008/318.

[6] BRASSARD, G.; CHAUM, D.; CRÉPEAU, C. Minimum Disclosure Proofs of Knowledge. *JCSS*, Vol. 37, No. 2, pp. 156–189, 1988.

[7] BRASSARD, G.; CRÉPEAU, C. Oblivious Transfers and Privacy Amplification. *EUROCRYPT 1997*. pp. 334–347. 1997.

[8] BRASSARD, G.; CRÉPEAU, C.; ROBERT, J. Information Theoretic Reductions among Disclosure Problems. *27th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1986)*. pp. 168–173. 1986.

[9] BRASSARD, G.; CRÉPEAU, C.; WOLF, S. Oblivious Transfers and Privacy Amplification. *Journal of Cryptology*. 16(4). pp. 219–237. 2003.

[10] CANETTI, R. Universally Composable Security: A New Paradigm for Cryptographic Protocols. Available at http://eprint.iacr.org/2000/067. Extended Abstract appeared in: *42nd Symposium on Foundations of Computer Science (FOCS)*. pp. 136–145. 2001.

[11] CANETTI , R.; HALEVI, S.; KATZ, J. Chosen-Ciphertext Security from Identity-Based Encryption. *EUROCRYPT 2004*. pp. 207–222. 2004.

[12] CANTEAUT, A.;CHABAUD, F. A New Algorithm for Finding Minimum-weight Words in a Linear Code: Application to Primitive Narrow-sense BCH Codes of Length 511. *IEEE Trans. Inf. Theory*. Vol. 44(1). pp. 367–378. 1998.

[13] CHAUM, D.; CRÉPEAU, C.; DAMGÅRD, I. Multiparty Unconditionally Secure Protocols (Extended Abstract) *STOC 1988*. pp. 11–19. 1988.

[14] CHAUM, D.; DAMGÅRD, I.; VAN DE GRAAF, J. Multiparty Computations Ensuring Privacy of Each Party's Input and Correctness of the Result. *CRYPTO 1987*. pp. 87–119. 1987.

[15] CHAUM, D.; FIAT, A.; NAOR, M. Untraceable Electronic Cash. *CRYPTO 1988*. pp. 319–327. 1988.

[16] COURTOIS, N.; FINIASZ, M.; SENDRIER, N. How to Achieve a McEliece Digital Signature Scheme. *Asiacrypt 2001*. pp. 157–174. 2001.

[17] CRAMER, R.; SHOUP, V. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. *CRYPTO 1998*. pp. 13–25. 1998.

[18] CRÉPEAU, C. Equivalence Between Two Flavors of Oblivious Transfers. *CRYPTO 1987*. pp. 350–354. 1988.

[19] CRÉPEAU, C.; VAN DE GRAAF, J.; TAPP, A. Committed Oblivious Transfer and Private Multi-Party Computations. *CRYPTO 1995*. pp. 110–123. 1995.

[20] CRÉPEAU, C.; KILIAN, J. Achieving Oblivious Transfer using Weakened Security Assumptions. 29^{th} *FOCS*. pp. 42–52. 1988.

[21] CRÉPEAU, C.; MOROZOV, K.; WOLF, S. Efficient Unconditional Oblivious Transfer from Almost any Noisy Channel. *In Proceedings of Fourth Conference on Security in Communication Networks (SCN)*. pp. 47–59. 2004.

[22] DAMGÅRD, I.; KILIAN, J.; SALVAIL, L. On the (Im)possibility of Basing Oblivious Transfer and Bit Commitment on Weakened Security Assumptions. *EUROCRYPT 1999*. pp. 56–73. 1999.

[23] DENG, R.; LI, Y.; WANG, X. The Equivalence of McEliece's and Niederreiter's Public-key Cryptosystems. *IEEE Transactions on Information Theory*. Vol. 40. pp. 271–273. 1994.

[24] DIFFIE, W.; HELLMANN, M. New Directions in Cryptography. *IEEE Transactions on Information Theory*. Vol. IT-22. pp. 644–654. 1976.

[25] DODIS, Y.; MICALI, S. Lower Bounds for Oblivious Transfer Reductions. *EUROCRYPT 1999*. pp. 42–55. 1999.

[26] DOLEV, D.; DWORK, CYNTHIA.; NAOR, M. Nonmalleable Cryptography. *SIAM J. Comput. 30(2)*. pp. 391–437, 2000.

[27] DOWSLEY, R.; MÜLLER-QUADE, J.; NASCIMENTO, A. C. A. A CCA2 Secure Public Key Encryption Scheme Based on the McEliece Assumptions in the Standard Model. *CT-RSA 2009*. pp. 240–251. 2009.

[28] DOWSLEY, R.; VAN DE GRAAF, J.; MÜLLER-QUADE, J.; NASCIMENTO, A. C. A. Oblivious Transfer Based on the McEliece Assumptions. *ICITS 2008*. pp. 107–117. 2008.

[29] DOWSLEY, R.; HANAOKA, G.; IMAI, H.; NASCIMENTO, A. C. A. Reducing the Ciphertext Size of Dolev-Dwork-Naor like Public Key Cryptosystems. Available at Cryptology ePrint Archive 2009/271.

[30] EVEN, S.; GOLDREICH, O.; LEMPEL, A. A Randomized Protocol for Signing Contracts. *Communications of the ACM 28(6)*. pp. 637–647, 1985.

[31] FISCHER, J.;STERN, J. An Efficient Pseudo-Random Generator Provably as Secure as Syndrome Decoding. *EUROCRYPT 1996*. pp. 245–255. 1996.

[32] GARAY, J.; MACKENZIE, P.; YANG, K. Efficient and Universally Composable Committed Oblivious Transfer and Applications. *TCC 2004*. pp. 297–316. 2004.

[33] GERTNER, Y;KANNAN, S.;MALKIN T.;REINGOLD, O.;VISWANATHAN, M. The Relationship between Public Key Encryption and Oblivious Transfer. *FOCS 2000*. pp. 325–335. 2000.

[34] GOLDREICH, O. Foundations of Cryptography: Volume 1 - Basic Tools. Cambridge University Press. 2001.

[35] GOLDREICH, O. Foundations of Cryptography: Volume 2 - Basic Applications. Cambridge University Press. 2004.

[36] GOLDREICH, O.;LEVIN, L. Hard-Core Predicates for Any One-Way Function. In *21st ACM Symposium on the Theory of Computing*. pp. 25–32. 1989.

[37] GOLDREICH, O.; MICALI, S.; WIGDERSON, A. How to Play Any Mental Game or a Completeness Theorem for Protocols with Honest Majority. *STOC 1987*. pp. 218–229. 1987.

[38] GOLDREICH, O.; MICALI, S.; WIGDERSON, A. Proofs that Yield Nothing but their Validity or All Languages in NP have Zero-Knowledge Proof System. *J. ACM 38(3)*. pp. 691–729. 1991.

[39] GOLDWASSER, S.;MICALI, S. Probabilistic Encryption. J. Comput. Syst. Sci. 28(2). pp. 270–299. 1984.

[40] GOLDWASSER, S.; MICALI, S.; RIVEST, R. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM J. Comput. 17(2)*. pp. 281–308. 1988.

[41] GOLDWASSER, S.;VAIKUNTANATHAN, V. Correlation-secure Trapdoor Functions from Lattices. Manuscript, 2008.

[42] HAITNER, I. Implementing Oblivious Transfer Using Collection of Dense Trapdoor Permutations. *TCC 2004*. pp. 394–409. 2004.

[43] HANAOKA, G.; IMAI, H.; OGAWA, K.; WATANABE, H. Chosen Ciphertext Secure Public Key Encryption with a Simple Structure. *IWSEC 2008*. pp. 20–33. 2008.

[44] HOFHEINZ, D.; MÜLLER-QUADE, J.; STEINWANDT, R. On Modeling IND-CCA Security in Cryptographic Protocolos. *WARTACRYPT 2004*. pp. 47–49. 2004. Full version at http://eprint.iacr.org/2003/024.

[45] HOFHEINZ, D.; KILTZ, E. Secure Hybrid Encryption from Weakened Key Encapsulation. *CRYPTO 2007*. pp. 553–571. 2007.

[46] IMAI, H.; KOBARA, K.; MOROZOV, K.; NOJIMA, R. Semantic Security for the McEliece Cryptosystem without Random Oracles. *Proceedings of International Workshop on Coding and Cryptography (WCC) 2007*. pp. 257–268. 2007. Journal version in Designs, Codes and Cryptography. Vol. 49. No. 1-3. pp. 289–305. 2008.

[47] IMPAGLIAZZO, R.; RUDICH, S. Limits on the Provable Consequences of One-way Permutations. *21st Annual ACM Symposium on Theory of Computing (STOC 1989)*. pp. 186–208. 1989.

[48] KALAI, Y. Smooth Projective Hashing and Two-Message Oblivious Transfer. *EUROCRYPT 2005*. pp. 78–95. 2005.

[49] KATZ, J. Lecture Notes. Available at http://www.cs.umd.edu/ jkatz/gradcrypto2/scribes.html. 2004.

[50] KATZ, J.; SHIN, J. Parallel and Concurrent Security of the HB and HB+ Protocols. *EUROCRYPT 2006*. pp. 73–87. 2006.

[51] KILIAN, J. Founding Cryptography on Oblivious Transfer. *20th ACM STOC*. pp. 20–31. 1988.

[52] KOBARA, K.;MOROZOV, K.;OVERBECK, R. Oblivious Transfer via McEliece's PKC and Permuted Kernels. *Cryptology ePrint Archive 2007/382*. 2007.

[53] LINDELL, Y. A Simpler Construction of CCA2-Secure Public-Key Encryption under General Assumptions. *EUROCRYPT 2003*. pp. 241–254. 2003.

[54] MCELIECE, R. The Theory of Information and Coding. *Vol. 3 of The Encyclopedia of Mathematics and Its Applications.*, Reading, Mass., Addison-Wesley. 1977.

[55] MCELIECE, R. A Public-Key Cryptosystem Based on Algebraic Coding Theory. *In Deep Space Network progress Report*. 1978.

[56] NAOR, M. Bit Commitment Using Pseudorandomness. *J. Cryptology 4(2)*. pp. 151–158. 1991.

[57] NAOR, M.; PINKAS, B. Efficient Oblivious Transfer Protocols. *SODA 2001*. pp. 448–457. 2001.

[58] NAOR, M.; YUNG, M. Universal One-Way Hash Functions and their Cryptographic Applications. *21st STOC*. pp. 33–43. 1989.

[59] NIEDERREITER, H. Knapsack-type Cryptosystems and Algebraic Coding Theory. *Prob. of Control and Inf. Theory*. Vol. 15(2). pp. 159–166. 1986.

[60] PASS, R.; SHELAT, A.; VAIKUNTANATHAN, V. Construction of a Non-malleable Encryption Scheme from Any Semantically Secure One. *CRYPTO 2006*. pp. 271–289. 2006.

[61] PEIKERT, C.; WATERS, B. Lossy Trapdoor Functions and Their Applications. *STOC 2008*. pp. 187–196. 2008.

[62] RABIN, M. How to Exchange Secrets by Oblivious Transfer. Technical Memo TR-81, Aiken Computation Laboratory, Harvard University. 1981.

[63] RACKOFF, C.; SIMON, D. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. *CRYPTO 1991*. pp. 433–444. 1991

[64] REGEV, O. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. *37th STOC*. pp. 84–93. 2005.

[65] ROSEN, A.; SEGEV, G. Chosen-Ciphertext Security via Correlated Products. *TCC 2009*. pp. 419–436. 2009.

[66] SAHAI, A. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. *40th FOCS*. pp. 543–553. 1999.

[67] SENDRIER, N. Finding the Permutation Between Equivalent Linear Codes: The Support Splitting Algorithm. *IEEE Trans. Inf. Theory*. 46(4). pp. 1193–1203. 2000.

[68] SHAMIR, A. An Efficient Identification Scheme based on Permuted Kernels. *CRYPTO 1989*. pp. 606–609. 1990.

[69] SHOR, P. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *35nd Annual Symposium on Foundations of Computer Science*. pp. 124–134. 1994.

[70] WIESNER, S. Conjugate coding. *Sigact News*, Vol. 15, No. 1. pp. 78–88. 1983.

[71] YAO, A. Protocols for Secure Computations (Extended Abstract). *FOCS 1982*. pp. 160–164. 1982.

[72] YAO, A. How to Generate and Exchange Secrets (Extended Abstract). *FOCS 1986*. pp. 162–167. 1986.