Proposta de Esquemas de Criptografia e de Assinatura sob Modelo de Criptografia de Chave Pública sem Certificado

Denise Hideko Goya

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO GRAU DE MESTRE
EM
CIÊNCIAS

Área de concentração: Ciência da Computação

Orientador: Prof. Dr. Routo Terada

São Paulo, julho de 2006.

Proposta de Esquemas de Criptografia e de Assinatura sob Modelo de Criptografia de Chave Pública sem Certificado

Este exemplar corresponde à redação final
da dissertação, devidamente corrigida e
defendida por *Denise Hideko Goya*,
e aprovada pela comissão julgadora.

São Paulo, 11 de julho de 2006.

Banca examinadora:

- Prof. Dr. Routo Terada (orientador) – IME/USP
- Prof. Dr. Paulo S. L. M. Barreto – POLI/USP
- Prof. Dr. Ricardo Dahab – IC/UNICAMP

Agradecimentos

Agradeço a todos que ajudaram direta ou indiretamente na elaboração desta dissertação. Sem exceção, a todos meus familiares e amigos, colegas de classe e trabalho, por conselhos e apoio.

Gostaria de agradecer, em especial, ao meu orientador, professor Routo Terada, pois, sem sua participação, este trabalho não seria possível. Agradeço aos professores Paulo Barreto e Wilson Ruggiero, cujas sugestões durante o exame de qualificação foram muito importantes para o aprimoramento de nossas idéias, e ao professor Ricardo Dahab que participou da banca examinadora, com valiosos comentários. Fico muito grata aos amigos Eduardo Takeo Ueda e Mehran Misaghi pelos comentários finais a esta dissertação.

Por fim, agradeço, a todos professores com os quais mantive contato durante o programa de mestrado e graduação, pois contribuíram na minha formação de base, viabilizando a construção deste trabalho.

Resumo

Sob o modelo de criptografia de chave pública baseada em identidades (ID-PKC), a própria identidade dos usuários é usada como chave pública, de modo a dispensar a necessidade de uma infra-estrutura de chaves públicas (ICP), na qual o gerenciamento de certificados digitais é complexo.

Por outro lado, sistemas nesse modelo requerem uma entidade capaz de gerar chaves secretas. Essa entidade é conhecida por PKG *(Private Key Generator)*; ela possui uma chave-mestra e mantém custódia das chaves secretas geradas a partir dessa chave-mestra. Naturalmente, a custódia de chaves é indesejável em muitas aplicações.

O conceito de Criptografia de Chave Pública sem Certificado, ou *Certificateless Public Key Cryptography* (CL-PKC), foi proposto para que a custódia de chaves fosse eliminada, mantendo, porém, as características de interesse: a não necessidade de uma ICP e a eliminação de certificados digitais. CL-PKC deixa de ser um sistema baseado em identidades, pois é introduzida uma chave pública, gerada a partir de uma informação secreta do usuário.

Em um esquema CL-PKC, um adversário A pode substituir a chave pública de uma vítima por uma nova chave, digamos X, de modo que A conhece a correspondente informação secreta relacionada com X. Entretanto, mesmo nessas condições, A não é capaz de fraudar o sistema, isto é, não consegue decriptografar uma mensagem criptografada com a chave púbica original (se o esquema for de criptografia/decriptografia, ou *Certificateless Public Key Encryption*, CL-PKE), e é incapaz de forjar uma assinatura (se o esquema for de assinatura, *Certificateless Public Key Signature*, ou CL-PKS). Essa importante propriedade é alcançada graças ao fato de que a autoridade de confiança é a única capaz de gerar uma infomação que associa a identidade do usuário com seu par de chaves.

Nesta dissertação, apresentamos a construção de dois esquemas, um CL-PKE e um CL-PKS, baseados em emparelhamentos bilineares sobre curvas elípticas. Ambas propostas: (1) eliminam custódia de chaves; (2) dispensam certificados digitais; (3) são mais eficientes, sob certos aspectos, que esquemas anteriormente publicados; (4) e são seguros contra ataques adaptativos de texto cifrado escolhido (em CL-PKE) e contra ataques adaptativos de mensagem escolhida (em CL-PKS), sob o modelo de oráculos aleatórios.

Abstract

Under the model of Identity Based Cryptography (ID-PKC), the public key can be the user's identity, therefore it does not require a Public Key Infrastructure (PKI) with its complex management of Digital Certificates.

On the other hand, this system requires a Private Key Generator (PKG), a trusted authority who is in possession of a master key and can generate any of the private keys. In this way, PKG can exercise the so-called *key escrow*, which is undesirable in many applications.

The concept of Certificateless Public Key Cryptography (CL-PKC) was proposed in order to remove the *key escrow* characteristic of IBC, while it does not require PKI neither Digital Certificates to certify the public keys. CL-PKC is no more an IBC because public keys are introduced, to bind the identities with its secret keys.

In a CL-PKC scheme, an adversary A may replace the victm's public key with another one, say X, so that A knows the private key corresponding to X; but still A is not able to decrypt the message encrypted with the original published public key (in a Certificateless Public Key Encryption, CL-PKE scheme), neither to forge signatures (in a Certificateless Public Key Signature, CL-PKS scheme). This important property is accomplished by the fact that only the PKG can bind the key pair for any other entity with that entity.

In this thesis we construct two schemes, one CL-PKE and one CL-PKS, based on bilinear pairing functions which: (1) does not allow *key escrow* by the PKG; (2) does not require Digital Certificates; (3) is more efficient, in some aspects, than previously published CL-PKE and CL-PKS schemes; (4) and is secure in the sense that it is strong against adaptive chosen ciphertext attacks (in CL-PKE) and adaptive chosen message attacks (in CL-PKS), under Random Oracle Model.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

CBE – Criptografia de Chave Pública Baseda em Certificado

CCA – Ataque por Texto Ilegível Escolhido

CCA2 – Ataque Adaptativo por Texto Ilegível Escolhido

CL-PKC – Criptografia de Chave Pública sem Certificado

CL-PKE – Esquema de Criptografia e Decriptografia no modelo de Criptografia de Chave Pública sem Certificado

CL-PKS – Esquema de Assinatura no modelo de Criptografia de Chave Pública sem Certificado

CMA – Ataque Adaptativo de Mensagem Escolhida

CPA – Ataque por Texto Legível Escolhido

EUF-CMA – Existencialmente não-forjável sob Ataque Adaptativo de Mensagem Escolhida

IBE – Esquema de Criptografia e Decriptografia no modelo de Criptografia de Chave Pública Baseada em Identidades

IBS – Esquema de Assinatura no modelo de Criptografia de Chave Pública Baseada em Identidades

ICP – Infra-estrutura de Chaves Públicas

ID-PKC – Criptografia de Chave Pública Baseada em Identidades

IND-CCA2 – Incapacidade de Distinção sob Ataque Adaptativo por Texto Ilegível Escolhido

KGC – Centro de Geração de Chaves

PKC – Criptografia de Chave Pública

PKG – Gerador Chaves Privadas

Capítulo 1

Introdução

Uma das principais preocupações de usuários de sistemas de comunicação que envolvem o uso de computadores é aquela que genericamente chamamos de segurança dos dados. Garantia de sigilo das informações que trafegam em uma rede de computadores é requisito primordial para grande parte das aplicações. Tal sigilo pode ser assegurado com o uso de técnicas de criptografia.

A palava criptografia tem origem grega, resultado da união de *kryptós* (escondido) e *grápho* (grafia), ou seja, escrita secreta. Os mais primitivos protocolos de criptografia, que adotam esse significado da palavra, são milenares e incluem um procedimento (algoritmo) para cifrar e decifrar textos. Em geral, o funcionamento desse procedimento depende de uma informação secreta, que deve ser compartilhada entre os que trocam mensagens cifradas.

Protocolos dessa natureza são chamados de protocolos de *criptografia de chave secreta*, ou *criptografia simétrica*. São capazes de garantir sigilo das mensagens entre os que conhecem a chave secreta, pois só com ela é possível cifrar e decifrar textos, ou, como costuma-se dizer, criptografar e decriptografar mensagens.

Entretanto, para muitas aplicações, não é suficiente simplesmente a certeza de que os dados trafegados são lidos (e compreendidos) exclusivamente pelos interlocutores, remetente e receptor. Segurança da informação abrange também as seguintes noções:

Autenticidade. É desejável que o remetente e o receptor das mensagens sejam autênticos, isto é, um intruso não deve poder se passar por outro usuário. Há aplicações em que é necessário que o destinatário seja o único capaz de ler o dado (a isto chamamos *autenticação de destino*); ou é preciso se ter a certeza de que quem enviou o dado é quem realmente diz ser (a isto chamamos *autenticação de origem*).

Integridade. Remetente e/ou receptor querem garantia de que o dado não foi adulterado (tentativa de fraude), nem durante a comunicação, nem mesmo depois, quando eventualmente houver gravação em arquivo (a isto chamamos *detecção de integridade de informação*).

Com criptografia simétrica, quando mais de duas pessoas conhecem a chave secreta, não há como assegurar autenticidade. Também não há garantia de integridade. Outra dificuldade em criptografia simétrica é como combinar a chave secreta, de modo que nenhum intermediário a descubra.

Autenticidade, integridade, combinação segura de chave, além da premissa sigilo, são características que podem ser obtidas com a implementação de sistemas de *criptografia de chave assimétrica*, em que há o envolvimento de um par de chaves: uma é usada para cifrar, outra para decifrar. A chave de ciframento é tornada pública, enquanto a de decriptografia deve ser mantida em segredo. Criptossistemas de chave assimétrica também são conhecidos por sistemas de *criptografia de chave pública*.

1.1 Chaves Públicas e Certificados

O modelo de chave pública teve origem em (DIFFIE; HELLMAN, 1976). O protocolo apresentado naquele trabalho permite que duas pessoas combinem uma chave secreta (para posterior uso em protocolo simétrico). O acordo de chave se dá por meio de um canal de comunicação, porém sem garantia de autenticação nem de origem, nem de destino. Essa dupla ausência de autenticidade possibilita uma ofensiva ao sistema, chamado ataque *homem-na-linha*[1], no qual um impostor, no meio da linha de comunicação, intercepta as mensagens alterando-as. Esse falso usuário personifica o receptor perante o emissor e ainda se faz passar pelo emissor perante o receptor. Dessa forma, o impostor se torna capaz de decifrar as mensagens trocadas.

Sob o protocolo de (DIFFIE; HELLMAN, 1976), o ataque *homem-na-linha* torna-se viável porque as chaves públicas não estão vinculadas às respectivas identidades. Uma pequena modificação nesse protocolo produz um modelo muito útil, aparentemente inatacável, que é utilizado até os dias atuais. A modificação requer a inclusão de um valor secreto adicional, segredo que só um terceiro conhece, uma entidade de confiança entre os interlocutores.

A partir de então, o protocolo de Diffie-Hellman deu origem a algoritmos variantes, chamados genericamente de algoritmos de criptografia de chave pública. Com o objetivo de associar univocamente a chave pública a uma entidade, foi sugerido por (KOHNFELDER, 1978) a criação de uma mensagem assinada por uma autoridade de confiança, contendo a representação da identidade, a correspondente chave pública e um identificador temporal (período

[1] Do inglês, *man-in-the-middle*.

de validade). Essa mensagem que associa uma chave pública ao respectivo dono é chamada *certificado*, ou *certificado digital*.

Portanto, para impedir ataques *homem-na-linha*, sistemas de criptografia de chave pública requerem o gerenciamento de certificados. A infra-estrutura que gerencia chaves públicas e entidades que emitem certificados é chamada infra-estrutura de chaves públicas (ICP, ou PKI – *Public Key Infrastructure*). Na prática, porém, a manutenção de ICPs é difícil e custosa: (BERBECARU; LIOY; MARIAN, 2001; GUTMAN, 2002; ELLISON; SCHNEIER, 2000) são apenas algumas referências que tratam das dificuldades relacionadas ao gerenciamento de certificados.

Para exemplificar uma limitação em ICPs, podemos citar a pouca flexibilidade em se trocar chaves secretas. Uma vez que a chave pública é dependente do valor secreto, o certificado só pode ser gerado e distribuído posteriormente à ação de duas entidades: primeiro, o usuário deve gerar (ou recriar) o par de chaves; a seguir, a autoridade de confiança precisa emitir, assinar e distribuir o novo certificado. Ademais, a infra-estrutura de chaves públicas requer um controle de certificados revogados (certificados que se tornam inválidos, ainda dentro de seu período de validade). No contexto de uma aplicação sobre uma ICP, um usuário que queira transmitir uma mensagem cifrada deve, antes, obter o certificado do destinatário (que deve ser válido e não-revogado) para, então, cifrar e transmitir.

Em (SHAMIR, 1984), foi apresentado o conceito de criptografia de chave pública baseada em identidades, ID-PKC[2], em que identificadores como nome, endereço de e-mail ou CPF, funcionam como chave pública. Desse modo, automaticamente se cria um vínculo entre chave pública e respectivo dono, mudando o paradigma de certificados e infra-estrutura de chave pública.

O primeiro protocolo de criptografia e decriptografia que foi demonstrado seguro e ao mesmo tempo eficiente, do ponto de vista computacional, foi o esquema de (BONEH; FRANKLIN, 2001), que se valeu de emparelhamentos bilineares sobre grupos baseados em curvas elípticas, tratados originalmente em (SAKAI; OHGISHI; KASAHARA, 2000).

A simplicidade conceitual em torno de protocolos de criptografia e decriptografia de chave pública baseados em identidades (ou IBE, de *Identity Based Encryption*) é bastante atraente. Para aplicações em correio eletrônico ou telefonia móvel, por exemplo, as mensagens poderiam ser transmitidas com sigilo e de forma transparente aos usuários: o remetente precisaria apenas conhecer o identificador do destinatário (o endereço de e-mail ou o número telefônico) — situação bem diferente do que ocorre em criptografia de chave pública convencional, em que a chave pública normalmente é uma cadeia binária de difícil memorização, pois guarda uma relação matemática não trivial com a chave secreta de decriptografia.

Uma característica inerente aos modelos de Shamir e Boneh-Franklin é a de *custódia de*

[2]Do inglês, *Identity Based Public Key Cryptography*.

chaves[3], em que a autoridade que gerencia o sistema tem o conhecimento de todas chaves secretas associadas a cada identidade. Trata-se de uma propriedade indesejável em muitas aplicações, onde se faz necessária a certeza de que somente remetente e receptor têm conhecimento de um determinado sigilo. Surgiram, então, várias propostas que estendem ou modificam ligeiramente o esquema de Boneh-Franklin, com o objetivo de eliminar tal característica.

Uma derivação sobre (BONEH; FRANKLIN, 2001) deu origem ao trabalho de (AL-RIYAMI; PATERSON, 2003), que introduziu a noção de criptografia de chave pública sem certificado (ou *Certificateless Public Key Cryptography*, CL-PKC). Trata-se de um modelo em que, assim como no ID-PKC, não há a necessidade de certificados para relacionar chaves públicas com suas entidades. Uma das vantagens presentes em CL-PKC é a eliminação da custódia de chaves, ou seja, a autoridade de confiança é incapaz de decriptografar os textos cifrados de seus usuários. No entanto, sob CL-PKC a chave pública deixa de ser simplesmente a identidade do usuário. Desse modo, CL-PKC não é um sistema de criptografia de chave pública baseado em identidade e perde uma das características mais atraentes em ID-PKC: a simplicidade da chave de criframento. Felizmente, o custo dessa perda é fortemente compensado por outras vantagens, como irretratabilidade e diminuição do poder da chave-mestra, conforme descrito a seguir.

Em ID-PKC, a autoridade de confiança, por conhecer as chaves secretas, pode forjar assinaturas para quaisquer de seus usuários. Um esquema de assinatura é um protocolo de criptografia de chave pública que garante integridade da informação e autenticação de origem. Em sistemas convencionais de criptografia de chave pública, quem assina um documento não pode negar que o assinou (irretratabilidade). Para que ID-PKC possa oferecer a propriedade de irretratabilidade aos esquemas de assinatura, é necessário que se implementem modificações sobre o IBE de (BONEH; FRANKLIN, 2001). O CL-PKC de (AL-RIYAMI; PATERSON, 2003) dá origem a protocolo de assinatura com garantia de irretratabilidade.

Outra característica indesejável em ID-PKC é que se a chave-mestra de geração de chaves secretas for comprometida (descoberta por alguém não autorizado), as conseqüências serão desastrosas para todo o sistema. Em CL-PKC, a perda da chave-mestra não compromete o sigilo das mensagens cifradas.

1.2 Motivações

Se, de um lado, a infra-estrutura de chaves públicas, necessária em sistemas de criptografia de chave pública convencionais, é de difícil gerenciamento, de outro lado, a criptografia baseada em identidades elimina a exigência de ICPs, porém enfraquece alguns aspectos de segurança, com a custódia de chaves, a não irretratabilidade e o alto grau de criticidade da chave-mestra.

[3]Do inglês, *key escrow*.

O modelo CL-PKC de criptografia de chave pública sem certificado melhora características de segurança de ID-PKC, ao mesmo tempo que dispensa a necessidade de uma infra-estrutura de chaves públicas. Trata-se, portanto, de um modelo que merece ser explorado e desenvolvido.

1.3 Objetivos e Organização da Dissertação

Nosso trabalho tem por objetivo descrever o sistema de criptografia de chave pública sem certificado de (AL-RIYAMI; PATERSON, 2003) e apresentar protocolos sob o modelo, construídos a partir da derivação de esquemas existentes.

Os esquemas propostos devem ser praticáveis e comprovados seguros, de acordo com alguma noção de segurança. Por fim, vantagens e desvantagens das propostas são levantadas em relação a esquemas existentes.

Esta dissertação está organizada da seguinte maneira:

- no capítulo 2, encontram-se notações e fundamentos matemáticos utilizados na dissertação;
- no capítulo 3, há definições sobre os sistemas de criptografia de chave pública e os conceitos relativos à segurança desses criptossistemas;
- no capítulo 4, são apresentadas as definições formais de esquemas de criptografia e de assinatura no modelo de criptografia de chave pública sem certificado e adversários contra o modelo;
- no capítulo 5, são apresentadas nossas propostas de esquemas em criptografia de chave pública sem certificado, um primeiro esquema para criptografia e decriptografia e outro para assinatura;
- no capítulo 6, são tratadas as demonstrações de segurança dos esquemas propostos;
- no capítulo 7, são elaboradas comparações dos esquemas propostos com os trabalhos existentes;
- no capítulo 8, seguem nossas contribuições e sugestões de trabalhos futuros;
- no apêndice A, são descritos trabalhos relacionados ao modelo de criptografia de chave pública sem certificado.
- no apêndice B, é descrita uma segunda proposta de CL-PKE, mais eficiente que a primeira proposta, porém ainda sem demonstrações de segurança.

Capítulo 2

Fundamentos Matemáticos

Neste capítulo são apresentados fundamentos matemáticos importantes para a compreensão dos conceitos em criptografia de chave pública e de algumas das características do esquemas estudados.

2.1 Notações

Seguem algumas notações adotadas ao longo da dissertação:

- $\mathbb{Z}_q$ denota o conjunto de inteiros módulo q, isto é, $\{0, 1, \ldots, q-1\}$, onde o resultado das operações aritméticas elementares de $\mathbb{Z}$ é reduzido módulo q.
- $\mathbb{Z}_q^* = \mathbb{Z}_q \backslash \{0\}$.
- $\mathbb{Z}[x]$ denota o conjunto dos polinômios na variável x, com coeficientes em $\mathbb{Z}$.
- $\mathbb{G}^* = \mathbb{G} \backslash \{O\}$, onde O denota o elemento identidade do grupo $\mathbb{G}$, conforme seção 2.2.
- $\{0,1\}^n$ representa o conjunto de cadeias binárias de comprimento n.
- $\{0,1\}^*$ representa o conjunto de cadeias binárias de comprimento arbitrário.
- $(a \parallel b)$ representa a concatenação das cadeias a e b.
- $(a \oplus b)$ denota a operação *ou-exclusivo*, bit a bit, com as cadeias a e b.

2.2 Grupo

Um *grupo* é um conjunto não vazio $\mathbb{G}$, dotado de uma operação binária $\circ : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}$, que satisfaz as seguintes propriedades (KOBLITZ, 1994):

- Elemento Identidade: $\exists n \in \mathbb{G} : \forall a \in \mathbb{G} : a \circ n = n \circ a = a$;
- Elemento Inverso: $\forall a \in \mathbb{G} : \exists \bar{a} \in \mathbb{G} : a \circ \bar{a} = n$;
- Associatividade: $\forall a, b, c \in \mathbb{G} : (a \circ b) \circ c = a \circ (b \circ c)$;
- Fechamento: $\forall a, b \in \mathbb{G}, a \circ b \in \mathbb{G}$.

Um *grupo abeliano* (também chamado *grupo comutativo*) é um grupo com a propriedade adicional:

- Comutatividade: $\forall a, b \in \mathbb{G} : a \circ b = b \circ a$.

Quando o símbolo $\circ$ é substituído pelo sinal de adição "+", dizemos que o grupo é um *grupo aditivo*. Nesse caso, o elemento neutro (identidade) é escrito como 0 e o inverso de a é escrito como $-a$. Define-se *multiplicação por escalar*, para um inteiro positivo λ e $a \in \mathbb{G}$, como sendo a operação $\lambda a = a + a + \ldots + a$, onde a soma consiste de λ termos.

Quando o símbolo $\circ$ é substituído pelo sinal de multiplicação ".", dizemos que o grupo é um *grupo multiplicativo*. Nesse caso, o elemento identidade é escrito como 1 e o inverso de a é escrito como a^{-1}. Define-se *potenciação*, para um inteiro positivo λ e $a \in \mathbb{G}$, como sendo a operação $a^{\lambda} = a.a. \ \ldots \ .a$, com λ fatores.

Quando o número de elementos de $\mathbb{G}$ é finito, esse número é chamado *ordem* de $\mathbb{G}$.

Um grupo aditivo é dito *grupo cíclico* se existe um elemento $P \in \mathbb{G}$ tal que qualquer elemento $Q \in \mathbb{G}$ pode ser escrito como múltiplo escalar de P, isto é, $Q = \lambda P$, para algum inteiro λ. Um elemento P com tal propriedade é chamado *gerador* de $\mathbb{G}$.

2.3 Corpo

Um *corpo* é uma estrutura algébrica que consiste de um conjunto K e duas operações binárias (KOBLITZ, 1994):

- (Adição) $+ : K \times K \rightarrow K$
- (Multiplicação) $\cdot : K \times K \rightarrow K$

satisfazendo as seguintes propriedades:

- $(K, +)$ é um grupo abeliano, com identidade aditiva denotada por 0.
- $(K^*, \cdot)$, onde $K^* = K \backslash \{0\}$, é um grupo abeliano, com identidade multiplicativa denotada por 1.
- $0 \neq 1$.
- Distributividade: $x \cdot (y + z) = x \cdot y + x \cdot z, \forall x, y, z \in K$

A *característica de um corpo* K, denotada por $char(K)$, é o número de vezes que a identidade multiplicativa de K deve ser adicionada a si mesma para que o resultado seja igual à identidade aditiva. Se o resultado nunca for a identidade aditiva, define-se $char(K) = 0$.

Dizemos que existe um *isomorfismo* entre dois corpos quando há uma função bijetora entre eles, que, a grosso modo, mapeia os elementos identidade e os inversos de um corpo no outro.

Alguns resultados de interesse (KOBLITZ, 1994):

- A característica de um corpo é sempre um número primo ou é zero.
- Se $char(K) = 0$, então K contém um subconjunto isomorfo a $\mathbb{Q}$.
- Se $char(K) = p$, para algum primo p, então K contém um subconjunto isomorfo a $\mathbb{Z}_p$.
- O número de elementos de um corpo finito é sempre na forma p^m, onde p é a característica do corpo e m, um inteiro positivo.
- Para cada primo p e cada inteiro m, existe um único corpo finito (a menos de isomorfismo) com p^m elementos, denotado por $GF(p^m)$ ou por $\mathbb{F}_{p^m}$.
- Vale observar que $GF(p)$ é isomorfo a $\mathbb{Z}_p$. O corpo $GF(p^m)$ consiste dos polinômios com coeficientes em $\mathbb{Z}_p$ e grau menor que m, módulo um polinômio redutível de grau m. Para aplicações em criptografia (e computacionais em geral), os corpos $GF(2^m)$ e $GF(p)$ são os de maior interesse.

2.4 Curvas Elípticas

Curvas elípticas aplicadas à criptografia foram sugeridas independentemente por (KOBLITZ, 1987) e (MILLER, 1985), com base na segurança aparentemente superior que se pode alcançar (ver seção 3.5.2) e na possibilidade do uso de chaves criptográficas de tamanhos significativamente menores quando comparado a criptossistemas de outros modelos (ver seção 2.4.4). As

curvas elípticas são a base dos esquemas de criptografia aqui estudados. Os conceitos apresentados nesta seção foram extraídos de (BARRETO, 1999), (KOBLITZ, 1994) e (TERADA, 2000).

Considere um corpo K, os elementos $a, b, c, d, e \in K$, e a equação:

$$y^2 + axy + by = x^3 + cx^2 + dx + e \tag{2.1}$$

Definimos uma *curva elíptica* sobre K, denotada $E(K)$, como o conjunto dos pares $(x, y) \in K \times K$ que satisfazem a equação 2.1, mais um ponto $\mathfrak{O}$, chamado de ponto no infinito.

O número de pontos de uma curva elíptica E (isto é, o número de soluções da equação da curva mais o ponto no infinito) é chamado *ordem da curva* e é denotado por $\#E$. Para corpos finitos $GF(q)$, o *Teorema de Hasse* diz que a ordem da curva satisfaz:

$$q + 1 - 2\sqrt{q} \leq \#E \leq q + 1 + 2\sqrt{q}$$

A equação 2.1 pode ser simplificada por meio de mudanças de coordenadas. Se a característica de K for 2 (isto é, para $GF(2^m)$), a equação é reduzida a uma das duas formas:

$$y^2 + xy = (x^3 + ax^2 + b) \tag{2.2}$$
$$y^2 + cy = (x^3 + ax + b) \tag{2.3}$$

Para corpos de característica 3 (isto é, para $GF(3^m)$), a equação pode ser reescrita como

$$y^2 = (x^3 + ax^2 + bx + c) \tag{2.4}$$

Para $p > 3$, $GF(p^m)$), a equação 2.1 pode ser reescrita como

$$y^2 = (x^3 + ax + b) \tag{2.5}$$

Para que os pontos da curva elíptica formem um grupo, precisamos definir uma operação binária sobre eles, e especificar os elementos identidade e inverso. A construção da lei de grupo para curvas sobre $\mathbb{R}$ tem caráter geométrico, com base em retas secante e tangente à curva. Em certos tipos de curva não é possível traçar uma reta tangente em todos os pontos da curva; nesses casos não é possível a construção de um grupo. Isso ocorre quando o polinômio $f(x) = x^3 + ax + b$ possui raízes múltipas, ou seja, o discriminate $\triangle \equiv 4a^3 + 27b^2$ é igual a zero. Sobre $\mathbb{R}$, existem as chamadas curvas *singulares* ou *degeneradas*, em que o discriminante sempre é zero; elas possuem a forma geral $y^2 = x^3 - 3t^2x + 2t^3$. Quando o corpo é finito, os discriminantes para cada equação simplificada são descritos na tabela 2.1; eles devem ser diferentes de zero (módulo $\#E$) para que seja possível a constituição de um grupo sobre os pontos da curva.

Tipo da curva	Discriminante
Equação 2.2	b
Equação 2.3	c^4
Equação 2.4	$a^2(b^2 - ac) - b^3$
Equação 2.5	$-16(4a^3 + 27b^2)$

Tabela 2.1: Discriminantes de curvas elípticas

A curva elíptica é simétrica em relação ao eixo x. Define-se, então, o elemento inverso de um ponto $P = (x, y)$ como sendo o ponto simétrico $-P = (x, -y)$. O elemento identidade é o ponto no infinito, $\mathfrak{O}$. Em curvas com discriminante diferente de zero, a operação binária do grupo é a soma de dois pontos P e Q, definida da seguinte maneira:

- $P + \mathfrak{O} = \mathfrak{O} + P = P, \forall P$;
- Se $P = -Q$, então $P + Q = \mathfrak{O}$;
- Se $P \neq Q$, traça-se uma reta secante à curva, pelos pontos P e Q. Pode-se provar que essa reta sempre intercepta a curva num terceiro ponto, R; então define-se $P + Q = -R$;
- Se $P = Q$, traça-se uma reta tangente à curva pelo ponto P. Pode-se provar que essa reta sempre intercepta a curva num segundo ponto, R; então define-se $P + Q = P + P = 2P = -R$;

A partir da construção geométrica sobre $\mathbb{R}$, é possível deduzir expressões algébricas para o cálculo das coordenadas do ponto resultante da soma de outros dois pontos. Embora sobre corpos finitos as "curvas" sejam apenas pontos esparsos e não faz sentido falarmos em retas tangentes ou secantes, a idéia geométrica sobre $\mathbb{R}$ leva às expressões a seguir e induzem uma lei de grupo também quando K é finito. Como para aplicações criptográficas interessam apenas as curvas sobre $GF(2^m)$ ou sobre $GF(p)$ (por questões práticas e de eficiência), vamos exemplificar apenas alguns casos.

2.4.1 Soma de Pontos em Curva Elíptica sobre $GF(2^m)$

Considere que $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ dois pontos na curva elíptica definida pela equação 2.3: $y^2 + cy = (x^3 + ax + b)$, com $c \neq 0 \ mod \ 2^m$.

Então $-P = (x_1, y_1 + c)$. Para os casos em que P e Q não coincidem com o ponto no infinito, e quando P não é o inverso de Q, a soma $P + Q = (x_3, y_3)$ é dada por:

$$x_3 = \begin{cases} \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + x_1 + x_2 & \text{se} \quad P \neq Q \\ \frac{x_1^4+a^2}{c^2} & \text{se} \quad P = Q \end{cases}$$

$$y_3 = \begin{cases} \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1+x_3) + y_1 + c & \text{se} \quad P \neq Q \\ \left(\frac{x_1^2+a}{c}\right)(x_1+x_3) + y_1 + c & \text{se} \quad P = Q \end{cases}$$

Observe que quando $P = Q$, a soma é igual a $2P$, por isso é chamada de *duplicação de* P. Uma multiplicação escalar λP pode ser obtida com sucessivas duplicações (e algumas somas eventuais).

Para evitar as divisões por c, em geral, adota-se $c = 1$. Com $c = 1$, existem exatamente três classes isomorfas de curvas elípticas sobre $GF(2^m)$, com m ímpar. Uma curva representativa de cada uma dessas três classes é:

$$\begin{aligned} E_1 &: y^2 + y = x^3 \\ E_2 &: y^2 + y = x^3 + x \\ E_3 &: y^2 + y = x^3 + x + 1 \end{aligned}$$

2.4.2 Soma de Pontos em Curva Elíptica sobre $GF(p)$

Considere que $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ dois pontos na curva elíptica definida pela equação $y^2 = (x^3 + ax + b)$, com $4a^3 + 27b^2 \neq 0 \ mod \ p$. Então $-P = (x_1, -y_1)$. Para os casos em que P e Q não coincidem com o ponto no infinito, e quando P não é o inverso de Q, $P + Q = (x_3, y_3)$ é dado por:

$x_3 = t^2 - x_1 - x_2$

$y_3 = t(x_1 - x_3) - y_1$, onde

$$t = \begin{cases} \left(\frac{y_2-y_1}{x_2-x_1}\right) & \text{se} \quad P \neq Q \\ \left(\frac{3x_1^2+a}{2y_1}\right) & \text{se} \quad P = Q \end{cases}$$

Observe que quando $P \neq Q$, t é a inclinação da reta secante.

2.4.3 Considerações para Aplicações em Criptografia

Quando o grupo formado pelos pontos da curva elíptica for cíclico, tem-se a vantagem de que um gerador desse grupo é um facilitador do cálculo da soma de pontos. Por exemplo, se G for um gerador para uma curva de ordem q, então a soma dos pontos $P = \alpha G$ e $Q = \beta G$ equivale a $P + Q = \gamma G$, onde $\gamma = (\alpha + \beta) \bmod q$.

Para aplicações em criptografia, consideram-se curvas que contenham subgrupos cíclicos de ordem prima, para se evitar certos tipos de ataques conhecidos (como o de Pohlig-Hellman; ver seção 3.5.2). O ideal é que a própria ordem da curva seja prima, mas nem sempre é possível se alcançar essa condição. Também para aplicações criptográficas, é importante se evitar certos tipos particulares de curvas: as supersingulares e as anômalas.

Uma curva elíptica E sobre um corpo finito $GF(q)$, de característica p, é chamada *supersingular* se, e somente se, $p \mid t$, onde $\#E = q + 1 - t$.

Quando E é definida sobre $GF(p)$, com $p > 3$, é possível provar que E é supersingular se, e somente se, $t^2 \in \{0, p, 2p, 3p, 4p\}$, onde $\#E = p + 1 - t$.

Uma curva elíptica E sobre um corpo finito $GF(q)$ é chamada *anômala* se, e somente se, $\#E = q$.

Por essas definições, percebe-se que são poucas as curvas supersingulares ou anômalas, comparando-se com o total de curvas para o mesmo corpo finito.

Em (WIN; PRENEEL, 1998), são apresentadas referências e considerações a respeito da implementação das operações básicas sobre essas curvas.

2.4.4 Sobre os Tamanhos de Chaves

Uma das vantagens que o modelo de criptografia baseado em curvas elípticas traz sobre os demais é a possibilidade do uso de chaves criptográficas de tamanhos significativamente menores. Na tabela 2.2 são comparados os tamanhos de chave necessários, para um esforço computacional de tentativa de encontrar a chave secreta por métodos de exaustão, para sistemas de criptografia de chave pública (sobre curvas elípticas ou RSA[1]), em relação a tamanhos fixos de chave num modelo de criptografia simétrica[2].

Os dados da tabela 2.2 foram extraídos de (BARRETO, 1999) e (CERTICOM, 2004). É importante observar nessa tabela o comportamento praticamente exponencial do tamanho de

[1] Uma definição de criptografia de chave pública, adotada pelo esquema RSA (RIVEST; SHAMIR; ADLEMAN, 1978), é descrita na seção 3.1.

[2] Conceitos de criptografia simétrica são abordados no capítulo introdutório.

Modelo de criptografia		
Simétrico	Elíptico	RSA
80	163	1024
128	256	3072
192	384	7680
256	512	15360

Tabela 2.2: Comparação entre tamanhos de chaves

chave para RSA, quando comparado ao de sistemas sobre curvas elípticas.

2.5 Mapeamento Bilinear e Emparelhamento

Sejam $\mathbb{G}_1$ e $\mathbb{G}_2$ grupos cíclicos de ordem prima q. Considere $\mathbb{G}_1$ um grupo aditivo e $\mathbb{G}_2$ um grupo multiplicativo.

Um mapeamento $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ é dito um *mapeamento bilinear* se é satisfeita a igualdade $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$, para todo $P, Q \in \mathbb{G}_1$ e $a, b \in \mathbb{Z}_q$.

Observe que a propriedade de bilinearidade acima implica também as igualdades $\hat{e}(aP, Q) = \hat{e}(P, aQ) = \hat{e}(P, Q)^a$, que são amplamente aproveitadas para se definirem os esquemas criptográficos aqui estudados.

Um mapeamento $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ é dito um *mapeamento bilinear admissível* se satisfaz as propriedades (BONEH; FRANKLIN, 2001):

1. Bilinearidade: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$, para todo $P, Q \in \mathbb{G}_1$ e $a, b \in \mathbb{Z}_q$;
2. Não-degeneração: o mapeamento não leva todos os pares de $\mathbb{G}_1 \times \mathbb{G}_1$ para a identidade de $\mathbb{G}_2$. Como os grupos possuem ordem prima, é equivalente dizer: para um gerador P de $\mathbb{G}_1$, $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$, isto é, $\hat{e}(P, P)$ é um gerador de $\mathbb{G}_2$;
3. Eficiência computacional: existe um algoritmo de complexidade de tempo polinomial que calcula $\hat{e}(P, Q)$, para todo $P, Q \in \mathbb{G}_1$.

Chamamos de *emparelhamento bilinear* (ou simplesmente *emparelhamento*) um tal mapeamento bilinear admissível[3] (AL-RIYAMI; PATERSON, 2005). Exemplos de emparelhamentos

[3] A rigor, a definição de emparelhamento (ou pareamento, de *pairing*) é mais genérica e envolve outras estruturas algébricas. Para os propósitos dos trabalhos aqui discutidos, a definição mais restritiva é conveniente.

desse tipo são os de Tate e Weil, que adotam certos grupos sobre pontos de curvas elípticas (BONEH; FRANKLIN, 2001).

Emparelhamentos assim definidos são ditos *emparelhamentos simétricos* e, em geral, só estão disponíveis sobre curvas supersingulares.

Essa definição pode ser generalizada para o que chamamos de *emparelhamentos assimétricos* (aqui denotados por e), com $e : G_1 \times G_2 \rightarrow G_T$, em que G_1, G_2 e G_T são grupos de mesma ordem prima, e $G_1 \neq G_2$. Nesse caso, há dois geradores, $P \in G_1^*, Q \in G_2^*$, tais que $P = \psi(Q)$, onde ψ é um homomorfismo de G_2^* em G_1^*, eficientemente computável (isto é, de complexidade de tempo polinomial).

2.6 Notação $O()$ e Função Ínfima

Considere funções $f(n)$ definidas sobre inteiros $n > 0$, com valores reais $f(n) > 0$, para infinitos valores de n. Escrevemos $f(n) = O(g(n))$ se existem uma constante $c > 0$ e um inteiro n_0, tais que $0 \leq f(n) \leq cg(n)$, para todo $n > n_0$. Ou seja, para n suficientemente grande, $f(n)$ não cresce mais que $g(n)$, a menos de um fator constante.

Uma função $f : \mathbb{N} \rightarrow \mathbb{R}$ é dita *ínfima*[4] se, para todo polinômio p, existe um inteiro n_0, tal que $\mid f(n) \mid \leq 1/ \mid p(n) \mid$, para todo $n \geq n_0$ (DENT; KUDLA, 2005).

2.7 Complexidade de Algoritmo

Seja n o comprimento dos dados de entrada para um algoritmo A e seja $f(n)$ a função do tempo de execução, no pior caso, de A.

Dizemos que A tem complexidade de tempo *polinomial*, se $f(n) = O(n^k)$, para alguma constante k.

Dizemos que A tem complexidade de tempo *exponencial*, se $f(n) = O(k^n)$, para alguma constante $k > 1$.

Dizemos que um problema é *computacionalmente inviável, difícil* ou *intratável*, se não se conhece qualquer algoritmo de tempo polinomial para resolvê-lo.

[4]Do original inglês, *negligible*.

2.8 Resumo

Neste capítulo foram apresentados fundamentos matemáticos referenciados nos próximos capítulos.

Capítulo 3

Conceitos em Criptografia de Chave Pública

Neste capítulo são apresentadas definições de criptografia de chave pública (determinística e um preâmbulo para criptografia probabilística), além de noções de segurança relacionadas. Para as definições de segurança, primeiramente se fazem necessárias abordagens sobre o papel da teoria da complexidade na criptografia e o estudo de alguns problemas matemáticos.

3.1 Criptografia Determinística de Chave Pública

O conceito de criptografia de chave pública foi possível a partir do artigo seminal de (DIFFIE; HELLMAN, 1976). Um criptossistema de chave pública é composto das seguintes partes (RIVEST, 1990):

Um algoritmo de geração de chaves. Este algoritmo recebe como entrada um parâmetro de segurança k, produz uma chave pública E e a correspondente chave secreta D;

Um algoritmo de criptografia. Este algoritmo recebe como entrada uma chave pública E e uma mensagem M, e gera um texto cifrado C, denotado por $E(M)$. Esta operação é *um-para-um*;

Um algoritmo de decriptografia. Este algoritmo recebe como entrada uma chave secreta D e um texto cifrado C, e produz uma mensagem $M = D(C)$.

Esses algoritmos possuem as seguintes propriedades:

1. Para toda mensagem $M, D(E(M)) = M$;
2. Para toda mensagem $M, E(D(M)) = M$;
3. Os algoritmos de geração de chaves, de criptografia e decriptografia são executados em tempo polinomial no tamanho de suas entradas;
4. O algoritmo de geração de chaves é um algoritmo probabilístico (sua saída depende de um sorteio); os algoritmos de criptografia e decriptografia são determinísticos (sempre terminam e produzem a mesma resposta);
5. Dada uma chave pública E, mas não a chave secreta D, a chance de que um adversário de tempo polinomial possa decriptografar um texto cifrado $C = E(M)$ é menor que qualquer fração polinomial (pelo menos k^{-c}, para uma constante c, com k suficientemente grande; veja a seção 3.4, para uma referência de como o estudo da complexidade computacional direciona a construção de sistemas de criptografia).

A quinta propriedade é descrita alternativamente em termos de *funções unidirecionais com segredo*[1]. Uma função é *unidirecional* se for computacionalmente viável calculá-la e computacionalmente inviável computar sua inversa. Dizemos que uma função unidirecional é *com segredo* se existe uma informação (o "segredo", a chave de decriptografia) que torna a computação de sua inversa viável (LUCCHESI, 1984).

Sistemas que seguem o modelo acima, são chamados de sistemas de criptografia determinística de chave assimétrica, ou criptografia determinística de chave pública. Aqui, um exemplo clássico que se pode citar é o RSA (RIVEST; SHAMIR; ADLEMAN, 1978).

3.2 Adversários e Ataques

Algoritmos criptográficos basicamente objetivam "esconder" informações sigilosas de pessoas desautorizadas a lê-las, isto é, de quem não conhece a chave secreta. Chamamos de *criptanalista* um especialista em criptografia que analisa um algoritmo ou criptossistema, sem o conhecimento da chave secreta, com a finalidade de descobrir uma vulnerabilidade no objeto de estudo (TERADA, 2000).

Um *adversário* é um algoritmo probabilístico (GOLDWASSER; BELLARE, 2001), construído por um criptanalista, com o intuito de extrair alguma informação útil para os seus propósitos. Informalmente, dizemos que um *ataque* a um criptossistema é uma ofensiva aplicada por um adversário.

[1] Do original inglês, *trapdoor one-way function*.

Numa forma mais simples de ataque, o adversário é passivo e simplesmente observa usuários legítmos usando um sistema. Ataques potencialmente mais poderosos permitem que o adversário atue ativamente, manipulando o canal de comunicação; ações que são permitidas a usuários reais são possíveis ao adversário (RIVEST, 1990).

Freqüentemente se usa a expressão "quebrar um criptossistema". Uma definição acerca dessa expressão, em (MENEZES; VANSTONE; OORSCHOT, 1996), considera que um esquema de criptografia é dito "*quebrável*"se um terceiro, sem conhecimento do par de chaves (E, D), pode sistematicamente recuperar textos legíveis a partir do texto cifrado correspondente, num tempo apropriado.

Os objetivos de um criptanalista são: obter o texto legível de um ilegível interceptado ou descobrir a chave secreta.

As formas de ataque são classificadas como segue, em ordem decrescente de insegurança (TERADA, 2000):

1. *Ataque por só-texto-ilegível*: o criptanalista tenta adquirir conhecimento útil analisando apenas um ou mais textos crifrados.
2. *Ataque por texto legível conhecido*: o criptanalista possui e analisa pares (x, y) de legível e ilegível correspondentes.
3. *Ataque por texto legível escolhido*: além do suposto no tipo anterior, o criptanalista pode escolher os legíveis x e obter os y correspondentes. A escolha é por valores de x que apresentem alguma característica estrutural que aumenta o conhecimento do algoritmo e da chave em uso[2].
4. *Ataque adaptativo por texto legível escolhido*: além do suposto no tipo anterior, a escolha de um novo x pode depender dos ilegíveis analisados anterioremente. Desta forma, a escolha de um novo x é condicionada ao conhecimento já adquirido.
5. *Ataque por texto ilegível escolhido*: o criptanalista escolhe inicialmente o ilegível y e então obtém o legível x correspondente. Supõe-se que o criptanalista tem acesso apenas ao algoritmo de decriptografia (sem conhecimento da chave) e seu objetivo é, mais tarde, sem ter acesso à decriptografia, ser capaz de deduzir o x correspondente a um y novo.
6. *Ataque adaptativo por texto ilegível escolhido*: além do suposto no tipo anterior, a escolha de um novo y pode depender dos ilegíveis analisados anterioremente.

[2]Durante a Segunda Guerra mundial, os norte-americanos enviaram, de Pearl Harbor para o continente, uma mensagem bem escolhida, sem cifrar, relatando problemas no abastecimento de água. Os japoneses, na escuta, captaram a mensagem e a retransmitiram cifrada para Tóquio. Isto permitiu aos americanos determinar a chave dos japoneses (LUCCHESI, 1984).

Os ataques acima enumerados, com exceção do primeiro, permite que o criptanalista tenha acesso aos algoritmos (de criptografia e/ou decriptografia, dependendo do caso). Costuma-se chamar o acesso ao algoritmo de decriptografia por consulta ao *oráculo de criptografia*, conforme descrito na seção 3.6. Não necessariamente o criptanalista é um mal-intencionado; pode ser apenas um especialista que deseja avaliar o grau de segurança do esquema.

Um requisito em projetos de criptossistemas deve ser a segurança de que um possível adversário não obtenha sucesso. O sucesso de um adversário, entretanto, pode ser definido de várias formas. Por exemplo, um objetivo modesto para um projetista seria que, para a maioria das mensagens, o adversário não possa derivar a mensagem inteira, a partir de seu texto cifrado. É modesto porque não leva em conta os seguintes problemas:

- O criptossistema pode não ser seguro para algumas distribuições de probabilidade sobre o espaço de mensagens[3] (por exemplo, o espaço de mensagens consiste de um número polinomial de mensagens, conhecidas pelo adversário);
- Informações parciais a respeito das mensagens podem ser facilmente calculadas a partir do texto cifrado;
- Pode ser fácil detectar fatos simples porém úteis a respeito do tráfego das mensagens (tais como, as mesmas mensagens são enviadas mais de uma vez).

Os criptossistemas de chave pública determinísticos (conforme descritos na seção 3.1), sofrem com esses problemas e ainda (RIVEST, 1990):

- Se m é uma mensagem fortemente estruturada, então $f(m)$ pode ser de fácil inversão (por exemplo, $f(0)$);
- Algumas informações a respeito de m podem ser fáceis de calcular, a partir de $f(m)$, como, por exemplo, a sua paridade (ou pior ainda: o cálculo do valor do bit menos significativo da chave secreta).

Preocupações como essas norteiam a construção de esquemas de chave pública. Deseja-se que um adversário não possa predizer qualquer informação a respeito de textos legíveis, correspondentes a textos cifrados conhecidos. Em caso contrário, o criptanalista poderia chegar à descoberta da chave secreta.

Essencialmente, essas preocupações regram as definições de segurança formal, discutidas na seção 3.6.

[3]Chamamos de *espaço de mensagens* o conjunto de cadeias (os textos legíveis) sobre um alfabeto.

3.3 Criptografia de Chave Pública – Nova Definição

Antes de abordarmos noções de segurança formal, faremos uma nova definição de criptografia de chave pública (não determinística), mais apropriada para contornar os problemas discutidos na seção 3.2. A definição a seguir é dada por (GOLDWASSER; BELLARE, 2001).

Um esquema de *criptografia de chave pública* é uma tripla (G, E, D), de algoritmos de tempo polinomial que satisfazem as seguintes condições:

Algoritmo de geração de chaves. G é um algoritmo probabilístico de tempo polinomial que recebe como entrada um parâmetro de segurança 1^k (k escrito em unário), produz um par de chaves (e, d), onde e é chamado chave pública (a chave de criptografia) e d é a correspondente chave secreta (chave de decriptografia);

Algoritmo de criptografia. E é um algoritmo probabilístico de tempo polinomial que recebe como entrada um parâmetro de segurança 1^k, uma chave pública e (dentro do intervalo de $G(1^k)$), além de uma mensagem $m \in \{0,1\}^k$, e produz um texto cifrado $c \in \{0,1\}^*$ (usamos $c \in E(1^k, e, m)$, para denotar uma possível cifra de m, usando e e k, com o algoritmo E);

Algoritmo de decriptografia. D é um algoritmo probabilístico de tempo polinomial que recebe como entrada um parâmetro de segurança 1^k, uma chave secreta d (dentro do intervalo de $G(1^k)$) e um texto cifrado $c \in E(1^k, e, m)$, e produz uma cadeia $m' \in \{0,1\}^*$, tal que, para todo par (e, d) (dentro do intervalo de $G(1^k)$), para todo m, e para todo $c \in E(1^k, e, m)$, então é ínfima a probabilidade $\text{Prob}[D(1^k, d, c) \neq m']$.

Segurança. O esquema é seguro, de acordo com alguma noção a ser definida na seção 3.6.

Quando o algoritmo de criptografia é probabilístico, mensagens repetidas, para uma mesma chave pública, são de difícil detecção na análise do tráfego.

A condição de que o algoritmo de decriptografia seja probabilístico possibilita maior grau de segurança. Pode-se, porém, relaxar essa condição.

3.4 Teoria da Complexidade e Criptografia

Outro tópico importante a ser comentado, antes de definirmos noções de segurança, é o papel da teoria da complexidade na criptografia.

A criptografia moderna é fundamentada na teoria da complexidade computacional. Quando se diz que um sistema pode ser provado seguro, significa que é possível se provar um limite

mínimo necessário sobre a quantidade de passos computacionais para se quebrar esse sistema (RIVEST, 1990).

Para se aplicar os conceitos da teoria da complexidade computacional, as funções criptográficas são vistas como famílias de funções, parametrizadas por um *parâmetro de segurança* k. Para cada valor de k deve haver um criptossistema específico ou uma família deles. Podemos imaginar que um sistema com um parâmetro de segurança k possui chaves, entradas e saídas, todas escritas na forma de cadeias de comprimento polinomial em k. À medida que o valor de k cresce substancialmente, pode-se esperar que o sistema dificilmente será quebrado, isto é, é inviável a recuperação de qualquer informação útil, num tempo razoável. É equivalente dizer que adversários de complexidade de tempo polinomial em k não obtêm sucesso em suas formas de ataque.

Entretanto, uma vez que não se sabe com certeza se certos tipos de problemas realmente não possuem solução de tempo polinomial, uma "prova"de segurança depende de hipóteses não comprovadas (embora plausíveis) a respeito da dificuldade computacional desses problemas.

Em linhas gerais, o que se demonstra a respeito da segurança de um criptossistema é que a habilidade de um adversário em obter informações sobre o sistema, acaba por levar à contradição em torno da dificuldade em resolver um problema difícil[4]. Normalmente, isso toma a forma de *redução polinomial* de um problema (conhecido e tido por difícil) a outro (quebrar o criptossistema).

A grosso modo, uma *redução* é uma transformação de um problema em outro. Se um problema A é reduzido a um problema B, significa que A não é mais difícil que B (e B não é mais fácil que A). Às vezes uma redução toma a forma de um algoritmo que usa a solução de um problema (como subrotina) para resolver o outro.

Cabe observar, no entanto, que muitos dos problemas intratáveis possuem certas instâncias, sob as quais eles são eficientemente resolvidos. É importante fazer escolhas adequadas de parâmetros para evitar que uma implementação inconveniente resulte num sistema inseguro (MORIN, 1996).

3.4.1 Sobre a Escolha do Problema

A teoria da complexidade agrupa os problemas em classes de complexidade. Resumidamente, a classe P consiste de todos problemas que possuem solução de tempo polinomial; a classe NP consiste dos problemas que podem ser resolvidos por uma máquina de Turing não-determinística. A classe NP inclui a classe P, pois qualquer problema que pode ser resolvido por uma máquina de Turing determinística em tempo polinomial, também tem uma solução não-determinística.

[4]Conforme definição na seção 2.7

Embora muitos problemas em NP não pareçam pertencer a P, até hoje ninguém pôde provar se $P \neq NP$. Alguns problemas são considerados tão difíceis que ganham uma nomenclatura própria, os chamados *NP-difíceis*. Um problema *NP-difícil* não pode ser resolvido em tempo polinomial, a menos que $P = NP$. Um problema *NP-difícil*, que está em NP, é chamado *NP-completo*. Para saber mais sobre esse assunto, veja, por exemplo, (GAREY; JOHNSON, 1979).

Se partirmos do pressuposto que $P \neq NP$, parece natural se projetar criptossistemas de tal forma que o problema em quebrá-lo é um problema *NP-completo*. Entretanto, esta abordagem é difícil, pois os problemas criptográficos normalmente possuem soluções *únicas* enquanto os problemas *NP-completo* podem possuir muitas soluções. Não é fácil reduzir um problema com muitas soluções a um problema de solução única. Ademais, em (LEMPEL, 1979) foi apresentado um curioso sistema criptográfico que é *NP-completo*, porém facilmente quebrável na prática (RIVEST, 1990).

Em (SHAMIR, 1979), são apontadas algumas razões para um problema computacionalmente difícil não implicar necessariamente num criptossistema forte:

- A teoria da complexidade normalmente trata instâncias isoladas de um problema. Um criptanalista, por sua vez, em geral possui grandes coleções de dados estatisticamente relacionados (por exemplo, vários textos cifrados gerados com a mesma chave);
- A complexidade computacional de um problema tipicamente é medida pelo seu comportamento no pior caso ou caso médio. Para ser útil em aplicações da criptografia, o problema deve ser de difícil solução em *praticamente todos os casos*;
- Um problema difícil arbitrário nem sempre pode ser transformado em um criptossistema. É preciso que se possa inserir no problema uma informação adicional com a qual (e somente com ela) seja possível solucioná-lo. Isto é, o problema deve permitir a derivação de uma *função unidirecional com segredo*, conforme exposto na seção 3.1.

Boa parte dos esquemas de criptografia em voga são fundamentados em problemas que mantêm em aberto questões a seu respeito. Por exemplo, o problema da fatoração de inteiros (que ninguém conseguiu provar se está ou não em P) é reduzido polinomialmente a dois subproblemas[5] que levariam à quebra do RSA (RIVEST; SHAMIR; ADLEMAN, 1978).

[5] A saber, cálculo do $\phi(n)$ sem fatorar n, e determinação da chave secreta sem fatorar n e sem calcular $\phi(n)$.

3.5 Problemas Relacionados

Certos problemas considerados difíceis (conforme definição na seção 2.7 e discussão na seção 3.4) são selecionados para amparar uma noção de dificuldade em se quebrar um criptossistema. Aqui, são descritos apenas os problemas que foram adotados nos esquemas estudados (ou que têm implicações diretas neles).

Mais especificamente, como os esquemas de chave pública estudados baseiam-se em emparelhamentos com certas propriedades (ver seção 2.5), vamos descrever os problemas de interesse levando-se em conta:

- $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos cíclicos de ordem prima q;
- $\mathbb{G}_1$ é grupo aditivo;
- $\mathbb{G}_2$ é grupo multiplicativo;
- O mapeamento $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ é um emparelhamento (mapeamento bilinear admissível);
- DL refere-se ao problema do logaritmo discreto (*Discrete-Log*);
- DDH refere-se ao problema de decisão Diffie-Hellman (*Decision Diffie-Hellman*);
- CDH refere-se ao problema Diffie-Hellman computacional, ou simplesmente problema Diffie-Hellman (*Computational Diffie-Hellman*);
- BDH refere-se ao problema Diffie-Hellman bilinear (*Bilinear Diffie-Hellman*);
- BPI refere-se ao problema de inversão do emparelhamento bilinear (*Bilinear Pairing Inversion*);
- GDH refere-se ao problema Diffie-Hellman lacunar (*Gap Diffie-Hellman*);
- q-SDH refere-se ao problema *q-Strong Diffie-Hellman*;
- Um índice $\mathbb{G}_i$ em DL, DDH, CDH ou GDH denota o grupo $\mathbb{G}_1$ ou $\mathbb{G}_2$ sobre o qual o problema está definido.

3.5.1 Definições dos Problemas

Seguem as definições dos problemas de interesse:

1. $\text{DDH}_{\mathbb{G}_1}$ – dados: $P, xP, yP, zP \in \mathbb{G}_1$; decidir: $xyP = zP$?
2. $\text{DDH}_{\mathbb{G}_2}$ – dados: $g, g^x, g^y, g^z \in \mathbb{G}_2$; decidir: $g^{xy} = g^z$?

3. $\text{CDH}_{\mathbb{G}_1}$ – dados: $P, xP, yP \in \mathbb{G}_1$; encontrar: xyP.
4. $\text{CDH}_{\mathbb{G}_2}$ – dados: $g, g^x, g^y \in \mathbb{G}_2$; encontrar: g^{xy}.
5. $\text{GDH}_{\mathbb{G}_1}$ – dados: $P, xP, yP \in \mathbb{G}_1$; encontrar: xyP, com a ajuda de um oráculo de decisão Diffie-Hellman (que dados $P, xP, yP, zP \in \mathbb{G}_1$, decide se $xyP = zP$ ou não).
6. $\text{GDH}_{\mathbb{G}_2}$ – dados: $g, g^x, g^y \in \mathbb{G}_2$; encontrar: g^{xy}, com a ajuda de um oráculo de decisão Diffie-Hellman (que dados $g, g^x, g^y, g^z \in \mathbb{G}_2$, decide se $g^{xy} = g^z$ ou não).
7. $\text{DL}_{\mathbb{G}_1}$ – dados: $P, xP \in \mathbb{G}_1$; encontrar: x.
8. $\text{DL}_{\mathbb{G}_2}$ – dados: $g, g^x \in \mathbb{G}_2$; encontrar: x.
9. BPI – dados: $Q \in \mathbb{G}_1$ e $\hat{e}(P, Q) \in \mathbb{G}_2$; encontrar: P.
10. BDH – dados: $P, xP, yP, zP \in \mathbb{G}_1$; encontrar: $\hat{e}(P, P)^{xyz}$.
11. q-SDH – dados: $P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^q Q$, encontrar: um par $(c, \frac{1}{c+\alpha} P)$.

3.5.2 Problema do Logaritmo Discreto

Aqui, valem comentários especiais ao problema do logaritmo discreto, pois os demais problemas dele dependem. A expressão *logaritmo discreto* provém da notação multiplicativa de $\mathbb{G}_2$, onde o cálculo de x, dados g e g^x, sugere que x seja o logaritmo de g^x na base g.

O método da força bruta para cálculo do logaritmo discreto atinge complexidade exponencial (no tamanho do grupo). Quaisquer outros algoritmos que não explorem características particulares do grupo subjacente também são exponenciais, embora possam alcançar melhor desempenho que pela força bruta. Alguns exemplos de técnicas com complexidade de tempo exponencial são os algoritmos de Shanks, Pohlig-Hellman, Pollard e Oorschot-Wiener (BARRETO, 1999).

Em (SHOUP, 1997) são apresentados limites inferiores sobre a complexidade computacional do logaritmo discreto (e Diffie-Hellman), considerando-se ataques genéricos, que não se valem de qualquer propriedade especial do grupo.

Sabe-se que, quando $\mathbb{G}_2$ é um grupo multiplicativo módulo um primo, existe um algoritmo de complexidade de tempo subexponencial que resolve $\text{DL}_{\mathbb{G}_2}$. Contudo, um tal algoritmo é desconhecido para grupos de pontos de uma curva elíptica (ADLEMAN; MCCURLEY, 1994), segue daí uma das razões do interesse de especialistas em criptografia pelos grupos elípticos.

O algoritmo mais rápido conhecido para se resolver o problema do logaritmo discreto utiliza uma variante da técnica conhecida por cálculo de índices. Esse algoritmo foi criado por (GORDON, 1993) e é chamado *crivo de corpo numérico generalizado*[6].

[6]Do original inglês, *generalized number field sieve*.

Em (SILVERMAN; SUZUKI, 1998) foram apresentadas evidências de que não existe uma extensão viável do método do cálculo de índices para o problema do logaritmo discreto sobre curvas elípticas.

Entretanto, há certos tipos de curvas que devem ser evitadas em aplicações de criptografia. É o caso das curvas supersingulares e curvas anômalas. Em (MENEZES; VANSTONE; OKAMOTO, 1991), o emparelhamento de Weil foi utilizado para demonstrar uma redução de tempo polinomial probabilístico para a classe de curvas elípticas supersingulares. E em (SMART, 1999), é demonstrado que o cálculo do logaritmo para grupos elípticos de curvas anômalas pode ser feito com complexidade de tempo polinomial.

Em 1998, vários pesquisadores descobriram um meio de acelerar o cálculo do logaritmo discreto para curvas elípticas sobre corpos GF(2^m), onde m é um número composto. Porém o ganho não é significativo ao ponto de conferir ameça aos criptossistemas nelas baseados (BARRETO, 1999).

3.5.3 Resultados Conhecidos

As reduções envolvendo os problemas de interesse são esboçadas na figura 3.1. Para interpretração, onde se vê $A \Rightarrow B$ (A implica B) significa que existe uma redução polinomial do problema A para o problema B. Para emparelhamentos assimétricos, leia-se G_T no lugar de $\mathbb{G}_2$.

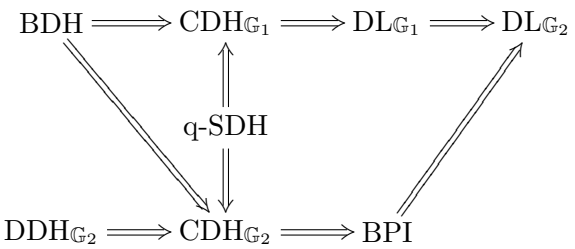

Figura 3.1: Mapa de reduções entre os problemas.

A existência de mapeamento $\hat{e}$ admissível faz com que DDH$_{\mathbb{G}_1}$ seja fácil, pois dados $P, xP, yP, zP \in \mathbb{G}_1$, basta comparar se $\hat{e}(P,P)^{xy} = \hat{e}(P,P)^z$. E pela redução de (MENEZES; VANSTONE; OKAMOTO, 1991), DL$_{\mathbb{G}_1}$ não é mais difícil que DL$_{\mathbb{G}_2}$. Segue que, para o problema do logaritmo discreto ser difícil em $\mathbb{G}_1$, é preciso escolher um parâmetro de segurança tal que o problema do logaritmo discreto seja difícil em $\mathbb{G}_2$ (BONEH; FRANKLIN, 2001).

Em (JOUX; NGUYEN, 2001) foram costruídos grupos de curvas elípticas onde DDH é fácil

e CDH é equivalente a DL. Exemplos como esses permitem que se conjecture que para alguns dos grupos onde DDH é tratável, CDH permaneça intratável. Esses grupos são chamados de grupos lacunares[7], e induziram a definição do problema GDH. Essa idéia é estendida a toda uma classe de problemas: os lacunares (OKAMOTO; POINTCHEVAL, 2001).

Quando se analisa DL em relação a CDH, sabe-se que, sob determinada condição, DL é equivalente a CDH: em (MAURER; WOLF, 1999) é demonstrado que a existência de uma curva elíptica sobre $GF(q)$ onde DL tem solução, faz com que, sob grupos elípticos com ordem q, DL é reduzido a CDH. Sob essas mesmas condições, $\text{CDH}_{\mathbb{G}_2}$ implica $\text{CDH}_{\mathbb{G}_1}$. No entanto, não é claro como se construir uma curva elíptica sobre $GF(q)$ que tenha uma dada ordem.

Certos tipos de mapeamentos eficientes $\varphi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$ levam à solução de $\text{DDH}_{\mathbb{G}_2}$, em tempo polinomial probabilístico. Isso ocorre quando $\varphi(g^x) = f(x)P$, com $f(x) = ax^n + b$, para quaisquer constantes a, b, n (YACOBI, 2002).

Sobre a dificuldade de BDH, ainda há o que se estudar. Dado que BDH não é mais difícil que CDH, faz-se a questão: sob que condições BDH é polinomialmente equivalente a CDH? Uma resposta a essa questão é importante, pois os esquemas aqui estudados dependem da dificuldade de BDH, porém ainda não se sabe, ao certo, quão difícil é BDH.

Em (CHEON; LEE, 2002) é provado que quando o mapeamento $\hat{e}$ é *fracamente inversível*[8], BDH é equivalente tanto a $\text{CDH}_{\mathbb{G}_1}$ quanto a $\text{CDH}_{\mathbb{G}_2}$. Entretanto não é conhecida a existência de condição menos forte que essa. Ainda no mesmo trabalho, foi demonstrada a não existência de um mapeamento *auto-bilinear* $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_1$, quando $\text{CDH}_{\mathbb{G}_1}$ é difícil.

Nos esquemas de criptografia de chave pública aqui abordados, que dependem da hipótese de que BDH seja difícil, reside a confiança de que todos os isomorfismos $f_Q : \mathbb{G}_1 \rightarrow \mathbb{G}_2$ (induzidos pelo mapeamento bilinear $\hat{e}$, para um ponto fixo Q) são todas funções unidirecionais (conforme definição na seção 3.1) (BONEH; FRANKLIN, 2001).

3.5.4 Sobre as Reduções

Com exceção das reduções que envolvem o problema q-SDH, (YACOBI, 2002) fornece um roteiro para as demonstrações das reduções esquematizadas na figura 3.1.

O problema q-SDH foi introduzido por (BONEH; BOYEN, 2004a), para uso em sistemas com emparelhamentos assimétricos. A hipótese de sua dificuldade foi utilizada em uma de nossas propostas. Vamos, a seguir, rapidamente mostrar que q-SDH é reduzido a CDH. Para tanto, primeiramente definimos outros problemas relacionados:

[7] Do original inglês, *gap group*.

[8] $\hat{e}$ é *fracamente inversível* quando é fácil calcular g_1, g_2, dado $h = \hat{e}(g_1, g_2)$, qualquer que seja h.

- InvCDH$_{\mathbb{G}_1}$ – dados: $P, xP \in \mathbb{G}_1$; encontrar: $x^{-1}P$.
- InvCDH$_{\mathbb{G}_2}$ – dados: $g, g^x \in \mathbb{G}_2$; encontrar: $g^{x^{-1}}$.

Em (BAO; DENG; ZHU, 2003) é demonstrada a equivalência de InvCDH e CDH sobre grupos de ordem prima p. Logo, InvCDH $\Rightarrow$ CDH.

Para reduzirmos q-SDH a InvCDH$_{\mathbb{G}_1}$, considere a existência de um oráculo InvCDH$_{\mathbb{G}_1}$, que responde $x^{-1}P$, dados P e xP. Então, dados os geradores P, Q, com $P = \psi(Q)$, e os pontos $\alpha Q, \alpha^2 Q, \ldots, \alpha^q Q$, o par $(c, \frac{1}{\alpha+c}P)$ é obtido com os seguintes passos:

- Escolher $c \in Z_p^*$;
- Calcular cQ e $(\alpha Q + cQ) = (\alpha + c)Q$.
- Submeter $\psi((\alpha + c)Q)$ ao oráculo InvCDH$_{\mathbb{G}_1}$, para obter $\frac{1}{\alpha+c}P$

O par $(c, \frac{1}{\alpha+c}P)$ é resposta para a instância q-SDH dada. Analogamente, obtém-se redução de q-SDH para InvCDH$_{\mathbb{G}_2}$, de modo que:

$$q\text{-SDH} \Rightarrow \text{InvCDH} \Rightarrow \text{CDH}$$

3.6 Noções de Segurança

A partir do trabalho de (GOLDWASSER; MICALI, 1984), com uma formalização de criptografia probabilística, e contribuições posteriores de diversos autores, foi possível se definir noções fortes de segurança. O conjunto desses trabalhos compõe o que chamamos de *modelo padrão* de *segurança que se pode provar*[9].

A expressão *segurança que se pode provar*, que é adotada por muitos autores, pode entretanto induzir a interpretações errôneas. Em matemática, é consenso que algo *provado* fornece garantia de 100% de certeza a respeito do fato demonstrado. Não é o que ocorre no modelo de provas de segurança, onde o que realmente é demonstrado são *reduções* de problemas. Por esse motivo, Bellare sugere o uso da expressão *segurança reducionista*[10], em vez de *segurança que se pode provar* (BELLARE, 1998).

Vamos descrever, a seguir, os principais conceitos relativos ao modelo padrão.

[9] Do original inglês, *provable security*.
[10] Do original inglês, *reductionist security*.

As noções clássicas de segurança em criptografia de chave pública são classificadas segundo os *objetivos* desejados para os esquemas e os *modelos de adversário*. De acordo com (BELLARE et al., 1998), são dois os objetivos e três os modelos de adversários de interesse.

Os *objetivos* desejados para os criptossistemas de chave pública são:

IND – Incapacidade de Distinção[11]: o adversário escolhe duas mensagens e recebe a criptografia y de uma dessas duas mensagens (escolhida ao acaso pelo desafiante). O adversário é incapaz de distinguir (com chance significativamente maior do que 50%) qual das duas mensagens foi escolhida para ser cifrada (detalhes nas seções 3.6.2 e 3.6.3). Incapacidade de distinção é provada ser equivalente à inabilidade do adversário em, dado um texto cifrado y, aprender qualquer informação a respeito do texto legível x (GOLDREICH, 2004). Esse conceito foi desenvolvido por (GOLDWASSER; MICALI, 1984) e é conhecido como *segurança semântica*.

NM – Não-maleabilidade[12]: define a inabilidade do adversário em, dado um texto cifrado y, obter um texto cifrado y', diferente, tal que os correspondentes textos legíveis x e x' sejam significativamente relacionados (por exemplo, $x = x' + 1$).

Os três *modelos de adversário* de interesse são (definições foram descritas na seção 3.2):

CPA [13]– modelo de ataque por texto legível escolhido: o adversário tem uma chave pública e pode fazer criptografias de mensagens à sua escolha;

CCA1 [14]– modelo de ataque por texto ilegível escolhido: o adversário possui uma chave pública e consulta o oráculo de decriptografia, porém somente antes de receber um desafio de texto cifrado;

CCA2 – modelo de ataque adaptativo por texto ilegível escolhido: o adversário possui uma chave pública e consulta o oráculo de decriptografia a qualquer instante, só não pode solicitar a decriptografia do texto cifrado que lhe é dado como desafio.

Os objetivos e os modelos de adversários são combinados aos pares, compondo as seguintes *noções de segurança* em criptografia de chave pública: IND-CPA, IND-CCA1, IND-CCA2, NM-CPA, NM-CCA1 e NM-CCA2. Dessas, a noção de segurança mais forte é IND-CCA2, a incapacidade de distinção em ataques adaptativos por texto ilegível escolhido, que se demonstra ser equivalente a NM-CCA2, a não-maleabilidade em ataques adaptativos por texto cifrado escolhido.

[11] Do original inglês, *indistinguishability*.
[12] Do original inglês, *non-malleability*.
[13] CPA é acrônimo de *Chosen-Plaintext Attack*.
[14] CCA é acrônimo de *Chosen-Ciphertext Attack*.

3.6.1 Relações entre as Noções

Essas noções de segurança mantêm relacionamentos entre si, que foram comprovados por (BELLARE et al., 1998). As demonstrações dessas relações foram feitas com a utilização do modelo de oráculo aleatório (descrito na seção 3.8). No mesmo trabalho, são apresentados ainda outros dois resultados relativos à noção *ciência de texto legível* (PA[15]). PA formaliza a inabilidade do adversário em criar um texto cifrado y sem o conhecimento de seu respectivo legível x.

Na figura 3.2, os resultados conhecidos são esquematizados de uma forma um pouco diferenciada da exibida em (BELLARE et al., 1998), porém equivalente. Para interpretração, onde se vê $A \rightarrow B$ (A implica B) significa que qualquer esquema de criptografia de chave pública que satisfaz a noção A também satisfaz a noção B. E onde há $A \not\rightarrow B$ (A não implica B), significa que existe pelo menos um esquema assimétrico que possui a noção A e que não satisfaz a noção B (FUJISAKI; OKAMOTO, 2000).

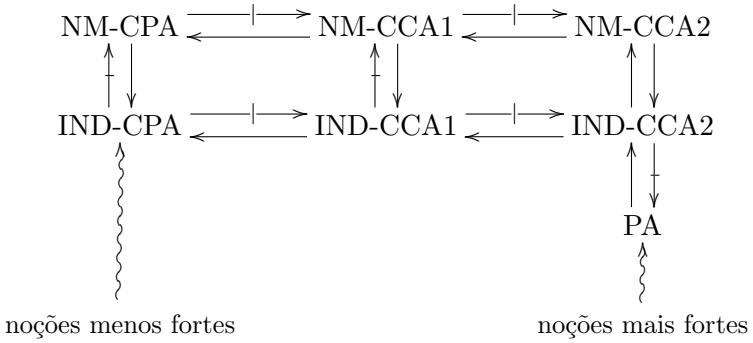

Figura 3.2: Relações entre as noções de segurança.

3.6.2 Jogo IND-CCA2

Para se chegar à definição do que vem a ser um esquema seguro, especifica-se primeiramente uma forma de interação entre o adversário e o criptossistema. Isso se dá estabelecendo-se um jogo, com regras bem definidas, em que nenhuma das partes (nem o adversário, nem seu desafiante) de antemão pode saber se vai ganhar ou não.

Um jogo IND-CCA2 ocorre entre um desafiante e um adversário que imprime ataques adaptativos por texto ilegível escolhido. É composto das seguintes fases:

[15]Do original inglês, *plaintext awareness*.

Inicialização. O desafiante supre o adversário com uma chave pública gerada aleatoriamente.

Consultas. O adversário solicita a decriptografia de textos ilegíveis à sua escolha; o desafiante responde a tais solicitações, entregando os textos legíveis correspondentes.

Desafio. O adversário encerra a fase de consultas e entrega ao desafiante duas mensagens m_0 e m_1 de igual tamanho. O desafiante escolhe ao acaso $b \in \{0, 1\}$ e devolve ao adversário o texto cifrado de m_b.

Novas consultas. O adversário pode solicitar a decriptografia de outros textos ilegíveis, porém não do texto cifrado do desafio.

Palpite. O adversário emite um palpite b' e vence o jogo se $b = b'$.

Um sistema de chave pública é dito seguro no senso IND-CCA2 se nenhum adversário de tempo polinomial, que realiza ataques adaptativos por texto ilegível escolhido, vence o jogo acima com chance significativamente maior do que 50%.

3.6.3 Jogo IND-CPA

Similarmente ao jogo IND-CCA2, um jogo IND-CPA ocorre entre um desafiante e um adversário que imprime ataques por texto legível escolhido. É composto das seguintes fases:

Inicialização. O desafiante supre o adversário com uma chave pública gerada aleatoriamente.

Consultas. O adversário solicita a criptografia de textos legíveis à sua escolha; o desafiante responde a tais solicitações.

Desafio. O adversário encerra a fase de consultas e entrega ao desafiante duas mensagens m_0 e m_1 de igual tamanho. O desafiante escolhe ao acaso $b \in \{0, 1\}$ e devolve ao adversário o texto cifrado de m_b.

Palpite. O adversário emite um palpite b' e vence o jogo se $b = b'$.

Um sistema de chave pública é dito seguro no senso IND-CPA se nenhum adversário de tempo polinomial, que realiza ataques por texto legível escolhido, vence o jogo acima com chance significativamente maior do que 50%.

3.7 Função de Hash

Intuitivamente falando, uma função $H()$ é chamada *função de hash* (ou *função hashing, ou função de espalhamento*), se calcula um valor $y = H(x)$ de comprimento relativamente menor que o comprimento do texto legível x. O valor y é chamado resumo de x.

Dada uma função $H()$, diz-se que há *colisão* se existe um par de legíveis x_1, x_2, com $x_1 \neq x_2$, que acarreta $H(x_1) = H(x_2)$. Deseja-se que $H()$ seja de tal forma que a probabilidade de ocorrerem colisões seja mínima. Sendo assim, o valor $H(x)$ é um representante de x no sentido de que, se $y \approx x$ (isto é, y difere de x em poucos bits), então a desigualdade $H(y) \neq H(x)$ ocorre com alta probabilidade.

Algumas pessoas interpretam $H()$ como uma função de ciframento sem chave, pois com grande probabilidade $H(x)$ é o único representante de x e $H^{-1}(y) = x$ pode ser recuperado por um destinatário que possuir os pares $(x, H(x))$ previamente calculados para *todos* os possíveis x que um remetente possa lhe enviar (TERADA, 2000).

Formalmente, definimos uma *função de hash*, de tamanho n, como sendo uma função $H : \{0,1\}^* \rightarrow \{0,1\}^n$ que satisfaz as propriedades a seguir (MENEZES; VANSTONE; OORSCHOT, 1996):

- Dada uma cadeia binária h de comprimento n é computacionalmente inviável encontrar uma mensagem M tal que $H(M) = h$;
- Dada uma cadeia binária h de comprimento n e uma mensagem M tal que $H(M) = h$, é computacionalmente inviável encontrar outra mensagem M' tal que $H(M') = h$;
- É computacionalmente inviável encontrar duas mensagens M e M' tais que $H(M) = H(M')$, independentemente do valor de $H(M)$.

3.8 Oráculo Aleatório

Uma noção de extensão do conceito de função de hash é obtida com a definição de oráculo aleatório, desenvolvida por (BELLARE; ROGAWAY, 1993). Informalmente falando, o modelo de oráculo aleatório é um mundo em que todas as partes (incluindo os adversários) têm acesso a um oráculo público que gera valores ao acaso. Não há como se distinguir entre a saída de um oráculo aleatório de uma cadeia verdadeiramente aleatória.

Formalmente, um *oráculo aleatório* é um mapeamento $R : \{0,1\}^* \rightarrow \{0,1\}^\infty$, onde cada bit de $R(x)$ é escolhido independentemente e de modo uniforme, para todo x. A notação $\{0,1\}^\infty$ refere-se a cadeias suficientemente longas, geradas pelo oráculo.

Funções de hash populares, como SHA e MD5, não podem ser consideradas aproximações razoáveis de oráculos aleatórios por possuírem estruturas que induzem uma pseudo-aleatoriedade na saída. Há, entretanto, construções que potencialmente simulam bem o comportamento de oráculos aleatórios (BELLARE; ROGAWAY, 1993).

O modelo do oráculo aleatório produz protocolos mais eficientes que os gerados no modelo

padrão de comprovação de segurança. Além disso, permite que sejam estabelecidas relações entre as noções de segurança mais populares para esquemas de criptografia de chave pública, conforme abordado na seção 3.6. Sob o modelo de oráculo aleatório, surgiram trabalhos importantes, como por exemplo as transformações de Fujisaki-Okamoto que possibilitam que esquemas de chave pública, que alcançam certa noção de segurança mais fraca, sejam melhorados com a adição de oráculos aleatórios em seus protocolos (FUJISAKI; OKAMOTO, 1999) e (FUJISAKI; OKAMOTO, 2000).

3.8.1 Discussões

Entretanto, alguns autores questionam o modelo de oráculo aleatório e preferem seguir o modelo padrão de segurança que se pode provar (o modelo padrão é descrito parcialmente na seção 3.6). Provar que um esquema é seguro no modelo do oráculo aleatório não garante sua segurança em implementações reais. Por (CANETTI; GOLDREICH; HALEVI, 1998) foram apresentados exemplos de esquemas criptográficos que se mostram seguros no modelo do oráculo aleatório, porém são inseguros em *qualquer* implementação real.

Apesar dos exemplos de (CANETTI; GOLDREICH; HALEVI, 1998), que foram direcionadamente fabricados para fazer com que o paradigma do oráculo aleatório falhasse, as preocupações induzidas por esses casos não se aplicam a nenhum dos esquemas práticos concretos que foram provados seguros no modelo do oráculo aleatório (FUJISAKI; OKAMOTO, 2000). Comentários semelhantes são feitos em (KOBLITZ; MENEZES, 2004) a outros artigos posteriores que também colocam em dúvida o uso dos oráculos aleatórios.

Bellare afirma que é importante não superestimar nem subestimar o que o paradigma do oráculo aleatório traz em termos de garantias de segurança. Se, por um lado a função R usada no esquema final não é de fato aleatória (o faz com que muitos acreditem que os resultados das demonstrações sob o oráculo são questionáveis), por outro lado, algumas provas sob o modelo padrão também parecem incompletas. Na prática, os ataques sobre esquemas que envolvem teoria dos números e uma função de hash h, tratam h como se fosse aleatória. Isto é, a criptanálise desses esquemas "mistos"normalmente é feita sob a hipótese de que h é aleatória. Mas as provas para esses esquemas, no modelo padrão, mostram que tais ataques falham, a menos que o problema da teoria dos números seja fácil. Em outras palavras, essa análise padrão, no mínimo, provavelmente exclui uma classe comum de ataques – os chamados ataques genéricos (BELLARE, 1998).

Em meio a discussões sobre a validade ou não do modelo do oráculo aleatório, o próprio Bellare, junto com suas alunas, chegou a apresentar um exemplo prático e realístico, que possuía uma propriedade de segurança importante sob o oráculo aleatório, porém não instanciável com nenhuma função de hash concreta (BELLARE; BOLDYREVA; PALACIO, 2003). Afirmaram,

então, que talvez fosse prudente substituir os criptossistemas cuja demonstração de segurança dependia da hipótese do oráculo aleatório, por outros cujo argumento de segurança utilizasse um modelo mais fraco de função de hash. No entanto, foi encontrado um erro na argumentação daquele trabalho. Com esse episódio, Koblitz e Menezes tiraram conclusões favoráveis ao paradigma do oráculo aleatório, conforme exposto a seguir.

A inabilidade dos autores em (BELLARE; BOLDYREVA; PALACIO, 2003) em alcançar uma construção que comprovasse uma restrição ao modelo do oráculo aleatório, sem que houvesse violação de princípios da prática da criptografia, talvez seja uma evidência que sustenta o modelo (KOBLITZ; MENEZES, 2004). No mesmo artigo, é traçada a seguinte comparação.

A intratabilidade de um problema matemático, como o da fatoração de inteiros, é trazida ao mundo das implementações reais nos seguintes termos: se os melhores especialistas em teoria dos números conseguiram fatorar no máximo um módulo RSA com 576 bits, então é praticamente consenso se confiar em módulos de 1024 bits. Em raciocínio análogo, se um dos maiores especialistas no mundo em comprovação de segurança empreende o melhor de seus esforços em minar a validade da hipótese do modelo do oráculo aleatório (que ele próprio concebeu), e se falhou a construção de (BELLARE; BOLDYREVA; PALACIO, 2003), é o melhor que se pôde fazer até o momento, talvez seja a razão para se confiar no paradigma.

Enquanto permanecem abertas questões sobre qual modelo de demonstrações de segurança é mais completo, adequado ou válido, pesquisadores buscam por alternativas. Em (BELLARE; PALACIO, 2004), por exemplo, uma das linhas de pesquisa em noções de segurança alcança resultados sem a utilização de oráculos aleatórios. Em (WATERS, 2004), foi proposto um esquema de criptografia de chave pública baseada em identidades, com nível de segurança equivalente ao de (BONEH; FRANKLIN, 2001), sem a utilização de oráculos aleatórios.

3.9 Resumo

Neste capítulo foram apresentadas definições de criptografia de chave pública, noções de segurança e foram relacionados problemas matemáticos importantes.

Capítulo 4

Criptografia de Chave Pública sem Certificado

Neste capítulo é conceituada a criptografia de chave pública sem certificado. Definem-se esquemas CL-PKE e CL-PKS, respectivamente esquemas de criptografia e de assinatura, no modelo de criptografia de chave pública sem certificado. Modelos de adversários contra esquemas CL-PKE e CL-PKS também são especificados, com o objetivo de se definir noções de segurança para esses esquemas.

4.1 Nomenclatura em CL-PKC

Antes da conceituação do modelo, faz-se necessária uma observação a respeito da nomenclatura utilizada. A sigla CL-PKE (de *Certificateless Public Key Encryption*) é adotada com o significado de esquema (ou protocolo) que especifica procedimentos de criptografia e decriptografia, no modelo de chave pública sem certificado, visando essencialmente o sigilo de mensagens.

CL-PKC (*Certificateless Public Key Cryptography*) denota qualquer criptossistema de chave pública sem certificado, como, por exemplo, CL-PKE, esquema de assinatura (CL-PKS), protocolo de troca de chaves, criptografia autenticada (CL-Auth-PKE), entre outras possíveis derivações de um modelo de criptografia assimétrica.

4.2 Definição de CL-PKE

Um esquema CL-PKE (*Certificateless Public Key Encryption*) é um protocolo de criptografia e decriptografia sob o modelo de criptografia de chave pública sem certificado. Envolve uma entidade confiável, responsável por emitir parcialmente as chaves secretas associadas aos destinatários de mensagens criptografadas. Essa entidade confiável é denotada por KGC (*Key Generating Centre*, centro de geração de chaves). Um CL-PKE consiste de cinco algoritmos, resumidos na tabela 4.1 e descritos na seqüência. Neste texto, a definição de CL-PKE segue (CHENG; COMLEY, 2005) que é ligeiramente mais sintética que a de (AL-RIYAMI; PATERSON, 2003).

Algoritmo	Entrada	Saída
inicializa	parâmetro k	*chave-mestra* e *params*
extrai	*params*, ID_A e a *chave-mestra*	d_A ($SecEsq$)
publica	*params*	t_A ($SecDir$) e N_A (chave pública)
cript	ID_A, N_A e $m \in \mathcal{M}$	$C \in \mathcal{C}$
decript	$C \in \mathcal{C}$, d_A e t_A	$m \in \mathcal{M}$, ou o sinal $\perp$

Tabela 4.1: Algoritmos para um esquema CL-PKE.

Definição 4.1 *Um esquema de criptografia no modelo de criptografia sem certificado, CL-PKE (Certificateless Public Key Encryption) consiste dos seguintes algoritmos de complexidade de tempo polinomial ((AL-RIYAMI; PATERSON, 2003)(CHENG; COMLEY, 2005)):*

inicializa. Este é um algoritmo probabilístico que recebe um parâmetro de segurança k e devolve os parâmetros do sistema params e uma chave-mestra. Os parâmetros do sistema são publicamente conhecidos e incluem descrições do espaço de mensagens $\mathcal{M}$ e do espaço de textos cifrados $\mathcal{C}$. A chave-mestra é um segredo guardado por KGC.

extrai. Este é um algoritmo determinístico que recebe params, chave-mestra e um identificador da entidade A, denotado por $ID_A \in \{0,1\}^*$. Devolve uma chave secreta parcial d_A, denotada por $SecEsq$.

publica. Este é um algoritmo determinístico que recebe params e devolve uma informação secreta t_A, denotada por $SecDir$. Publica a chave pública N_A associada à entidade A e dependente de t_A.

cript. Este é um algoritmo probabilístico que recebe params, um identificador ID_A e a chave pública N_A da entidade A, uma mensagem $m \in \mathcal{M}$. Devolve um texto cifrado $C \in \mathcal{C}$.

decript. Este é um algoritmo determinístico que recebe params, $C \in \mathcal{C}$ e as chaves secretas d_A e t_A ($SecEsq$ e $SecDir$). Devolve uma mensagem $m \in \mathcal{M}$ ou o sinal $\perp$ identificador de falha na decriptografia.

Os algoritmos **extrai** e **publica** podem ser executados em qualquer ordem. O algoritmo **extrai** é executado por KGC, detentor da chave-mestra, enquanto o algoritmo **publica** é executado pela entidade A, que obtém $SecDir$. Além dos segredos chave-mestra e $SecDir$, há um terceiro componente secreto $SecEsq$, compartilhado entre KGC e A (ou seja, o valor $SecEsq$ precisa ser transmitido de forma segura).

Os três valores secretos são relacionados com os valores públicos, de forma a criar um vínculo automático entre cada chave de criptografia e o destinatário correspondente. De fato, a chave secreta parcial $SecEsq$ é calculada como função da chave-mestra de KGC e da identidade ID_A do usuário. A chave secreta parcial $SecDir$, que só a respectiva entidade A conhece, é usada para gerar a chave pública N_A.

Esse modelo dispensa a necessidade de certificado digital, pois a chave de criptografia é composta pela identidade ID_A e pela chave pública N_A. Sem o conhecimento simultâneo de $SecEsq$ e $SecDir$ de A, não há como decriptografar um texto criptografado com ID_A, N_A. $SecEsq$ tem a garantia de KGC, por ser função da chave-mestra; $SecDir$ autentica A, por ser segredo só dela.

Na tabela 4.2, há um resumo das relações entre os valores de chave.

Tipo de Chave	Componentes	Descrição e dependências
Chave de criptografia	ID_A	identidade de A
	N_A	chave pública (que depende de t_A)
Chave de decriptografia	$t_A(SecDir)$	segredo da entidade A
	$d_A(SecEsq)$	é segredo compartilhado entre A e KGC; depende de ID_A e da chave-mestra de KGC

Tabela 4.2: CL-PKE e relação entre chaves.

4.3 Definição de CL-PKS

Um esquema CL-PKS (*Certificateless Public Key Signature*) é um protocolo de assinatura sob o modelo de criptografia de chave pública sem certificado. Envolve uma entidade confiável, responsável por emitir parcialmente as chaves secretas associadas aos usuários que assinam

mensagens. Essa entidade confiável é denotada por KGC (*Key Generating Centre*, centro de geração de chaves). A seguir, a definição formal de CL-PKS.

Definição 4.2 *Um esquema de assinatura no modelo de criptografia sem certificado, CL-PKS (Certificateless Public Key Signature) consiste dos seguintes algoritmos de complexidade de tempo polinomial ((AL-RIYAMI; PATERSON, 2003)(HUANG et al., 2005)):*

inicializa. Este é um algoritmo probabilístico que recebe um parâmetro de segurança 1^k e responde uma lista de parâmetros do sistema chamada params e uma chave-mestra.

gera-parcial. Este é um algoritmo determinístico que recebe uma identidade ID_A, params e chave-mestra, para gerar uma chave secreta parcial D_A.

gera-info-secreta. Este é um algoritmo probabilístico que gera uma informação secreta t_A para um usuário de identidade ID_A.

gera-secreta. Este é um algoritmo determinístico que recebe uma identidade ID_A, uma chave secreta parcial D_A, uma informação secreta t_A e params, para gerar uma chave secreta (D_A, t_A) que permite assinatura de mensagens.

publica. Este é um algoritmo determinístico que recebe uma identidade ID_A, uma informação secreta t_A e params, para gerar uma chave pública N_A para verificação de assinaturas.

assina. Este é um algoritmo probabilístico que recebe uma identidade ID_A, uma chave secreta (D_A, t_A), params e uma mensagem M, para gerar uma assinatura σ.

verifica. Este é um algoritmo determinístico que recebe uma identidade ID_A, a chave pública N_A e params, uma mensagem M e uma assinatura σ. A resposta é um bit b; $b = 1$ significa que a assinatura é aceita, enquanto $b = 0$ significa assinatura inválida.

O algoritmo **gera-parcial** é executado por KGC, detentor da chave-mestra, enquanto os algoritmos **gera-info-secreta**, **gera-secreta** e **publica** são executados pela entidade A, que conhece a chave secreta de assinatura (a chave completa, composta pela secreta parcial e informação secreta). A verificação de uma assinatura para uma determinada mensagem pode ser feita por qualquer usuário que conheça a chave pública daquele que assinou a mensagem.

4.4 Modelo de Segurança para CL-PKE

Para se avaliar a segurança do esquema CL-PKE, faz-se necessário modelar situações em que quaisquer dos três componentes secretos são comprometidos. Como a exposição de chave-mestra imediatamente compromete $SecEsq$ de qualquer entidade, são definidos dois tipos de adversários, um que conhece a chave-mestra e outro que não, conforme exposto a seguir.

São descritos dois tipos de adversários que realizam ataque adaptativo de texto cifrado escolhido. Deseja-se que esses adversários sejam incapazes de aprender qualquer informação a respeito de uma mensagem, se lhes for dado o respectivo texto cifrado. Essa noção de segurança se refere ao que chamamos de IND-CCA2, na seção 3.6.

Resumidamente, o adversário Tipo-I contra CL-PKE não tem o conhecimento da chave-mestra, porém pode substituir valores de chaves pública. Por sua vez, o adversário Tipo-II possui conhecimento da chave-mestra, mas não pode substituir valores de chaves pública. Aqui, a definição dos adversários contra CL-PKE segue o que foi proposto em (AL-RIYAMI; PATERSON, 2005), que é mais forte que a de (CHENG; COMLEY, 2005) (este último, não permite que adversário Tipo-II obtenha chave secreta, ainda que de identidades diferentes daquela a ser desafiada).

4.4.1 Adversário Tipo-I-CCA2

O adversário Tipo-I-CCA2 contra CL-PKE não conhece a chave-mestra e toma parte no Jogo 1[1] contra um desafiante, conforme segue:

Inicializações. O desafiante usa um parâmetro de segurança k e executa o algoritmo **inicializa**. Entrega ao adversário os parâmetros do sistema params e mantém em segredo a chave-mestra.

Fase 1. O adversário realiza consultas $q_1, \ldots, q_n$ dos tipos a seguir:

- Extração de chave secreta parcial ($SecEsq$) sobre ID_i. O desafiante responde executando o algoritmo **extrai** e repassando o resultado d_{ID_i} ao adversário.
- Publicação da chave pública de ID_i. O desafiante responde executando o algoritmo **publica** e repassando N_{ID_i} ao adversário, porém mantém uma lista de pares de chaves $\langle N_{ID_i}, t_{ID_i} \rangle$ já geradas.
- Substituição da chave pública de ID_i por um novo valor N'_{ID_i}. O desafiante responde com atualização da chave.
- Extração da chave secreta $SecDir$ de ID_i. Se a chave pública de ID_i não tiver sido substituída, o desafiante responde devolvendo o valor t_{ID_i} correspondente; caso contrário, aborta o jogo.
- Decriptografia sobre $\langle ID_i, C_i, N_i \rangle$. O desafiante decriptografa o texto cifrado, após obter os valores secretos d_{ID_i} (por meio da execução de **extrai**, se necessário) e de t_{ID_i} (pela busca na lista de pares de chaves, que possui tamanho limitado à quantidade de publicações). Se t_{ID_i} não for encontrado, o desafiante devolve $\perp$.

[1] Trata-se de uma variante do jogo IND-CCA2, definido na seção 3.6.2, para tratar as variáveis e condições adicionais em CL-PKE.

Desafio. Uma vez que o adversário resolve encerrar a Fase 1, entrega ao desafiante duas mensagens $m_0, m_1 \in \mathcal{M}$, de igual tamanho, uma identidade ID_{ch} e uma chave pública N_{ch} sobre as quais se deseja aplicar o desafio. O desafiante escolhe aleatoriamente $b \in \{0,1\}$ e devolve ao adversário o texto cifrado $C^* = \mathbf{cript}(\mathsf{params}, ID_{ch}, m_b, N_{ch})$, sob a condição de que ID_{ch} não tenha sido utilizado em nenhuma consulta de extração de chave secreta $SecEsq$, na Fase 1 (assim, a chave N_{ch} ser substituída, caso $SecDir$ não seja solicitada).

Fase 2. O adversário emite novas consultas $q_{n+1}, \ldots, q_l$ dos tipos a seguir:

- Extração de chave secreta parcial sobre ID_i, com $ID_i \neq ID_{ch}$. O desafiante responde como na Fase 1.
- Publicação da chave pública de ID_i. O desafiante responde como na Fase 1.
- Substituição da chave pública de ID_i por um novo valor N'_{ID_i}. O desafiante responde como na Fase 1.
- Extração da chave secreta $SecDir$ de ID_i. O desafiante responde como na Fase 1.
- Decriptografia sobre $\langle ID_i, C_i, N_i \rangle \neq \langle ID_{ch}, C^*, N_{ch} \rangle$. O desafiante responde como na Fase 1.

Palpite. O adversário gera um palpite $b' \in \{0,1\}$ e vence o jogo se $b' = b$

Em outras palavras, um adversário $\mathcal{A}$ Tipo-I contra CL-PKE não tem o conhecimento da chave-mestra, porém pode substituir valores de chave pública além de extrair chaves secretas $SecEsq$ e $SecDir$, requisitar chaves públicas e solicitar decriptografias, para identidades à sua escolha, sob as seguintes restrições:

1. Em momento algum $\mathcal{A}$ pode extrair a chave secreta $SecEsq$ para ID_{ch}.
2. $\mathcal{A}$ não pode extrair a chave secreta $SecDir$ para identificadores que já tenham tido sua chave pública substituída.
3. Na Fase 2, $\mathcal{A}$ não pode fazer uma consulta de decriptografia sobre o texto cifrado C^* do desafio, para ID_{ch} ou N_{ch}, que foram usados para criptografar m_b.

Um adversário assim definido é denominado Tipo-I-CCA2. Dizemos que a vantagem de um adversário $\mathcal{A}$ Tipo-I-CCA2 contra o esquema $\mathcal{E}$ é função do parâmetro de segurança k, definida por:

$$Vant^{I}_{\mathcal{E},\mathcal{A}}(k) = \mid Pr[b = b'] - 1/2 \mid$$

4.4.2 Adversário Tipo-II-CCA2

O adversário Tipo-II-CCA2 contra CL-PKE conhece a chave-mestra (conseqüentemente possui $SecEsq$ de qualquer entidade) e toma parte no Jogo 2 contra um desafiante, conforme segue:

Inicializações. O desafiante usa um parâmetro de segurança k e executa o algoritmo **inicializa**. Entrega ao adversário os parâmetros do sistema params e a chave-mestra.

O adversário escolhe uma vítima com identidade ID_{ch}.

Fase 1. O adversário realiza consultas $q_1, \ldots, q_n$ dos tipos a seguir:

- Publicação da chave pública de ID_i. Como na Fase 1, do Jogo 1, o desafiante responde executando o algoritmo **publica** e repassando N_{ID_i} ao adversário, porém mantém uma lista de pares de chaves $\langle N_{ID_i}, t_{ID_i} \rangle$ já geradas.
- Extração de chave secreta $SecDir$ para $ID_i \neq ID_{ch}$.
- Decriptografia sobre $\langle ID_{ch}, C_i, N_{ch} \rangle$. O desafiante responde como na consulta de Decriptografia da Fase 1, do Jogo 1.

Desafio. Uma vez que o adversário resolve encerrar a Fase 1, entrega ao desafiante duas mensagens $m_0, m_1 \in \mathcal{M}$, de igual tamanho, sobre as quais se deseja aplicar o desafio. O desafiante escolhe aleatoriamente um bit $b \in \{0, 1\}$ e devolve ao adversário, como desafio, o texto cifrado $C^* = \textbf{cript}(\mathsf{params}, ID_{ch}, m_b, N_{ch})$.

Fase 2. O adversário emite novas consultas $q_{n+1}, \ldots, q_l$ dos tipos a seguir:

- Publicação da chave pública de ID_i. O desafiante responde como na Fase 1.
- Extração de chave secreta $SecDir$ para $ID_i \neq ID_{ch}$.
- Decriptografia sobre $\langle ID_{ch}, C_i, N_{ch} \rangle$, onde $C_i \neq C^*$. O desafiante responde como na Fase 1.

Palpite. O adversário gera um palpite $b' \in \{0, 1\}$ e vence o jogo se $b' = b$.

Em outras palavras, um adversário $\mathcal{A}$ Tipo-II contra CL-PKE possui conhecimento da chave-mestra e, conseqüentemente, pode calcular chave secreta parcial $SecEsq$. Também pode requisitar chaves públicas, chaves secretas e solicitar decriptografias, para identidades à sua escolha, mas se submete às seguintes restrições:

1. Em momento algum $\mathcal{A}$ pode substituir valores de chaves pública.
2. Em momento algum $\mathcal{A}$ pode extrair chave secreta $SecDir$ para ID_{ch}.
3. Na Fase 2, $\mathcal{A}$ não pode fazer uma consulta de decriptografia sobre o texto cifrado C^* do desafio, para ID_{ch} e N_{ch}, que foram usados para criptografar m_b.

Um adversário assim definido é denominado Tipo-II-CCA2. Dizemos que a vantagem de um adversário $\mathcal{A}$ Tipo-II-CCA2 contra o esquema $\mathcal{E}$ é função do parâmetro de segurança k, definida por:

$$Vant^{II}_{\mathcal{E},\mathcal{A}}(k) = \mid Pr[b = b'] - 1/2 \mid$$

4.4.3 Noção de Segurança para CL-PKE

Dadas as descrições dos adversários contra CL-PKE, definimos a noção de segurança a seguir.

Definição 4.3 *Um esquema $\mathcal{E}$ CL-PKE satisfaz a noção de segurança IND-CCA2 se para quaisquer adversários $\mathcal{A}$ Tipo-I-CCA2 e Tipo-II-CCA2, de tempo polinomial em k, são ínfimas as vantagens $Vant^{I}_{\mathcal{E},\mathcal{A}}(k)$ e $Vant^{II}_{\mathcal{E},\mathcal{A}}(k)$.*

4.5 Modelo de Segurança para CL-PKS

Para se avaliar a segurança de um esquema CL-PKS, são definidos dois tipos de adversários, um que conhece a chave-mestra e outro que não, analogamente ao que foi feito para CL-PKE (ver seção 4.4).

Aqui, são descritos dois tipos de adversários que realizam ataque adaptativo de mensagem escolhida. Deseja-se que esses adversários sejam incapazes de aprender qualquer informação que lhes possa ser útil para forjar uma assinatura, isto é, gerar uma assinatura válida sem o conhecimento da chave secreta. Essa noção de segurança se refere ao que chamamos de EUF-CMA (de *Existentially UnForgeable under adaptive Chosen Message Attacks*, ou existencialmente não-forjável contra ataques adaptativos de mensagem escolhida).

Resumidamente, o adversário Tipo-I-CMA contra CL-PKS não tem o conhecimento da chave-mestra, porém pode substituir valores de chaves pública. Por sua vez, o adversário Tipo-II-CMA possui conhecimento da chave-mestra, mas não pode substituir valores de chaves pública. Nos próximos dois tópicos, são apresentadas as definições dos adversários contra CL-PKS. Adotamos as mesmas definições propostas por (ZHANG et al., 2006).

4.5.1 Adversário Tipo-I-CMA

O adversário Tipo-I contra CL-PKS não conhece a chave-mestra e toma parte no Jogo-I, abaixo, contra um desafiante. Considere um adversário Tipo-I $\mathcal{A}^I$ que interage com o desafiante $\mathcal{D}$, da forma como descrita a seguir:

Fase 1-I. O desafiante $\mathcal{D}$ executa **inicializa**(1^k) para gerar a`chave-mestra` e `params`. O desafiante entrega `params` para $\mathcal{A}^I$ e mantém a`chave-mestra` em sigilo.

Fase 2-I. $\mathcal{A}^I$ realiza consultas aos oráculos dos tipos a seguir:

Extração de Chave Parcial: Para um dado ID_A, o desafiante executa **gera-info-secreta** para obter a chave parcial D_A e a entrega para $\mathcal{A}^I$.

Extração de Chave Secreta: Para um dado ID_A, o desafiante obtém primeiramente a chave parcial D_A e executa **gera-info-secreta** para obter a informação secreta t_A. Entrega o par $S_A = (D_A, t_A)$ ao adversário.

Requisição de Chave Pública: Para um dado ID_A, obtém a chave parcial D_A e a informação secreta t_A. Executa **publica** para gerar N_A, que é entregue ao adversário.

Substituição de Chave Pública: Para um dado ID_A, $\mathcal{A}^I$ pode substituir a chave pública N_A por um valor à sua escolha. Não é requerido que $\mathcal{A}^I$ forneça o valor secreto, correspondente à nova chave pública.

Assinatura: Para um dado ID_A e uma mensagem escolhida M, se N_A não tiver sido substituída, o desafiante obtém a chave secreta S_A de uma lista de controle e gera uma assinatura para M, usando S_A. Caso N_A tenha sido substituída, o adversário pode ainda submeter o valor secreto t_A (referente ao novo valor de chave pública) ao oráculo de assinatura e obter uma resposta.

Fase 3-I. $\mathcal{A}^I$ gera uma mensagem M^* e uma assinatura σ^* para a identidade ID^* com N_A^*. ID^* não pode ser uma identidade para a qual a chave secreta tenha sido consultada na Fase 2-I; também não pode ter ocorrido simultaneamente uma substituição de chave pública e extração da chave secreta. Além disso, M^* não pode ter sido usada numa consulta ao oráculo de assinatura junto com ID^* e N_A^*. Entretanto, no caso de N_A^* ser diferente da chave pública original de ID^*, $\mathcal{A}^I$ não pode submeter o correspondente valor secreto da nova chave pública, sem que tenha feito uma consulta de assinatura para ID^* e N_A^*.

Um adversário assim definido é denominado Tipo-I-CMA. Dizemos que um adversário $\mathcal{A}^I$ Tipo-I-CMA contra o esquema $\mathcal{S}$ obtém sucesso se é capaz de gerar uma assinatura válida com probabilidade não ínfima.

4.5.2 Adversário Tipo-II-CMA

O adversário Tipo-II contra CL-PKS conhece a chave-mestra e toma parte no Jogo-II, abaixo, contra um desafiante. Considere um adversário Tipo-II $\mathcal{A}^{II}$ que interage com o desafiante $\mathcal{D}$, da forma como descrita a seguir:

Fase 1-II. O desafiante $\mathcal{D}$ executa **inicializa**(1^k) para gerar a `chave-mestra` e `params`. O desafiante entrega `params` e `chave-mestra` para $\mathcal{A}^{II}$.

Fase 2-II. $\mathcal{A}^{II}$ realiza operações dos tipos a seguir:

Cálculo de Chave Parcial: Para um dado ID_A, o adversário usa `chave-mestra` para calcular a chave parcial D_A.

Consulta ao Oráculo de Extração de Chave Secreta: Para um dado ID_A, o desafiante calcula primeiramente a chave parcial D_A e executa **gera-info-secreta** para obter a informação secreta t_A. Entrega o par $S_A = (D_A, t_A)$ ao adversário.

Consulta ao Oráculo de Requisição de Chave Pública: Para um dado ID_A, calcula a chave parcial D_A e obtém a informação secreta t_A. Executa **publica** para gerar N_A, que é entregue ao adversário.

Consulta ao Oráculo de Assinatura: Para um dado ID_A e uma mensagem escolhida M, o desafiante obtém a chave secreta S_A de uma lista de controle e gera uma assinatura para M, usando S_A.

Fase 3-II. $\mathcal{A}^{II}$ gera uma mensagem M^* e uma assinatura σ^* para a identidade ID^* com N_A^*. ID^* não pode ser uma identidade para a qual a chave secreta tenha sido consultada na Fase 2-I. Além disso, M^* não pode ter sido usada numa consulta ao oráculo de assinatura junto com ID^* e N_A^*.

Um adversário assim definido é denominado Tipo-II-CMA. Dizemos que um adversário $\mathcal{A}^{II}$ Tipo-II-CMA contra o esquema $\mathcal{S}$ obtém sucesso se é capaz de gerar uma assinatura válida com probabilidade não ínfima.

4.5.3 Noção de Segurança para CL-PKS

Dadas as descrições dos adversários contra CL-PKS, definimos a noção de segurança a seguir.

Definição 4.4 *Um esquema $\mathcal{S}$ CL-PKS é dito existencialmente não-forjável contra ataques adaptativos de mensagem escolhida Tipo-I ou Tipo-II, se é ínfima a probabilidade de obtenção de sucesso de qualquer adversário $\mathcal{A}$ Tipo-I-CMA (ou Tipo-II-CMA), de complexidade de tempo polinomial em k, durante interação no Jogo-I (ou Jogo-II). Em outras palavras, dado um parâmetro k,*

$$\forall \epsilon > 0 : \Pr_{\mathcal{A},\mathcal{S},k}\{\text{sucesso}\} \leq \varepsilon$$

4.6 Sobre a Denominação

Cabem comentários a respeito da denominação *certificateless*. Embora a expressão "sem certificado" possa sugerir a absoluta extinção de certificados no sistema, de certa forma, a chave secreta parcial acaba por cumprir o papel de um certificado. Isto é fato, uma vez que a chave

parcial é uma informação que associa univocamente a identidade de um usuário à entidade de confiança do sistema. Tal informação é assinada pela entidade de confiança e somente essa entidade é capaz de fazê-lo. Ou seja, a chave secreta parcial embute em si propriedade de um certificado digital; o que a diferencia realmente de um certificado convencional é sua forma de compartilhamento com os usuários.

A chave parcial é transmitida uma única vez entre KGC e usuário dono da identidade correspondente, e de forma segura (o que pode ser caro de se implementar). Os demais usuários, ao utilizarem uma chave pública, não necessitam buscar provas de que a chave pública realmente está associada com a identidade alvo da operação criptográfica, isto é, os usuários não precisam buscar certificados digitais, publicamente distribuídos, como ocorre em uma ICP tradicional.

Uma denominação mais justa para CL-PKC talvez seria "Criptografia de Chave Pública Auto-certificada", porém esta é uma nomeação de um outro modelo, de (GIRAULT, 1991), que serviu de inspiração para os autores de (AL-RIYAMI; PATERSON, 2003). Também poderíamos adotar algo como "Criptografia com Certificação Implícita", porém, nesta dissertação, optamos por manter o termo original *certificateless* (ou, literalmente, *sem certificado*).

4.7 Resumo

Neste capítulo foram conceituados criptografia de chave pública sem certificado e esquemas CL-PKE e CL-PKS. Também foram definidos modelos de adversários contra esses dois tipos de esquema e respectivas noções de segurança.

Capítulo 5

Proposta de Esquemas sob o Modelo CL-PKC

Neste capítulo, propomos dois novos esquemas CL-PKC, um CL-PKE de criptografia e decriptografia, e um esquema de assinatura CL-PKS, derivados dos trabalhos analisados. As propostas são seguidas de rápidas demonstrações de suas validades e cometários sobre suas origens. Os estudos sobre a segurança dos esquemas propostos são apresentados no capítulo logo a seguir.

5.1 CL-PKE-Proposto

Nesta seção, é apresentada nossa proposta de esquema de criptografia e decriptografia baseada no modelo de criptografia de chave pública sem certificado. O esquema que chamaremos de CL-PKE-Proposto é definido pelos algoritmos abaixo descritos.

inicializa. Dado um parâmetro de segurança 1^k, inteiros n e k_0, com $0 < k_0 < n$ e k_0 polinomial em n:

1. Gerar dois grupos cíclicos $\mathbb{G}_1$ e $\mathbb{G}_2$ de ordem prima $q > 2^k$ e um emparelhamento bilinear $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Escolher aleatoriamente um gerador $P \in \mathbb{G}_1^*$.
2. Escolher aleatoriamente $s \in \mathbb{Z}_q^*$ e calcular $P_{pub} := sP$.
3. Escolher três funções de *hash*
   $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$
   $H_2 : \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \rightarrow \{0,1\}^n$
   $H_3 : \{0,1\}^{n-k_0} \times \{0,1\}^{k_0} \rightarrow \mathbb{Z}_q^*$

4. Definir:
   $\mathcal{M} = \{0,1\}^{n-k_0}$, como espaço de mensagens;
   $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$, como espaço de textos cifrados;
   s, como chave-mestra do sistema;
   $\mathsf{params} := \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, k_0, P, P_{pub}, H_1, H_2, H_3 \rangle$, como parâmetros do sistema.

extrai. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s:

1. Calcular $Q_A := H_1(ID_A)$.
2. Devolver a chave secreta parcial $d_A := sQ_A$.

publica. Dado params, uma entidade A seleciona ao acaso uma informação secreta $t_A \in \mathbb{Z}_q^*$ e calcula sua chave pública $N_A := t_A P$. A guarda t_A em sigilo e publica N_A.

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A, params e a chave pública N_A:

1. Escolher aleatoriamente $\sigma \in \{0,1\}^{k_0}$
2. Calcular
   $r := H_3(m, \sigma)$
   $Q_A := H_1(ID_A)$
   $g^r := \hat{e}(P_{pub}, Q_A)^r$
   $f := rN_A$
3. Devolver o texto cifrado $C = \langle rP, (m \parallel \sigma) \oplus H_2(rP, g^r, f) \rangle$.

decript. Dados $C = \langle U, V \rangle \in \mathcal{C}$ e os valores secretos d_A e t_A:

1. Calcular
   $g' := \hat{e}(U, d_A)$
   $f' := t_A U$
   $(m \parallel \sigma) := V \oplus H_2(U, g', f')$
2. Desmembrar $(m \parallel \sigma)$ e calcular $r := H_3(m, \sigma)$
3. Se $U := rP$, devolver a mensagem m, senão devolver $\perp$.

5.2 CL-PKS-Proposto

Nesta seção, é apresentada uma proposta de esquema de assinatura baseada no modelo de criptografia de chave pública sem certificados. O esquema que chamaremos de CL-PKS-Proposto é definido pelos algoritmos descritos a seguir.

inicializa. Dado um parâmetro de segurança 1^k, o KGC:

1. Gera dois grupos cíclicos G_1, G_2, $G_1 \neq G_2$, e G_T de ordem prima $p > 2^k$ e um emparelhamento bilinear $e : G_1 \times G_2 \rightarrow G_T$.
   Escolhe aleatoriamente dois geradores $P \in G_1^*, Q \in G_2^*$, tais que $P = \psi(Q)$, onde $\psi()$ é um homomorfismo de G_2^* em G_1^*, eficientemente computável (isto é, de complexidade de tempo polinomial em k).
2. Calcula $g := e(P, Q) \in G_T$.
3. Escolhe aleatoriamente a chave-mestra $s \in Z_p^*$ e calcula $Q_{pub} := sQ$.
4. Escolhe funções de *hash*
   $H_1 : \{0,1\}^* \rightarrow Z_p^*$
   $H_2 : \{0,1\}^* \times \{0,1\}^* \times G_T \times G_T \rightarrow Z_p^*$.
5. Define:
   $\mathcal{M} = \{0,1\}^*$, como espaço de mensagens;
   $\mathcal{S} = G_1 \times Z_p^*$, como espaço de assinatura;
   s, como `chave-mestra` do sistema;
   `params` $:= \langle p, G_1, G_2, G_T, e, \psi, P, Q, Q_{pub}, g, H_1, H_2 \rangle$, como parâmetros do sistema.

gera-parcial. Dada uma identidade $ID_A \in \{0,1\}^*$, `params` e a `chave-mestra` s, o KGC:

1. Calcula $D_A := \frac{1}{H_1(ID_A)+s} P \in G_1^*$.
2. Entrega a chave secreta parcial D_A para a entidade cuja identidade é ID_A, por meio de um canal seguro.

gera-info-secreta. Dado `params`, a entidade A escolhe aletoriamente uma informação secreta $t_A \in Z_p^*$.

gera-secreta. Dados ID_A, D_A, t_A e `params`, a entidade A define e mantém em sigilo sua chave secreta de assinatura, formada pelo par $(D_A, t_A) \in G_1^* \times Z_p^*$.

publica. Dados ID_A, t_A e `params`, a entidade A calcula sua chave pública $N_A := g^{t_A} \in G_T$. A publica N_A.

assina. Dados `params`, uma mensagem $M \in \mathcal{M} = \{0,1\}^*$, uma identidade ID_A, a chave pública $N_A = g^{t_A}$ e a chave secreta de assinatura de A, formada pelo par (D_A, t_A), A assina M da seguinte forma:

1. Escolhe aleatoriamente $x \in Z_p^*$.
2. Calcula
$$\begin{cases} r := g^x \in G_T \\ h := H_2(M, ID_A, N_A, r) \in Z_p^* \\ S := (x + ht_A)D_A \in G_1 \end{cases}$$

3. A assinatura sobre M é $\sigma := (S, h) \in G_1 \times Z_p^*$.

verifica. Dados `params`, uma mensagem M, a assinatura $\sigma = (S, h)$, a identidade ID_A e a chave pública N_A, o algoritmo gera 1, aceitando σ como autêntica, se e somente se:

$$h = H_2(M, ID_A, N_A, r')$$
onde
$$r' := e[S, H_1(ID_A)Q + Q_{pub}](N_A)^{-h}$$

caso contrário, responde 0.

5.3 Validade dos Esquemas Propostos

Para se verificar a validade dos esquemas dados, basta lembrar que os emparelhamentos $\{\hat{e}, e\}$ são bilineares. Seguem as equações de validação.

5.3.1 Validade de CL-PKE-Proposto

$$\begin{aligned} g' &= \hat{e}(U, d_A) \\ &= \hat{e}(rP, sQ_A) \\ &= \hat{e}(P, Q_A)^{rs} \\ &= \hat{e}(sP, Q_A)^r \\ &= \hat{e}(P_{pub}, Q_A)^r \\ &= g^r \\ f' &= t_A U = t_A rP = rN_A = f \end{aligned}$$

Logo $H_2(U, g', f') = H_2(rP, g^r, f)$. Como $V = (m \parallel \sigma) \oplus H_2(rP, g^r, f)$, então

$$V \oplus H_2(rP, g^r, f) = (m \parallel \sigma)$$

Portanto **decript** recupera corretamente a mensagem cifrada por **cript**.

5.3.2 Validade de CL-PKS-Proposto

$$\begin{aligned}
r' &= e\{S, H_1(ID_A)Q + Q_{pub}\}(N_A)^{-h} \\
&= e\{[(x+ht_A)D_A], H_1(ID_A)Q + Q_{pub}\}(N_A)^{-h} \\
&= e\{[(x+ht_A)\tfrac{1}{H_1(ID_A)+s}P], H_1(ID_A)Q + sQ\}(N_A)^{-h} \\
&= e\{[(x+ht_A)\tfrac{1}{H_1(ID_A)+s}P], [H_1(ID_A)+s]Q\}(N_A)^{-h} \\
&= e\{[(x+ht_A)P], Q\}^{\frac{1}{H_1(ID_A)+s}(H_1(ID_A)+s)}(N_A)^{-h} \\
&= e\{[(x+ht_A)P], Q\}(N_A)^{-h} \\
&= e\{P, Q\}^{(x+ht_A)}(g^{t_A})^{-h} \\
&= g^{(x+ht_A)}g^{-ht_A} \\
&= g^x \\
&= r
\end{aligned}$$

Logo,

$$h = H_2(M, ID_A, N_A, r) = H_2(M, ID_A, N_A, r')$$

Portanto **verifica** responde corretamente se é válida ou não uma assinatura gerada por **assina**.

5.4 Origens dos Esquemas Propostos

De forma resumida, o esquema CL-PKE-Proposto pode ser descrito como uma reunião de construções apresentadas em vários trabalhos. A utilização de rP em H_2 foi sugerida por (CRAMER; SHOUP, 2004). O uso de rN_A e a formulação completa de H_2 foi encontrada por (CHENG; COMLEY, 2005), ao tentar otimizar os esquemas precedentes de CL-PKC de (AL-RIYAMI; PATERSON, 2005) e (AL-RIYAMI; PATERSON, 2003).

A escolha dos espaços de mensagem e de texto cifrado, além de H_3, foi inspirada no estudo de (GALINDO, 2005), que por sua vez adotou a transformação de (FUJISAKI; OKAMOTO, 2000) para fortalecimento de esquemas de criptografia de chave pública, com uso de menor quantidade de funções de *hash*. Galindo apresentou uma melhoria sobre o trabalho de (BONEH; FRANKLIN, 2001), que, por sua vez, permitiu a concretização da noção de criptografia de chave pública sem certificado.

Já o esquema de assinatura proposto pode ser descrito como a adaptação do esquema de assinatura baseado em identidades (IBS) de (BARRETO et al., 2005), para o modelo CL-PKS.

O esquema IBS de (BARRETO et al., 2005), por sua vez, é uma evolução da proposta de (SAKAI; KASAHARA, 2003), que sugeriu equações eficientes para verificações baseadas em emparelhamentos bilineares. Como resultado, foi possível alcançar esquemas cujo algoritmo de

assinatura não requer cálculo de emparelhamento; apenas o algoritmo de verificação envolve esse tipo de operação.

Anteriormente ao trabalho de (BARRETO et al., 2005), tinha-se notícia de apenas dois esquemas de assinatura CL-PKS, um já apontado em (AL-RIYAMI; PATERSON, 2003), e outro de (HUANG et al., 2005). Este último detectou que a proposta de (AL-RIYAMI; PATERSON, 2003) era forjável e apresentou novo CL-PKS, como sugestão de correção.

Paralelamente ao desenvolvimento desta dissertação, um trabalho de (ZHANG et al., 2006) foi apresentado com uma proposta de assinatura CL-PKS que também dispensa cálculos de emparelhamento na assinatura, porém requer vários emparelhamentos na verificação.

Todos esses trabalhos relacionados são descritos no apêndice A.

5.5 Resumo

Neste capítulo foram apresentados dois novos esquemas: CL-PKE-Proposto e CL-PKS-Proposto, ambos demonstrados válidos.

Capítulo 6

Análise de Segurança dos Esquemas Propostos

O objetivo deste capítulo é a demonstração do fato de que CL-PKE-Proposto e CL-PKS-Proposto são seguros no modelo de oráculos aleatórios, considerando-se premissas e noções de segurança fixadas.

6.1 Segurança de CL-PKE-Proposto

Nesta seção é apresentada a demonstração do fato de que CL-PKE-Proposto é seguro contra adversários Tipo-I-CCA2 e Tipo-II-CCA2, descritos na seção 4.4. Será utilizado o modelo de oráculos aleatórios e a mesma estratégia de (BONEH; FRANKLIN, 2001), que se vale de esquemas auxiliares de criptografia de chave pública, com propriedades específicas. Portanto, um passo a ser tomado consiste na definição de dois esquemas de criptografia de chave pública que chamaremos de Basico-Proposto e Basico-Hib-Proposto.

Para adversários Tipo-I e Tipo-II, será elaborada uma seqüência de reduções que envolvem Basico-Proposto e Basico-Hib-Prop. A seqüência de reduções demonstra que a existência de um adversário com vantagem não-ínfima contra CL-PKE-Proposto implica a existência de algoritmo com complexidade de tempo polinomial que resolve o problema BDH (Diffie-Hellman Bilinear).

6.1.1 Esquemas Auxiliares

Seguem definições dos esquemas de criptografia e decriptografia no modelo convencional de criptografia de chave pública, que chamamos de Basico-Proposto e Basico-Hib-Proposto.

Basico-Proposto é definido pelos três algoritmos:

gerachavesb. Dado um parâmetro de segurança k:

1. Gerar dois grupos cíclicos $\mathbb{G}_1$ e $\mathbb{G}_2$ de ordem prima q e um emparelhamento bilinear $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Escolher aleatoriamente um gerador $P \in \mathbb{G}_1^*$.
2. Escolher aleatoriamente $s \in \mathbb{Z}_q^*$ e calcular $P_{pub} = sP$.
3. Escolher aleatoriamente $Q_A \in \mathbb{G}_1^*$ e $t_A \in \mathbb{Z}_q^*$ e calcular $N_A = t_A P$.
4. Escolher uma função de *hash*
   $H_2 : \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \to \{0,1\}^n$
   para inteiro $n > 0$.

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$.

A chave pública é $K_{pub} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_2, Q_A, N_A \rangle = \langle \mathsf{paramsb}, Q_A, N_A \rangle$

A chave secreta é $K_{sec} = \langle d_A = sQ_A, t_A \rangle$.

criptb. Dados um texto $m \in \mathcal{M}$ e a chave pública K_{pub}:

1. Escolher aleatoriamente $r \in \mathbb{Z}_q^*$
2. Calcular $g^r = \hat{e}(P_{pub}, Q_A)$
3. Devolver o texto cifrado $C = \langle rP, m \oplus H_2(rP, g^r, rN_A) \rangle$.

decriptb. Dados $C = \langle U, V \rangle \in \mathcal{C}$, $\mathsf{paramsb}$ e a chave secreta K_{sec}:

1. Calcular $g' = \hat{e}(U, d_A)$
2. Calcular e devolver $V \oplus H_2(U, g', t_A U) = m$.

Basico-Hib-Proposto é definido pelos três algoritmos:

gerachavesh. Dado um parâmetro de segurança k:

1. Gerar dois grupos cíclicos $\mathbb{G}_1$ e $\mathbb{G}_2$ de ordem prima q e um emparelhamento bilinear $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Escolher aleatoriamente um gerador $P \in \mathbb{G}_1^*$.
2. Escolher aleatoriamente $s \in \mathbb{Z}_q^*$ e calcular $P_{pub} = sP$.
3. Escolher aleatoriamente $Q_A \in \mathbb{G}_1^*$ e $t_A \in \mathbb{Z}_q^*$; calcular $N_A = t_A P$ e $d_A = sQ_A$.
4. Escolher duas funções de *hash*
   $H_2 : \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \to \{0,1\}^n$
   $H_3 : \{0,1\}^{n-k_0} \times \{0,1\}^{k_0} \to \mathbb{Z}_q^*$
   para inteiros n e k_0, $0 < k_0 < n$, com k_0 polinomial em n.

O espaço de mensagens é $\mathcal{M} = \{0,1\}^{n-k_0}$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$.

A chave pública é $K_{pub} =$

$\langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, k_0, P, P_{pub}, H_2, H_3, Q_A, N_A\rangle = \langle \mathsf{paramsh}, Q_A, N_A\rangle$

A chave secreta é $K_{sec} = \langle d_A = sQ_A, t_A\rangle$.

cripth. Dados um texto $m \in \mathcal{M}$ e a chave pública K_{pub}:

1. Escolher aleatoriamente $\sigma \in \{0,1\}^{k_0}$
2. Calcular
   $r = H_3(m, \sigma)$
   $g^r = \hat{e}(P_{pub}, Q_A)^r$
3. Devolver o texto cifrado $C = \langle rP, (m \parallel \sigma) \oplus H_2(rP, g^r, rN_A)\rangle$.

decripth. Dados $C = \langle U, V\rangle \in \mathcal{C}$, $\mathsf{paramsh}$ e a chave secreta K_{sec} e um valor $t_A \in \mathbb{Z}_q^*$:

1. Calcular $g' = \hat{e}(U, d_A)$
2. Calcular $V \oplus H_2(U, g', t_A U) = (m \parallel \sigma)$.
3. Desmembrar $(m \parallel \sigma)$ e calcular $r = H_3(m, \sigma)$
4. se $U = rP$, devolver a mensagem m, senão devolver $\perp$.

6.1.2 Adversários CCA2 e Reduções

Com o auxílio dos esquemas Basico-Proposto e Basico-Hib-Proposto, será traçada, nesta seção, a demonstração de que CL-PKE-Proposto é seguro contra adversários Tipo-I-CCA2 e Tipo-II-CCA2. Na figura 6.1 está esquematizada a seqüência de reduções, que mostra a redução de BDH ao problema de encontrar algoritmo de tempo polinomial (em k) contra CL-PKE-Proposto. Para interpretração, onde se vê $A \rightarrow B$ (A implica B) significa que a existência de um algoritmo A de complexidade de tempo polinomial leva à existência de um algoritmo B, também polinomial.

Lema 6.1 *Suponha que H_1 é um oráculo aleatório. Seja $\mathcal{A}_1$ um adversário Tipo-I-CCA2 contra o esquema CL-PKE-Proposto, com vantagem ϵ_1, tempo de execução t_1, que realiza no máximo q_1 consultas a H_1, q_e consultas de extrações de chave e q_d consultas de decriptografia. Então existe um adversário $\mathcal{A}_2$ CCA2 contra Basico-Hib-Proposto com vantagem $\epsilon_2 \geq \epsilon_1/q_1$ e tempo de execução $t_2 \leq t_1 + c_{\mathbb{G}_1}(q_1 + q_d + q_e)$, onde $c_{\mathbb{G}_1}$ denota o tempo de multiplicação de um ponto em $\mathbb{G}_1$ por um escalar.*

***Demonstração*:** Vamos construir um adversário $\mathcal{A}_2$ Tipo-I-CCA2 que utiliza $\mathcal{A}_1$ para obter vantagem contra Basico-Hib-Proposto. Essa demonstração é muito semelhante à do Lema 1 de (CHENG; COMLEY, 2005).

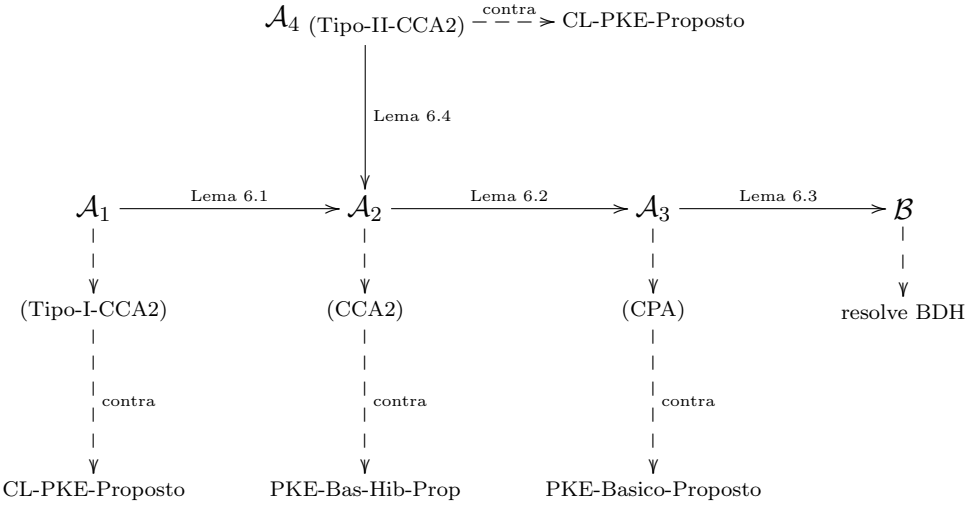

Figura 6.1: Seqüência de Reduções de BDH para Adversários Tipo-I-CCA2 e Tipo-II-CCA2 contra CL-PKE-Proposto.

O jogo entre o desafiante $\mathcal{D}$ e o adversário $\mathcal{A}_2$ se inicia com o desafiante gerando os parâmetros públicos, com a execução do algoritmo **gerachavesh**, de Basico-Hib-Proposto. Como resultado, são obtidas a chave pública $K_{pub} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, k_0, P, P_{pub}, s, H_2, H_3, Q_A, N_A\rangle$ $= \langle \mathsf{paramsh}, Q_A, N_A\rangle$ e a chave secreta $K_{sec} = \langle d_A = sQ_A, t_A\rangle$. O desafiante entrega K_{pub} para $\mathcal{A}_2$ e guarda K_{sec}.

O adversário $\mathcal{A}_2$ monta um ataque CCA2 contra Basico-Hib-Proposto utilizando K_{pub} e $\mathcal{A}_1$ da forma descrita a seguir.

$\mathcal{A}_2$ escolhe ao acaso um índice I, $1 \leq I \leq q_1$, que será associado a uma identidade sobre a qual $\mathcal{A}_2$ tentará aplicar o desafio. $\mathcal{A}_2$ simula o algoritmo **inicializa** de CL-PKE-Proposto, fornecendo para $\mathcal{A}_1$

$\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, k_0, P, P_{pub}, s, H_1, H_2, H_3\rangle (= \langle \mathsf{paramsh}, s, H_1\rangle)$

onde H_1 é um oráculo aleatório controlado por $\mathcal{A}_2$.

O adversário $\mathcal{A}_1$ pode realizar consultas a H_1 em qualquer instante. Essas consultas são manipuladas pelo algoritmo **consultasH$_1$**, conforme segue.

consultasH$_1$. Quando $\mathcal{A}_1$ consulta H_1 para ID_i, $\mathcal{A}_2$ responde sob auxílio de uma lista de triplas $\langle ID_j, Q_j, h_j\rangle$. A lista, denotada por H_1^{lista}, inicialmente encontra-se vazia. $\mathcal{A}_2$ pode fornecer uma das três respostas:

1. Se ID_i já aparece na H_1^{lista}, numa tripla $\langle ID_i, Q_i, h_i \rangle$, então $\mathcal{A}_2$ responde com $Q_i \in \mathbb{G}_1^*$.
2. Caso contrário, se a consulta é sobre o I-ésimo identificador distinto, então $\mathcal{A}_2$ armazena $\langle ID_I, Q_A, \perp \rangle$ na lista e responde $H_1(ID_I) = Q_A$.
3. Caso contrário, $\mathcal{A}_2$ seleciona ao acaso um inteiro $h_i \in \mathbb{Z}_q^*$, calcula $Q_i = h_i P \in \mathbb{G}_1^*$, armazena $\langle ID_i, Q_i, h_i \rangle$ na lista e responde Q_i.

O jogo entre o desafiante $\mathcal{D}$ e o adversário $\mathcal{A}_2$ prossegue nas fases a seguir.

Fase 1. O adversário $\mathcal{A}_1$ lança a Fase 1 de seu ataque ao fazer uma série de consultas (de extração de chave secreta parcial *SecEsq*, publicação de chave pública, substituição, extração de *SecDir* ou de decriptografia). Podemos considerar que $\mathcal{A}_1$ sempre fará uma consulta a H_1, para a mesma identidade sobre a qual tentará realizar as próximas consultas. $\mathcal{A}_2$ simula o desafiante de $\mathcal{A}_1$ e responde a essas consultas da seguinte forma:

- Extração de *SecEsq* sobre ID_i. Se $ID_i = ID_I$, então $\mathcal{A}_2$ aborta o jogo (**evento 1**); caso contrário responde com $h_i sP$, onde h_i provém da tripla correspondente a ID_i da H_1^{lista} e $sP = P_{pub}$.
- Publicação para ID_i. Para responder a esta consulta, $\mathcal{A}_2$ mantém outra lista, K^{lista}, com quáduplas na forma $\langle ID_i, t_i, t_i P, R_i \rangle$, indexada por ID_i. Para um novo ID_i, $\mathcal{A}_2$ seleciona aleatoriamente um inteiro $t_i \in \mathbb{Z}_q^*$ e insere a quádupla $\langle ID_i, t_i, t_i P, t_i P \rangle$ na lista, senão $\mathcal{A}_2$ responde com R_i.
- Substituição sobre ID_i por N_i. $\mathcal{A}_2$ substitui R_i por N_i, na quádupla indexada por ID_i na K^{lista}.
- Extração de *SecDir* para ID_i). $\mathcal{A}_2$ verifica se $R_i = t_i P$ na quádupla indexada por ID_i na K^{lista}. Na igualdade, $\mathcal{A}_2$ devolve t_i, senão aborta o jogo (**evento 2**).
- Decriptografia sobre $\langle ID_i, C_i, N_i \rangle$. $\mathcal{A}_2$ procura em K^{lista} por uma quádupla em que $N_i = t_i P$. Se tal quádupla não existir, então $\mathcal{A}_2$ aborta o jogo (**evento 3**). Se a requisição de decriptografia foi para $C_i = \langle U, V \rangle$ sobre ID_I (ou seja, $ID_i = ID_I$), então $\mathcal{A}_2$ faz uma consulta de decriptografia sobre C_i e t_i, e repassa a resposta de $\mathcal{D}$ para $\mathcal{A}_1$. Em caso contrário, $\mathcal{A}_2$ tenta decriptografar consultando H_2 (que é controlado por $\mathcal{D}$) para obter $H_2(U, g^r, t_i U)$, onde $g^r = \hat{e}(U, h_i sP)$ e $h_i sP$ é obtido por meio da extração de *SecEsq*. Ao desmembrar $V \oplus H_2(U, g^r, t_i U)$, $\mathcal{A}_2$ obtém um resultado que é repassado para $\mathcal{A}_1$.

Desafio. Em um dado momento, $\mathcal{A}_1$ encerra a Fase 1, escolhe ID_{ch}, N_{ch} e duas mensagens m_0, m_1 sobre as quais deseja desafiar. Se $ID_{ch} \neq ID_I$, então $\mathcal{A}_2$ aborta o jogo (**evento 4**); caso contrário, m_0, m_1 e N_{ch} são repassados para $\mathcal{D}$, que responde com o texto cifrado $C_{ch} = \langle U', V' \rangle$. $\mathcal{A}_2$ repassa para $\mathcal{A}_1$ o texto cifrado C_{ch}.

Fase 2. $\mathcal{A}_2$ continua a responder às requisições do mesmo modo que na Fase 1, porém aborta o jogo se for realizada alguma consulta de decriptografia sobre $\langle ID_I, C_{ch}, N_{ch}\rangle$ (**evento 5**).

Palpite. Se $\mathcal{A}_1$ emite um palpite b', $\mathcal{A}_2$ devolve o mesmo palpite b'.

Afirmativa: Se o algoritmo $\mathcal{A}_2$ não abortar durante a simulação, então o algoritmo $\mathcal{A}_1$ tem a mesma visão caso estivesse num ataque real.

Demonstração: As respostas de $\mathcal{A}_2$ às consultas a H_1 são independentes e uniformemente distribuídas em $\mathbb{G}_1^*$, pois H_1 é oráculo aleatório. Então $\mathcal{A}_1$ enxerga H_1, controlado por $\mathcal{A}_2$, da mesma forma como se estivesse aplicando um ataque real. Se $\mathcal{A}_2$ não abortar, todas as respostas às consultas de $\mathcal{A}_1$ são válidas (pertencentes aos domínios corretos). Em particular, $\mathcal{A}_2$ responde corretamente a consultas de decriptografia para $\mathcal{A}_1$:

1. Se $ID_i = ID_I$, $\mathcal{A}_2$ consulta $\mathcal{D}$ para decriptografar $\langle U, V\rangle$, com t_i. $\mathcal{D}$ usa $K_{sec} = d_A = sQ_A$ que está em seu poder. Como $H_1(ID_I) = Q_A$ e $N_i = t_iP$, $\mathcal{D}$ obtém a mesma resposta que o desafiante de $\mathcal{A}_1$ daria. Também por esses mesmos motivos, se $\mathcal{A}_2$ não abortar, o texto cifrado do desafio $\langle U', V'\rangle$ é uma criptografia válida para m_b em CL-PKE, e pode ser repassado para $\mathcal{A}_1$.
2. Caso contrário, $\mathcal{A}_2$ calcula $g^r = \hat{e}(U, h_isP)$ e t_iU, para então obter $H_2(U, g^r, t_iU)$. Mas $\hat{e}(U, h_isP) = \hat{e}(rP, sQ_i) = \hat{e}(P_{pub}, Q_i)^r = g^r$ e $t_iU = t_irP = rN_i$. Novamente $\mathcal{A}_2$ dá uma resposta válida para $\mathcal{A}_1$. □

Basta, então, calcularmos a probabilidade de $\mathcal{A}_2$ não abortar durante a simulação. São cinco os eventos que causam aborto:

- Evento 1, denotado por $\mathcal{H}_1$: $\mathcal{A}_1$ solicitou $SecEsq$ para ID_I;
- Evento 2, denotado por $\mathcal{H}_2$: $\mathcal{A}_1$ substituiu uma chave pública de determinada identidade e posteriormente solicitou sua $SecDir$;
- Evento 3, denotado por $\mathcal{H}_3$: $\mathcal{A}_1$ solicitou decriptografia sob uma chave pública N_i desconhecida;
- Evento 4, denotado por $\mathcal{H}_4$: $\mathcal{A}_1$ não escolheu ID_I como ID_{ch};
- Evento 5, denotado por $\mathcal{H}_5$: $\mathcal{A}_2$ solicitou decriptografia para C_{ch} na Fase 2;

Tentativa de extração de $SecDir$ para entidade que teve sua chave pública substituída não é permitida a $\mathcal{A}_1$. Como $\mathcal{A}_2$ apenas simula as requisições de $\mathcal{A}_1$, não ocorre $\mathcal{H}_2$ sem que $\mathcal{A}_1$ tenha abortado.

Uma solicitação de decriptografia com chave pública desconhecida não produz resultado (pois $\mathcal{A}_1$ receberia $\perp$ como resposta) e é possível para $\mathcal{A}_1$ evitar essa situação em sua implementação. Assim consideramos que $\mathcal{A}_2$ não recai em $\mathcal{H}_3$.

Também $\mathcal{H}_5$ só ocorre quando, no desafio, $\mathcal{A}_1$ escolheu $ID_I = ID_{ch}$, porém $\mathcal{A}_1$ seria abortado ao solicitar decriptografia sobre $\langle ID_{ch}, C_{ch}, N_{ch}\rangle$, da mesma forma que $\mathcal{A}_2$.

Além disso, não ter ocorrido $\mathcal{H}_4$ na fase do desafio (isto é, $ID_I = ID_{ch}$) implica que $\mathcal{H}_1$ não ocorreu (pois, se tivesse ocorrido, $\mathcal{A}_1$ teria pedido $SecEsq$ de ID_{ch} e seria abortado antes do desafio).

Logo, tem-se

$$\begin{aligned}\Pr[\mathcal{A}_2 \text{ não abortar}] &= \Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_2 \wedge \neg\mathcal{H}_3 \wedge \neg\mathcal{H}_4 \wedge \neg\mathcal{H}_5] \\ &= \Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_4 \wedge \neg\mathcal{H}_5] \\ &= \Pr[\neg\mathcal{H}_4]\end{aligned}$$

A escolha de I por $\mathcal{A}_2$ é independente da escolha de ID_I por $\mathcal{A}_1$, logo,

$$\Pr[\mathcal{A}_2 \text{ não abortar}] = 1/q_1$$

$\mathcal{A}_2$ usa o palpite de $\mathcal{A}_1$, cuja definição diz que $|\Pr[b = b'] - 1/2\,| \geq \epsilon_1$.

Se $\mathcal{A}_2$ não abortar e tiver sucesso no palpite, ganha o jogo. Combinando-se esses elementos, chega-se à vantagem:

$$\epsilon_2 \geq \epsilon_1/q_1$$

Para análise da complexidade de tempo de $\mathcal{A}_2$, observar os seguintes pontos:

- Como $\mathcal{A}_2$ basicamente simula o desafiante de $\mathcal{A}_1$, o tempo de execução de $\mathcal{A}_1$, denotado por t_1, é o principal componente.
- Na simulação, cada consulta de extração de $SecEsq$, decriptografia e consulta a H_1 envolve uma multiplicação escalar em $\mathbb{G}_1$, que $\mathcal{A}_1$ não realizaria num ataque real. Levando-se em consideração que $c_{\mathbb{G}_1}$ denota o tempo de multiplicação de um ponto em $\mathbb{G}_1$ por um escalar (aleatórios), então $c_{\mathbb{G}_1}(q_1 + q_d + q_e)$ é outro componente no cálculo de t_2.

Então $t_2 \leq t_1 + c_{\mathbb{G}_1}(q_1 + q_d + q_e)$. □

Lema 6.2 *Suponha que H_3 é um oráculo aleatório. Seja $\mathcal{A}_2$ um adversário CCA2 contra o esquema Basico-Hib-Proposto, com vantagem ϵ_2, tempo de execução t_2, que realiza no máximo*

q_3 consultas a H_3 e q_d consultas de decriptografia. Então existe um adversário $\mathcal{A}_3$ CPA contra Basico-Proposto com vantagem

$$\epsilon_3 \geq \left(\epsilon_2 - \frac{q_3}{2^{k_0-1}}\right)\left(1 - \frac{1}{q}\right)^{q_d}$$

e tempo de execução

$$t_3 \leq t_2 + q_3(T_{criptb} + c.n)$$

onde c é uma constante, $(n-k_0)$ é o tamanho em bits das mensagens não cifradas, q é a ordem de $\mathbb{G}_1$ e T_{criptb} é o tempo de execução de **criptb**.

***Demonstração*:** Decorre do Teorema 5.4 de (FUJISAKI; OKAMOTO, 2000).

Lema 6.3 *Suponha que H_2 é um oráculo aleatório. Seja $\mathcal{A}_3$ um adversário CPA contra o esquema Basico-Proposto, com vantagem ϵ_3, tempo de execução t_3 e que realiza no máximo q_2 consultas a H_2. Então existe um algoritmo $\mathcal{B}$ que resolve BDH em $\mathbb{G}_1$ com vantagem $\epsilon \geq 2\epsilon_3/q_2$, com complexidade de tempo $O(t_3))$.*

***Demonstração*:** Vamos construir um algoritmo $\mathcal{B}$ que resolve BDH, ao interagir com o adversário $\mathcal{A}_3$. Nossa demonstração é muito semelhante à do Lema 4.3 de (BONEH; FRANKLIN, 2001), que foi integralmente aproveitada na prova do Lema 3 de (CHENG; COMLEY, 2005).

Inicialmente, $\mathcal{B}$ recebe como entrada os parâmetros BDH $\langle q, \mathbb{G}_1, \mathbb{G}_2, P, \hat{e}\rangle$, produzidos a partir de um parâmetro de segurança k. Recebe, ainda, uma instância $\langle P, aP, bP, cP\rangle$, onde a, b, c são valores selecionados ao acaso de Z_q^*. O algoritmo $\mathcal{B}$ encontra o valor de $\hat{e}(P,P)^{abc}$, interagindo com $\mathcal{A}_3$ da seguinte forma:

Inicialização. $\mathcal{B}$ simula a execução do algoritmo **gerachavesb** para criar a chave pública $K_{pub} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_2, Q_A, N_A\rangle$, com $P_{pub} = aP$ (ou seja, $s = a$), $Q_A = bP$ e $N_A = t_A P$, onde t_A é escolhido aleatoriamente de $\mathbb{Z}_q^*$. A chave secreta $K_{sec} = d_A = sQ_A$, que $\mathcal{B}$ não conhece, é $d_A = abP$. H_2 é um oráculo aleatório controlado por $\mathcal{B}$. K_{pub} é entregue para $\mathcal{A}_3$ e o jogo prossegue nas fases a seguir.

Consultas. $\mathcal{B}$ responde às consultas de $\mathcal{A}_3$ ao oráculo H_2 (para criptografia dos textos) conforme segue:

consultasH$_2$ (X_i, Y_i, Z_i) : A qualquer instante $\mathcal{A}_3$ pode consultar H_2. $\mathcal{B}$ responde a essas consultas com o auxílio de uma lista de quáduplas $\langle X_i, Y_i, Z_i, H_i\rangle$, indexada pelos três primeiros termos. A lista, denotada por H_2^{lista}, inicialmente encontra-se vazia. $\mathcal{B}$ pode fornecer uma das duas respostas:

1. Se (X_i, Y_i, Z_i) indexa uma quádrupla na H_2^{lista}, então $\mathcal{B}$ responde com o H_i correspondente.
2. Caso contrário, $\mathcal{B}$ seleciona ao acaso uma cadeia $H_i \in \{0,1\}^n$, insere a quádupla $\langle X_i, Y_i, Z_i, H_i\rangle$ na lista e responde com H_i.

Desafio. $\mathcal{A}_3$ encerra a fase de consultas e fornece duas mensagens m_0 e m_1 de igual tamanho. $\mathcal{B}$ escolhe ao acaso uma cadeia $R \in \{0,1\}^n$, define o texto cifrado $C_{ch} = \langle U', V'\rangle = \langle cP, R\rangle$, que é passado para $\mathcal{A}_3$. Observe que a decriptografia de C_{ch} é:

$$V' \oplus H_2(U', \hat{e}(U', d_A), cN_A) = R \oplus H_2(cP, \hat{e}(cP, abP), cN_A)$$

Palpite. $\mathcal{A}_3$ gera um palpite $b \in \{0,1\}$ (que não vai ser utilizado). Neste momento, $\mathcal{B}$ escolhe ao acaso uma quádupla $\langle X_i, Y_i, Z_i, H_i\rangle$ de H_2^{lista}. $\mathcal{B}$ pressupõe que $X_i = cP$ e responde Y_i, como sendo a solução $\hat{e}(cP, abP) = \hat{e}(P,P)^{abc}$.

Seja $\mathcal{H}$ o evento em que o algoritmo $\mathcal{A}_3$ emite uma consulta sobre $H_2(cP, \hat{e}(cP, abP), cN_A) = H_2(X, Y, Z)$, em algum momento da simulação.

Afirmativa 1: $\Pr[\mathcal{H}]$ na simulação é igual a $\Pr[\mathcal{H}]$ num ataque real de $\mathcal{A}_3$.

Demonstração: Seja $\mathcal{H}_l$ o evento em que $\mathcal{A}_3$ faz uma consulta para $H_2(X,Y,Z)$ em alguma das primeiras l consultas ao oráculo H_2. Vamos provar, por indução em l, que $\Pr[\mathcal{H}_l]$ num ataque real é igual a $\Pr[\mathcal{H}_l]$ na simulação, para todo $l \geq 0$.

É imediato que $\Pr[\mathcal{H}_0]$=0 tanto na simulação quanto no ataque real. Suponha que $l > 0$ e temos $\Pr[\mathcal{H}_{l-1}]$ em igual valor no ataque real e na simulação. Vamos mostrar que o mesmo vale para $\Pr[\mathcal{H}_l]$. Sabemos que

$$\begin{aligned}\Pr[\mathcal{H}_l] &= \Pr[\mathcal{H}_l \mid \mathcal{H}_{l-1}]\Pr[\mathcal{H}_{l-1}] + \Pr[\mathcal{H}_l \mid \neg\mathcal{H}_{l-1}]\Pr[\neg\mathcal{H}_{l-1}] \\ &= \Pr[\mathcal{H}_{l-1}] + \Pr[\mathcal{H}_l \mid \neg\mathcal{H}_{l-1}]\Pr[\neg\mathcal{H}_{l-1}]\end{aligned}$$

Argumentamos que $\Pr[\mathcal{H}_l \mid \neg\mathcal{H}_{l-1}]$ na simulação é igual a $\Pr[\mathcal{H}_l \mid \neg\mathcal{H}_{l-1}]$ no ataque real. Basta observar que se $\mathcal{A}_3$ não faz uma consulta para $H_2(X,Y,Z)$ sua visão durante a simulação é idêntica à sua visão no ataque real. De fato, a chave pública e o desafio são uniformemente distribuídos, como no ataque real. Também todas as respostas a **consultas$\mathbf{H_2}$** são uniformes e independentes em $\{0,1\}^n$, da mesma forma que o oráculo aleatório H_2. Então $\Pr[\mathcal{H}_l \mid \neg\mathcal{H}_{l-1}]$ na simulação é igual a $\Pr[\mathcal{H}_l \mid \neg\mathcal{H}_{l-1}]$ no ataque real. E, pela hipótese de indução, segue que $\Pr[\mathcal{H}_l]$ no ataque real é igual a $\Pr[\mathcal{H}_l]$ na simulação. □

Afirmativa 2: Num ataque real, $\Pr[\mathcal{H}] \geq 2\epsilon_3$.

Demonstração: No ataque real, se $\mathcal{A}_3$ nunca fizer uma consulta para $H_2(X, Y, Z)$, então a decriptografia de C_{ch} é independente da visão de $\mathcal{A}_3$ (uma vez que $H_2(X, Y, Z)$ é independente da visão de $\mathcal{A}_3$). Por isso, no ataque real, $\Pr[b = b' \mid \neg\mathcal{H}]=1/2$. E, pela definição de $\mathcal{A}_3$, no ataque real $|\Pr[b = b']-1/2 \mid \geq \epsilon_3$. Esses dois fatos implicam que $\Pr[\mathcal{H}]\geq 2\epsilon_3$.

De fato, para se chegar a esse resultado, primeiramente vamos encontrar os limites inferior e superior para $\Pr[b = b']$:

$$\begin{aligned}
\Pr[\mathrm{b} = \mathrm{b}'] &= \Pr[\mathrm{b} = \mathrm{b}' \mid \neg\mathcal{H}]\Pr[\neg\mathcal{H}] + \Pr[\mathrm{b} = \mathrm{b}' \mid \mathcal{H}]\Pr[\mathcal{H}] \\
&\leq \Pr[\mathrm{b} = \mathrm{b}' \mid \neg\mathcal{H}]\Pr[\neg\mathcal{H}] + \Pr[\mathcal{H}] \\
&= \frac{1}{2}\Pr[\neg\mathcal{H}] + \Pr[\mathcal{H}] \\
&= \frac{1}{2}(1 - \Pr[\mathcal{H}]) + \Pr[\mathcal{H}] \\
&= \frac{1}{2} + \frac{1}{2}\Pr[\mathcal{H}]
\end{aligned}$$

$$\begin{aligned}
\Pr[\mathrm{b} = \mathrm{b}'] &\geq \Pr[\mathrm{b} = \mathrm{b}' \mid \neg\mathcal{H}]\Pr[\neg\mathcal{H}] \\
&= \frac{1}{2}\Pr[\neg\mathcal{H}] \\
&= \frac{1}{2}(1 - \Pr[\mathcal{H}]) \\
&= \frac{1}{2} - \frac{1}{2}\Pr[\mathcal{H}]
\end{aligned}$$

Então $\mid \Pr[b = b'] -1/2 \mid\leq \frac{1}{2}\Pr[\mathcal{H}]$ e $\epsilon_3 \leq|\Pr[b = b']-1/2 \mid\leq \frac{1}{2}\Pr[\mathcal{H}]$.

Portanto, num ataque real, $\Pr[\mathcal{H}] \geq 2\epsilon_3$. □

Decorre das duas afirmativas que, na simulação, $\Pr[\mathcal{H}] \geq 2\epsilon_3$. Então, ao fim da simulação, (X, Y, Z) aparece em alguma quádupla de H_2^{lista}, com probabilidade de pelo menos $2\epsilon_3$. Considerando-se que $\mathcal{A}_3$ faz q_2 consultas distintas, segue que $\mathcal{B}$ responde corretamente o cálculo de $\hat{e}(P, P)^{abc}$ com probabilidade de pelo menos $2\epsilon_3/q_2$.

Para análise da complexidade de tempo de $\mathcal{B}$, é preciso observar que, na fase de consultas, são realizadas q_2 operações com a lista H_2^{lista}. Então $t = q_2O(\log q_2) + O(1)$.

Por outro lado, $t_3 = q_2T_{criptb} + T_{palpite}$, onde T_{criptb} é o tempo de execução de **criptb** e $T_{palpite}$, tempo que $\mathcal{A}_3$ leva para produzir o palpite.

Pela definição de $\mathcal{A}_3$, seu tempo de execução é polinomial em k, bem como o tempo de **criptb**.

Considere os inteiros $0 \leq x, y, z = O(k)$, com $q_2 = O(k^x)$, $T_{criptb} = O(k^y)$, $T_{palpite} = O(k^z)$. Então

$t_3 = O(k^x)O(k^y) + O(k^z)$

e

$t = O(k^x)O(\log k^x) = O(k^x)O(\log k) = O(t_3)$

uma vez que $y > 0$, pois T_{criptb} envolve cálculo de emparelhamento $\hat{e}$. □

Lema 6.4 *Suponha que H_1 é um oráculo aleatório. Seja $\mathcal{A}_4$ um adversário Tipo-II-CCA2 contra o esquema CL-PKE-Proposto, com vantagem ϵ_4, tempo de execução t_4, que realiza no máximo q_1 consultas a H_1. Então existe um adversário $\mathcal{A}_2$ CCA2 contra Basico-Hib-Proposto com vantagem $\epsilon_2 \geq \epsilon_4/q_1$ e tempo de execução $t_2 \leq t_4 + c_{\mathbb{G}_1}(q_1)$, onde $c_{\mathbb{G}_1}$ denota o tempo de multiplicação de um ponto em $\mathbb{G}_1$ por um escalar.*

***Demonstração*:** Vamos construir um adversário $\mathcal{A}_2$ CCA2 que utiliza $\mathcal{A}_4$ para obter vantagem contra Basico-Hib-Proposto. Essa demonstração é muito semelhante à do Lema 6.1.

O jogo entre o desafiante $\mathcal{D}$ e o adversário $\mathcal{A}_2$ se inicia com o desafiante gerando os parâmetros públicos, com a execução do algoritmo **gerachavesh**, de Basico-Hib-Proposto. Como resultado, são obtidas a chave pública $K_{pub} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, k_0, P, P_{pub}, s, H_2, H_3, Q_A, N_A\rangle = \langle \mathsf{paramsh}, Q_A, N_A\rangle$ e a chave secreta $K_{sec} = \langle d_A = sQ_A, t_A\rangle$. O desafiante entrega K_{pub} e s para $\mathcal{A}_2$ e guarda K_{sec}.

O adversário $\mathcal{A}_2$ monta um ataque CCA2 contra Basico-Hib-Proposto utilizando K_{pub}, s e $\mathcal{A}_4$ da forma descrita a seguir.

$\mathcal{A}_2$ escolhe ao acaso um índice I, $1 \leq I \leq q_1$, que será associado a uma identidade sobre a qual $\mathcal{A}_2$ tentará aplicar o desafio. $\mathcal{A}_2$ simula o algoritmo **inicializa** de CL-PKE-Proposto, fornecendo para $\mathcal{A}_4$

$\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, k_0, P, P_{pub}, s, H_1, H_2, H_3\rangle (= \langle \mathsf{paramsh}, s, H_1\rangle)$

onde H_1 é um oráculo aleatório controlado por $\mathcal{A}_2$.

O adversário $\mathcal{A}_4$ pode realizar consultas a H_1 em qualquer instante. Essas consultas são manipuladas pelo algoritmo **consultasH$_1$**, conforme segue.

consultasH$_1$. Quando $\mathcal{A}_4$ consulta H_1 para ID_i, $\mathcal{A}_2$ responde sob auxílio de uma lista de triplas $\langle ID_j, Q_j, d_j\rangle$. A lista, denotada por H_1^{lista}, inicialmente encontra-se vazia. $\mathcal{A}_2$ pode fornecer uma das três respostas:

1. Se ID_i já aparece na H_1^{lista}, numa tripla $\langle ID_i, Q_i, d_i\rangle$, então $\mathcal{A}_2$ responde com $Q_i \in \mathbb{G}_1^*$.

2. Caso contrário, se a consulta é sobre o I-ésimo identificador distinto, então $\mathcal{A}_2$ calcula $d_A = sQ_A$ e armazena $\langle ID_I, Q_A, d_A \rangle$ na lista e responde $H_1(ID_I) = Q_A$.
3. Caso contrário, $\mathcal{A}_2$ seleciona ao acaso $Q_i \in \mathbb{G}_1^*$, calcula $d_i = sQ_i$, armazena $\langle ID_i, Q_i, d_i \rangle$ na lista e responde Q_i.

O jogo entre o desafiante $\mathcal{D}$ e o adversário $\mathcal{A}_2$ prossegue nas fases a seguir.

Fase 1. O adversário $\mathcal{A}_4$ lança a Fase 1 de seu ataque ao fazer uma série de consultas (de publicação de chave pública, extração de *SecDir* ou de decriptografia). Podemos considerar que $\mathcal{A}_4$ sempre fará uma consulta a H_1, para a mesma identidade sobre a qual tentará realizar as próximas consultas. $\mathcal{A}_2$ simula o desafiante de $\mathcal{A}_4$ e responde a essas consultas da seguinte forma:

- Publicação para ID_i. Para responder a esta consulta, $\mathcal{A}_2$ mantém outra lista, K^{lista}, com triplas na forma $\langle ID_i, t_i, N_i \rangle$, indexada por ID_i. Se ID_i já estiver na lista, $\mathcal{A}_2$ responde com N_i. Se ID_i não estiver na lista e $ID_i \neq ID_I$, $\mathcal{A}_2$ seleciona aleatoriamente um inteiro $t_i \in \mathbb{Z}_q^*$, insere a tripla $\langle ID_i, t_i, N_i = t_i P \rangle$ na lista e responde com N_i. Caso contrário, $\mathcal{A}_2$ armazena $\langle ID_I, \perp, N_A \rangle$ e responde com N_A.
- Extração de *SecDir* para ID_i). Se $ID_i \neq ID_I$, $\mathcal{A}_2$ devolve t_i, senão aborta o jogo (**evento 1**).
- Decriptografia sobre $\langle ID_i, C_i, N_i \rangle$. $\mathcal{A}_2$ procura em K^{lista} por uma tripla que contém N_i. Se tal tripla não existir, então $\mathcal{A}_2$ aborta o jogo (**evento 2**). Se a requisição de decriptografia foi para $C_i = \langle U, V \rangle$ sobre ID_I (ou seja, $ID_i = ID_I$), então $\mathcal{A}_2$ faz uma consulta de decriptografia sobre C_i e t_I, onde t_I é o valor que $\mathcal{A}_4$ usaria em seu ataque para eventualmente solicitar decriptografia sobre ID_I; $\mathcal{A}_2$ e repassa a resposta de $\mathcal{D}$ para $\mathcal{A}_4$. Em caso contrário, $\mathcal{A}_2$ tenta decriptografar consultando H_2 (que é controlado por $\mathcal{D}$) para obter $H_2(U, g^r, t_i U)$, onde $g^r = \hat{e}(U, d_i)$, d_i provém de H_1^{lista} e t_i pode ser obtido de K^{lista}. Ao desmembrar $V \oplus H_2(U, g^r, t_i U)$, $\mathcal{A}_2$ obtém um resultado que é repassado para $\mathcal{A}_4$.

Desafio. Em um dado momento, $\mathcal{A}_4$ encerra a Fase 1, escolhe ID_{ch}, N_{ch} e duas mensagens m_0, m_1 sobre as quais deseja desafiar. Se $ID_{ch} \neq ID_I$, então $\mathcal{A}_2$ aborta o jogo (**evento 3**); caso contrário, m_0, m_1 e N_{ch} são repassados para $\mathcal{D}$, que responde com o texto cifrado $C_{ch} = \langle U', V' \rangle$. $\mathcal{A}_2$ repassa para $\mathcal{A}_4$ o texto cifrado C_{ch}.

Fase 2. $\mathcal{A}_2$ continua a responder às requisições do mesmo modo que na Fase 1, porém aborta o jogo se for realizada alguma consulta de decriptografia sobre $\langle ID_I, C_{ch}, N_{ch} \rangle$ (**evento 4**).

Palpite. Se $\mathcal{A}_4$ emite um palpite b', $\mathcal{A}_2$ devolve o mesmo palpite b'.

Afirmativa: Se o algoritmo $\mathcal{A}_2$ não abortar durante a simulação, então o algoritmo $\mathcal{A}_4$ tem a mesma visão caso estivesse num ataque real.

Demonstração: As respostas de $\mathcal{A}_2$ às consultas a H_1 são independentes e uniformemente distribuídas em $\mathbb{G}_1^*$, pois H_1 é oráculo aleatório. Então $\mathcal{A}_4$ enxerga H_1, controlado por $\mathcal{A}_2$, da mesma forma como se estivesse aplicando um ataque real. Se $\mathcal{A}_2$ não abortar, todas as respostas às consultas de $\mathcal{A}_4$ são válidas (pertencentes aos domínios corretos). Em particular, $\mathcal{A}_2$ responde corretamente a consultas de decriptografia para $\mathcal{A}_4$:

1. Se $ID_i = ID_I$, $\mathcal{A}_2$ consulta $\mathcal{D}$ para decriptografar $\langle U, V\rangle$, com t_I e d_A. $\mathcal{D}$ produzirá a mesma resposta que $\mathcal{A}_4$ teria em um ataque real, pois $\mathcal{D}$ possui $d_A = sQ_A$, que é o mesmo valor de chave (parcial) que $\mathcal{A}_2$ possui e t_I é o mesmo valor que $\mathcal{A}_4$ usaria. Então, se $\mathcal{A}_2$ não abortar, o texto cifrado do desafio $\langle U', V'\rangle$ é uma criptografia válida para m_b em CL-PKE, e pode ser repassado para $\mathcal{A}_4$.
2. Caso contrário, $\mathcal{A}_2$ calcula $\hat{e}(U, d_i)$ e t_iU, para então obter $H_2(U, g^r, t_iU)$. Mas $\hat{e}(U, d_i) = \hat{e}(rP, sQ_i) = \hat{e}(P_{pub}, Q_i)^r = g^r$ e $t_iU = t_irP = rN_i$. Novamente $\mathcal{A}_2$ dá uma resposta válida para $\mathcal{A}_4$. □

Basta, então, calcularmos a probabilidade de $\mathcal{A}_2$ não abortar durante a simulação. São cinco os eventos que causam aborto:

- Evento 1, denotado por $\mathcal{H}_1$: $\mathcal{A}_4$ solicitou $SecDir$ para ID_I;
- Evento 2, denotado por $\mathcal{H}_2$: $\mathcal{A}_4$ solicitou decriptografia sob uma chave pública N_i desconhecida;
- Evento 3, denotado por $\mathcal{H}_3$: $\mathcal{A}_4$ não escolheu ID_I como ID_{ch};
- Evento 4, denotado por $\mathcal{H}_4$: $\mathcal{A}_2$ solicitou decriptografia para C_{ch} na Fase 2;

Tentativa de extração de $SecDir$ para ID_I não é permitida a $\mathcal{A}_4$. Como $\mathcal{A}_2$ apenas simula as requisições de $\mathcal{A}_4$, não ocorre $\mathcal{H}_1$ sem que $\mathcal{A}_4$ tenha abortado.

Uma solicitação de decriptografia com chave pública desconhecida não produz resultado (pois $\mathcal{A}_4$ receberia $\perp$ como resposta) e é possível para $\mathcal{A}_4$ evitar essa situação em sua implementação. Assim consideramos que $\mathcal{A}_2$ não recai em $\mathcal{H}_2$.

Também $\mathcal{H}_4$ só ocorre quando, no desafio, $\mathcal{A}_4$ escolheu $ID_I = ID_{ch}$, porém $\mathcal{A}_4$ seria abortado ao solicitar decriptografia sobre $\langle ID_{ch}, C_{ch}, N_{ch}\rangle$, da mesma forma que $\mathcal{A}_2$.

Além disso, não ter ocorrido $\mathcal{H}_3$ na fase do desafio (isto é, $ID_I = ID_{ch}$) implica que $\mathcal{H}_1$ não ocorreu (pois, se tivesse ocorrido, $\mathcal{A}_4$ teria pedido $SecEsq$ de ID_{ch} e seria abortado antes do desafio).

Logo, tem-se

$$\begin{aligned}\Pr[\mathcal{A}_2 \text{ não abortar}] &= \Pr[\neg\mathcal{H}_1 \wedge \neg\mathcal{H}_2 \wedge \neg\mathcal{H}_3 \wedge \neg\mathcal{H}_4] \\ &= \Pr[\neg\mathcal{H}_3 \wedge \neg\mathcal{H}_4] \\ &= \Pr[\neg\mathcal{H}_3]\end{aligned}$$

A escolha de I por $\mathcal{A}_2$ é independente da escolha de ID_I por $\mathcal{A}_4$, logo,

$$\Pr[\mathcal{A}_2 \text{ não abortar}] = 1/q_1$$

$\mathcal{A}_2$ usa o palpite de $\mathcal{A}_4$, cuja definição diz que $|\Pr[b = b'] - 1/2\,| \geq \epsilon_4$.

Se $\mathcal{A}_2$ não abortar e tiver sucesso no palpite, ganha o jogo. Combinando-se esses elementos, chega-se à vantagem:

$$\epsilon_2 \geq \epsilon_4/q_1$$

Para análise da complexidade de tempo de $\mathcal{A}_2$, observar os seguintes pontos:

- Como $\mathcal{A}_2$ basicamente simula o desafiante de $\mathcal{A}_4$, o tempo de execução de $\mathcal{A}_4$, denotado por t_4, é o principal componente.
- Na simulação, cada consulta a H_1 e cada requisição de uma chave pública envolvem uma multiplicação escalar em $\mathbb{G}_1$, que $\mathcal{A}_4$ possivelmente não realizaria num ataque real. Levando-se em consideração que $c_{\mathbb{G}_1}$ denota o tempo de multiplicação de um ponto em $\mathbb{G}_1$ por um escalar (aleatórios), e que a quantidade de requisições de chaves públicas é no máximo q_1, então $c_{\mathbb{G}_1}(q_1)$ é outro componente na delimitação de t_2.

Então $t_2 \leq t_4 + c_{\mathbb{G}_1}(q_1)$. □

Teorema 6.1 *Se o problema BDH é difícil em* $\mathbb{G}_1$ *e as funções de* hash H_1*,* H_2 *e* H_3 *são oráculos aleatórios, então o esquema CL-PKE-Proposto é IND-CCA2, ou seja, é seguro contra adversários Tipo-I-CCA2 e Tipo-II-CCA2.*

***Demonstração*:** Suponha que existe um adversário Tipo-I-CCA2 contra CL-PKE-Proposto, com vantagem ϵ_1 não ínfima, que realiza no máximo q_d consultas de decriptografia, q_e consultas de extração de chave secreta parcial e q_1, q_2, q_3 consultas a H_1, H_2, H_3, respectivamente, e com tempo de execução t_1.

Segue dos lemas 6.1, 6.2 e 6.3 que existe um algoritmo que resolve BDH com vantagem não ínfima

$$\epsilon_I \geq \frac{2}{q_2}\left(\frac{\epsilon_1}{q_1} - \frac{q_3}{2^{k_0-1}}\right)\left(1 - \frac{1}{q}\right)^{q_d}$$

e tempo de execução

$$t_I \leq t_1 + c_{\mathbb{G}_1}(q_1 + q_d + q_e) + q_3(T_{criptb} + c_0.n) + c_1$$

onde c_0, c_1 são constantes, $(n - k_0)$ é o tamanho em bits das mensagens não cifradas e T_{criptb} é o tempo de execução de **criptb**. Como t_1 é polinomial em k (bem como t_2 e t_3), t também é polinominal em k.

De forma análoga, suponha que existe um adversário Tipo-II-CCA2 contra CL-PKE-Proposto, com vantagem ϵ_2 não ínfima, que realiza no máximo q_d consultas de decriptografia e q_1, q_2, q_3 consultas a H_1, H_2, H_3, respectivamente, e com tempo de execução t_2.

Segue dos lemas 6.4, 6.2 e 6.3 que existe um algoritmo que resolve BDH com vantagem não ínfima

$$\epsilon_{II} \geq \frac{2}{q_2}\left(\frac{\epsilon_2}{q_1} - \frac{q_3}{2^{k_0-1}}\right)\left(1 - \frac{1}{q}\right)^{q_d}$$

e tempo de execução

$$t_{II} \leq t_2 + c_{\mathbb{G}_1}(q_1) + q_3(T_{criptb} + c_0.n) + c_1$$

Se renomearmos todos $q_i, i \in \{1, 2, 3\}$, por q_H, obtemos as aproximações

$$\epsilon_I \approx \frac{2\epsilon_1}{q_H^2}$$

$$\epsilon_{II} \approx \frac{2\epsilon_2}{q_H^2}$$

Logo CL-PKE-Proposto é seguro contra adversários Tipo-I-CCA2 e Tipo-II-CCA2. Ou seja, CL-PKE-Proposto é IND-CCA2.

□

6.2 Segurança de CL-PKS-Proposto

Nesta seção é apresentada a demonstração do fato de que CL-PKS-Proposto é seguro contra adversários Tipo-I-CMA e Tipo-II-CMA.

Para cada tipo de adversário, faremos uma redução de um problema pressuposto difícil, com a ajuda de um algoritmo auxiliar (um novo adversário derivado do primeiro).

6.2.1 Adversários CMA e Reduções

Para provar a segurança contra adversários Tipo-I-CMA e Tipo-II-CMA, primeiramente usaremos uma variante dos jogos Jogo-I e Jogo-II, em que o desafiante fornece uma identidade ID e o

adversário deve produzir uma assinatura válida para uma mensagem M qualquer e para aquela identidade ID dada. Dizemos que um esquema CL-PKS é existencialmente não-forjável contra ataques adaptativos de mensagem escolhida e *para uma dada identidade*, para adversários Tipo-I-CMA e Tipo-II-CMA, se qualquer adversário de complexidade de tempo polinomial em k tem probabilidade ínfima em obter sucesso.

O lema 6.5 a seguir mostra que, se nosso CL-PKS-Proposto for forjável, então existe um atacante mais fraco, desafiado sobre uma dada identidade. Usamos, então, esse adversário mais fraco para resolver o problema q-SDH (conforme discutido na seção 3.5), no lema 6.6 que trata adversários Tipo-I-CMA. Esse adversário mais fraco também é usado para resolver o problema BPI, no lema 6.7 que trata adversários Tipo-II-CMA.

Na figura 6.2 está esquematizado como o uso do lema 6.5 ajuda na redução dos problemas q-SDH e BPI. Para interpretração, onde se vê $A \rightarrow B$ (A implica B) significa que a existência de um algoritmo A de complexidade de tempo polinomial (em k) leva à existência de um algoritmo B, também polinomial.

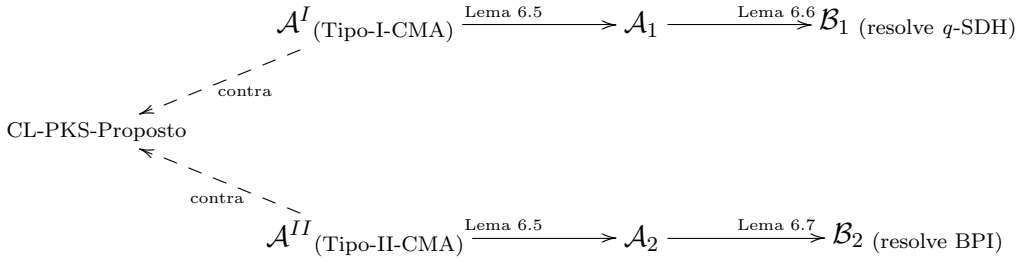

Figura 6.2: Seqüência de Reduções para Adversários contra CL-PKS-Proposto: de q-SDH para Tipo-I-CMA e de BPI para Tipo-II-CMA.

Lema 6.5 *Se existe um algoritmo $\mathcal{A}^I$ (respectivamente $\mathcal{A}^{II}$) para um ataque adaptativo de mensagem escolhida Tipo-I-CMA (resp. Tipo-II-CMA) contra CL-PKS-Proposto, com tempo de execução t, vantagem ϵ e que efetue q_{h_1} consultas ao oráculo aleatório H_1, então existe um algoritmo $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) que realiza ataque adaptativo de mensagem escolhida para uma dada identidade, Tipo-I-CMA (resp. Tipo-II-CMA), com vantagem $\epsilon_1 \geq \epsilon\left(1 - \frac{1}{2^k}\right)/q_{h_1}$, e tempo de execução $t_1 \leq t$ (resp. vantagem ϵ_2 e tempo t_2). Ademais, $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) faz a mesma quantidade de consultas que $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) faz.*

***Demonstração*:** Esta prova é similar à prova do lema 1 de (CHA; CHEON, 2003). Sem perda de generalidade, nós pressupomos que para cada ID, $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) realiza no máximo uma

consulta aos oráculos H_1, de Extração de Chave Secreta e de Requisição de Chave Pública; $\mathcal{A}^I$ pode ainda realizar uma consulta de Extração de Chave Parcial e uma Substituição de Chave Pública, para cada ID. $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) prossegue nos seguintes passos:

1. Escolha aleatoriamente $r \in_R \{1, \ldots, q_{h_1}\}$.
2. Seja ID_i a entrada da i-ésima consulta a H_1 solicitada por $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$).
3. Seja $ID'_i = ID$, se $i = r$; ou $ID'_i = ID_i$ em caso contrário.
4. Defina:
   - $H'_1(ID_i)$=$H_1(ID'_i)$
   - (Extração de Chave Secreta)$'(ID_i)$= (Extração de Chave Secreta)(ID'_i)
   - (Requisição de Chave Pública)$'(ID_i)$= (Requisição de Chave Pública)(ID'_i)
   - Assinatura$'(ID_i, m)$=Assinatura(ID'_i, m).
5. Exclusivamente para $\mathcal{A}^I$, defina adicionalmente:
   - (Extração de Chave Parcial)$'(ID_i)$= (Extração de Chave Parcial)(ID'_i)
   - (Substituição de Chave Pública)$'(ID_i)$= (Substituição de Chave Pública)(ID'_i)
6. Execute $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) com os parâmetros do sistema dados. $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) responde às consultas de $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) sobre H_1, H_2, Extração de Chave Secreta, Requisição de Chave Pública e de Assinatura, invocando respectivamente, H'_1, H_2, (Extração de Chave Secreta)$'$, (Requisição de Chave Pública)$'$ e Assinatura$'$. $\mathcal{A}^I$ ainda responde às consultas de Extração de Chave Parcial e de Substituição de Chave Pública, invocando respectivamente, (Extração de Chave Parcial)$'$ e (Substituição de Chave Pública)$'$.
7. Seja a saída de $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) igual a (ID_{out}, m, σ).
8. Se $ID_{out} = ID$, então $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) responde (ID, m, σ); caso contrário aborta o jogo.

Podemos facilmente observar que $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) realiza o mesmo número de consultas que $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) realiza durante um ataque. Também pode-se afirmar que o tempo de execução de $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) é no máximo t, pois eventualmente $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) leva significante tempo extra em análises das respostas das consultas.

Uma vez que as distribuições produzidas por H'_1, (Extração de Chave Secreta)$'$, (Requisição de Chave Pública)$'$, (Extração de Chave Parcial)$'$, (Substituição de Chave Pública)$'$ e Assinatura$'$ são indistinguíveis das produzidas por CL-PKS-Proposto, $\mathcal{A}^I$ (resp. $\mathcal{A}^{II}$) nada aprende sobre os resultados das consultas, logo.

$$Pr[(ID_{out}, m, \sigma) \text{ é válida}] \geq \epsilon$$

Como H_1 é um oráculo aleatório, é ínfima a probabilidade de que a saída (ID_{out}, m, σ) seja válida sem que uma consulta a $H_1'(ID_{out})$ tenha sido feita, isto é,

$$Pr[ID_{out} = ID_i \text{ para algum } i \mid (ID_{out}, m, \sigma) \text{ é válida }] \geq 1 - \tfrac{1}{2^k}$$

E, por r ser independente e aleatoriamente escolhido,

$$Pr[ID_{out} = ID_r \mid ID_{out} = ID_i \text{ para algum } i] \geq \frac{1}{q_{h_1}}$$

Combinando-se essas igualdades, tem-se:

$$\epsilon_1 = Pr[ID_{out} = ID_r = ID \text{ e } (ID_{out}, m, \sigma) \text{ é válida }] \geq \epsilon \left(1 - \tfrac{1}{2^k}\right) \left(\tfrac{1}{q_{h_1}}\right)$$

também vale:

$$\epsilon_2 \geq \epsilon \left(1 - \tfrac{1}{2^k}\right) \left(\tfrac{1}{q_{h_1}}\right)$$

Conclui-se, assim, a demonstração do lema. □

Lema 6.6 *Considere a existência de um algoritmo $\mathcal{A}_1$ que realiza ataque adaptativo de mensagem escolhida para uma dada identidade, Tipo-I-CMA, que realiza q_{h_i} consultas aos oráculos aleatórios H_i $(i = 1, 2)$ e q_s consultas de assinatura. Pressuponha que, num tempo t, $\mathcal{A}_1$ consegue forjar uma assinatura com probabilidade $\epsilon \geq 10(q_s + 1)(q_s + q_{h_2})/2^k$. Então existe um algoritmo $\mathcal{B}_1$ capaz de resolver o problema q-SDH em $(\mathbb{G}_1, \mathbb{G}_2)$ para $q = q_{h_1}$, num tempo esperado*

$$t' \leq 120686 q_{h_2} \frac{t + O(q_s(\tau_p + \tau_{exp})) + O(q\tau_{exp}) + O(\log(q_{h_2}))}{\epsilon(1 - q_s/2^k)} + O(q^2 \tau_{mult})$$

onde τ_p é o custo de calcular um emparelhamento bilinear, τ_{exp} representa o tempo de cálculo de uma exponenciação em $\mathbb{G}_T$ e τ_{mult} denota o tempo de uma multiplicação escalar em $\mathbb{G}_2$.

***Demonstração*:** Esta demonstração é similar à prova do lema 2 de (BARRETO et al., 2005). Vamos construir um algoritmo $\mathcal{B}_1$ que aproveita a habilidade de $\mathcal{A}_1$ para resolver q-SDH. Inicialmente, $\mathcal{B}_1$ recebe como entrada uma instância $(P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^q Q)$ e deseja encontrar um par $(c, \frac{1}{c+\alpha} P)$.

Na fase de inicializações, $\mathcal{B}_1$ constrói um gerador $G \in \mathbb{G}_1$ de modo que ele conheça $(q-1)$ pares na forma $(w_i, \frac{1}{w_i+\alpha} G)$ para $w_1, \ldots, w_{q-1} \in_R \mathbb{Z}_p^*$. Para tanto, $\mathcal{B}_1$:

1. Escolhe aleatoriamente $w_1, \ldots, w_{q-1} \leftarrow_R \mathbb{Z}_p^*$.
2. Expande $f(z) = \prod_{i=1}^{q-1}(z + w_i)$ para obter $c_0, \ldots, c_{q-1} \in \mathbb{Z}_p^*$ tal que $f(z) = \sum_{i=0}^{q-1}(c_i z^i)$.
3. Define os geradores
   $H = \sum_{i=0}^{q-1} c_i(\alpha^i Q) = f(\alpha)Q \in \mathbb{G}_2$ e
   $G = \psi(H) = f(\alpha)P \in \mathbb{G}_1$.
4. Define o parâmetro público $H_{pub} = \sum_{i=1}^{q} c_{i-1}(\alpha^i Q) \in \mathbb{G}_2$, de modo que $H_{pub} = \alpha H$, embora $\mathcal{B}_1$ não conheça α.
5. Para $1 \leq i \leq q-1$, $\mathcal{B}_1$ expande $f_i(z) = f(z)/(z + w_i) = \sum_{i=0}^{q-2}(d_i z^i)$ e

$$\sum_{i=0}^{q-2} d_i \psi(\alpha^i Q) = f_i(\alpha)P = \frac{f(\alpha)}{\alpha + w_i}P = \frac{1}{\alpha + w_i}G \tag{6.1}$$

   Os pares $(w_i, \frac{1}{w_i+\alpha}G)$ são calculados usando-se o lado esquerda da equação (6.1).

$\mathcal{B}_1$ escolhe aleatoriamente a identidade de desafio $ID^* \leftarrow_R \{0,1\}^*$; entrega para $\mathcal{A}_1$ essa ID^* e os parâmetros do sistema $\texttt{params} = \langle p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H, H_{pub}, g', H_1, H_2\rangle$, onde H_1 e H_2 são oráculos aleatórios controlados por $\mathcal{B}_1$ e $g' = e(G, H)$.

Por simplicidade, considere que:

1. as consultas a H_1 são todas distintas;
2. qualquer consulta a uma identidade ID é precedida por uma consulta a $H_1(ID)$ e
3. sempre é realizada uma consulta de (Requisição de Chave Púbica) para a identidade ID, antes que sejam consultados os oráculos de (Extração de Chave Secreta) e de (Substituição de Chave Púbica), sobre a mesma identidade ID.

$\mathcal{B}_1$ inicializa um contador l em 1 e cria duas listas inicialmente vazias L_1 e L_2. L_1 armazena quíntuplas na forma (ID, w, N, D, t), para valores de w aleatórios (escolhidos como acima), chave pública N, chave parcial secreta D e informação secreta t, relacionados com ID. Com a lista L_2, $\mathcal{B}_1$ controla suas respostas às consultas ao oráculo aleatório H_2, realizadas por $\mathcal{A}_1$ (para verificações de assinaturas).

$\mathcal{B}_1$ responde às consultas de $\mathcal{A}_1$ da seguinte maneira:

- **consultasH$_1$** sobre uma identidade ID: Se $ID \neq ID^*$, $\mathcal{B}_1$ armazena na lista L_1 uma quíntupla com os valores $(ID, w_l, N = \perp, D = \perp, t = \perp)$, responde w_l e incrementa l. Caso $ID = ID^*$, $\mathcal{B}_1$ escolhe aleatoriamente $w^* \in \mathbb{Z}_p^*$; se $w^* \in \{w_1, \ldots, w_{q-1}\}$, realiza novo sorteio. Armazena em L_1 $(ID, w^*, N = \perp, D = \perp, t = \perp)$ e responde w^*.

- **consultasH$_2$** sobre (M, ID, N, r): Se $\mathcal{B}_1$ não encontrar (M, ID, N, r, h_2) na lista L_2, aborta; caso contrário, responde h_2.
- **Requisição de Chave Pública** para ID: $\mathcal{B}_1$ recupera (ID, w, N, D, t) de L_1. Se $N \neq \perp$, responde N. Caso contrário, escolhe aleatoriamente $t \leftarrow_R \mathbb{Z}_p^*$, calcula $N = (g')^t$, atualiza em L_1 os valores de N e t; devolve N como resposta.
- **Extração de Chave Parcial** para ID: $\mathcal{B}_1$ recupera (ID, w, N, D, t) de L_1, atualiza e responde $D = (1/(\alpha + w))G$, calculado anteriormente.
- **Extração de Chave Secreta** de ID: $\mathcal{B}_1$ recupera (ID, w, N, D, t) de L_1. Se $D = \perp$ realiza uma consulta de Extração de Chave Parcial. Responde (D, t).
- **Substituição de Chave Pública** para ID com N': $\mathcal{B}_1$ recupera (ID, w, N, D, t) de L_1. Atualiza a quíntupla com $N = N'$ e $t = \perp$, em L_1.
- **Assinatura** sobre ID e M: $\mathcal{B}_1$ efetua os seguintes passos:
  1. Escolhe aleatoriamente $S \leftarrow_R \mathbb{G}_1$ e $h \leftarrow_R \mathbb{Z}_p^*$.
  2. Calcula $r = e(S, Q_{ID})N^{-h}$, onde $Q_{ID} = H_1(ID)H + H_{pub}$ e N pode ser extraído de L_1 (se a chave pública não tiver sido substituída, isto é, se $t \neq \perp$) ou $\mathcal{A}_1$ submete um valor t' correspondente à chave pública N (porém, se $ID = ID^*$ ou se $N \neq g^{t'}$, $\mathcal{B}_1$ é abortado).
  3. $\mathcal{B}_1$ define $H_2(M, ID, N, r)$ como sendo h, incluindo (M, ID, N, r, h) em L_2, porém, aborta se $H_2(M, ID, N, r)$ já tiver sido definido anteriormente.
  4. Responde $\sigma = (S, h)$.

Podemos afirmar que, se $\mathcal{B}_1$ não abortar durante a simulação acima, então $\mathcal{A}_1$ tem a mesma visão que teria caso estivesse em um ataque real. Essa afirmação é verdadeira uma vez que H_1 e H_2 são oráculos aleatórios e os valores de w e h (que os simulam) são escolhidos independente e uniformemente (respectivamente, na fase de inicializações e no tratamento de *consultas*H_2).

Ademais, $\mathcal{B}_1$ pode abortar (sem que $\mathcal{A}_1$ tenha abortado antes) apenas quando, numa consulta ao oráculo de assinatura, um mesmo valor de h é escolhido duas vezes. A probabilidade deste evento ocorrer é no máximo $q_s/2^k$, pois h_2 é escolhido ao acaso. Em outras palavras:

$$Pr[\mathcal{B}_1 \text{ não abortar}] \geq \epsilon \left(1 - \frac{q_s}{2^k}\right)$$

Agora, estamos prontos para aplicarmos o resultado do Lema da Bifurcação[1] (Teorema 3 de (POINTCHEVAL; STERN, 2000)) e procedermos na demonstração.

[1] Do original, em inglês, *Forking Lemma*.

O Lema da Bifurcação traduz, na essência, a seguinte idéia: considere um esquema que produza assinaturas na forma (M, r, h, S), e que exista um adversário capaz de produzir uma assinatura forjada num tempo esperado t, com probabilidade $\epsilon \geq 10(q_s + 1)(q_s + q_h)/2^k$, realizando q_s consultas de assinatura e q_h consultas a um oráculo aleatório. Se as triplas (r, h, S) podem ser simuladas sem o conhecimento da chave secreta de assinatura, então existe um algoritmo capaz de gerar duas assinaturas válidas (M, r, h_1, S_1) e (M, r, h_2, S_2), com $h_1 \neq h_2$, num tempo esperado $t' \leq 120686 q_h(t/\epsilon)$.

Em nosso caso, a partir de $\mathcal{A}_1$, construímos um algoritmo $\mathcal{A}$' que reexecuta $\mathcal{A}_1$, usando ID* e $\texttt{params} = \langle p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H, H_{pub}, g', H_1, H_2 \rangle$, até obter duas assinaturas forjadas (M^*, r, S_1, h_1) e (M^*, r, S_2, h_2), com $h_1 \neq h_2$.

$\mathcal{B}_1$ então executa $\mathcal{A}$' para obter (M^*, r, S_1, h_1) e (M^*, r, S_2, h_2) para a mesma mensagem M^* e mesmo r. $\mathcal{B}_1$ recupera (ID^*, w^*, t^*) de L_1. Como as duas assinaturas são válidas, para o mesmo valor r, tem-se:

$$e(S_1, Q_{ID^*})(N^*)^{-h_1} = e(S_2, Q_{ID^*})(N^*)^{-h_2}$$

onde $Q_{ID^*} = H_1(ID^*)H + H_{pub} = (w^* + \alpha)H$. Então, valem as igualdades:

$$\begin{aligned}
e(S_1, Q_{ID^*})(e(G,H)^{t^*})^{-h_1} &= e(S_2, Q_{ID^*})(e(G,H)^{t^*})^{-h_2} \\
e(S_1, Q_{ID^*})(e(G,H)^{-t^*h_1}) &= e(S_2, Q_{ID^*})(e(G,H)^{-t^*h_2}) \\
e(S_1, Q_{ID^*})e(S_2, Q_{ID^*})^{-1} &= (e(G,H)^{-t^*h_1})^{-1}e(G,H)^{-t^*h_2} \\
e((S_1 - S_2), Q_{ID^*}) &= e(G,H)^{t^*(h_1-h_2)} \\
e([t^*(h_1 - h_2)]^{-1}(S_1 - S_2), Q_{ID^*}) &= e(G,H) \\
(t^*(h_1 - h_2))^{-1}(S_1 - S_2) &= (1/(\alpha + w^*))G
\end{aligned}$$

A chave pública de ID^* não pode ter sido substituída (pois caso contrário t^* seria igual a $\perp$, ou $\mathcal{A}_1$ forneceria $t' = t^*$, o que significaria corrompimento de chave secreta). Portanto, se $\mathcal{B}_1$ não abortar, o valor de t^* pode ser obtido de L_1. Já os valores de (h_1, h_2, S_1, S_2) são fornecidos por $\mathcal{A}$'. Logo, $T^* = [t^*(h_1 - h_2)]^{-1}(S_1 - S_2)$ é calculável por $\mathcal{B}_1$.

A partir do valor de T^*, $\mathcal{B}_1$ procede com a técnica apresentada em (BARRETO et al., 2005) e (BONEH; BOYEN, 2004a) para extrair $\sigma^* = (1/(\alpha + w^*))P$:

$\mathcal{B}_1$ obtém $\gamma_{-1}, \gamma_0, \ldots, \gamma_{q-2} \in \mathbb{Z}_p^*$, tais que

$$f(z)/(z + w^*) = \gamma_{-1}/(z + w^*) + \sum_{i=0}^{q-2}(\gamma_i z^i)$$

e calcula

$$\sigma^* = \frac{1}{\gamma_{-1}}\left[T^* - \sum_{i=0}^{q-2}\gamma_i \psi(\alpha^i Q)\right] = \frac{1}{\alpha + w^*}P$$

$\mathcal{B}_1$ então gera (w^*, σ^*) como resposta da instância dada de q-SDH.

Para calcular a complexidade de tempo de execução de $\mathcal{B}_1$, considere os resultados:

- $\mathcal{A}_1$ forja uma assinatura em tempo t com probabilidade $\epsilon \geq 10(q_s+1)(q_s+q_{h_2})/2^k$ (para que seja aplicável o Lema da Bifurcação);
- $Pr[\mathcal{B}_1 \text{ não abortar}] \geq \epsilon(1 - q_s/2^k)$;
- o custo de processamento da fase de inicializações é $O(q_{h_1}^2 \tau_{mult})$, onde τ_{mult} representa o custo de uma multiplicação escalar em $\mathbb{G}_2$;
- o tempo de processamento das consultas a H_2 é $O(\log(q_{h_2}))$, pois envolve apenas tratamento da lista L_2.
- o número de consultas para extração de chave parcial, extração de chave secreta e substituição de chave pública é $O(q_{h_1})$. Como essas consultas, bem como **consultasH$_1$**, apenas efetuam operações sobre a lista L_1, tem-se que o custo de processamento das consultas de extração de chave parcial, extração de chave secreta, substituição de chave pública e consultasH_1 é $O(\log(q_{h_1}))$;
- o número de consultas para requisição de chave pública é $O(q_{h_1})$. Como essas consultas envolvem exponenciação em $\mathbb{G}_T$, sua complexidade de tempo é $O(q_{h_1}\tau_{exp})$, onde τ_{exp} representa o tempo de cálculo de uma exponenciação em $\mathbb{G}_T$;
- o custo de processamento das consultas de assinatura é $O(q_s(\tau_p+\tau_{exp}))$, em que τ_p denota o custo de cálculo de um emparelhamento bilinear;
- logo, o custo total de processamento de todos os tipos de consulta é

$$O(q_s(\tau_p + \tau_{exp})) + O(q_{h_1}\tau_{exp}) + O(\log(q_{h_1})) + O(\log(q_{h_2})) =$$
$$O(q_s(\tau_p + \tau_{exp})) + O(q_{h_1}\tau_{exp}) + O(\log(q_{h_2}))$$

- $q = q_{h_1}$.

Assim $\mathcal{B}_1$ resolve q-SDH em $(\mathbb{G}_1, \mathbb{G}_2)$ em tempo esperado:

$$t' \leq 120686 q_{h_2} \frac{t + O(q_s(\tau_p + \tau_{exp})) + O(q\tau_{exp}) + O(\log(q_{h_2}))}{\epsilon(1 - q_s/2^k)} + O(q^2\tau_{mult})$$

A inequação acima, pode ser simplificada em alguns casos.

Como $\mathcal{A}_1$ é um adversário de mensagem escolhida, tipicamente pode-se usar a aproximação $q_{h_1} = O(q_s)$ e $q_{h_2} = O(q_s)$. Dependendo das escolhas dos grupos bilineares e da implementação,

pode valer ainda a aproximação $O(q_s(\tau_p + \tau_{exp})) + O(q\tau_{exp}) + O(\log(q_{h_2})) = O(q_s\tau_p)$ de modo que

$$t' \leq 120686 q_{h_2} \frac{t + O(q_s\tau_p)}{\epsilon(1 - q_s/2^k)} + O(q^2\tau_{mult})$$

Conclui-se, então, a demonstração do lema. □

Lema 6.7 *Considere a existência de um algoritmo $\mathcal{A}_2$ que realiza ataque adaptativo de mensagem escolhida para uma dada identidade, Tipo-II-CMA, que realiza q_{h_2} consultas ao oráculo aleatório H_2 e q_s consultas de assinatura. Pressuponha que, num tempo t, $\mathcal{A}_2$ consegue forjar uma assinatura com probabilidade $\epsilon \geq 10(q_s + 1)(q_s + q_{h_2})/2^k$. Então existe um algoritmo $\mathcal{B}_2$ capaz de resolver o problema BPI, num tempo esperado*

$$t' \leq 120686 q_{h_2} \frac{t + O(q_s(\tau_p + \tau_{exp})) + O(q(\tau_{exp} + \tau_{mult})) + O(\log(q_{h_2}))}{\epsilon(1 - q_s/2^k)}$$

onde τ_p é o custo de calcular um emparelhamento bilinear, τ_{exp} representa o tempo de cálculo de uma exponenciação em $\mathbb{G}_T$ e τ_{mult} denota o tempo de uma multiplicação escalar em $\mathbb{G}_1$.

***Demonstração*:** Vamos construir um algoritmo $\mathcal{B}_2$ que aproveita a habilidade de $\mathcal{A}_2$ para resolver BPI. Inicialmente, $\mathcal{B}_2$ recebe como entrada os pontos $H \in \mathbb{G}_2$ e $e(G, H) \in \mathbb{G}_T$ e deseja encontrar o ponto $G = \alpha P \in \mathbb{G}_1$.

Na fase de inicializações, $\mathcal{B}_2$ escolhe aleatoriamente a identidade de desafio $ID^* \leftarrow_R \{0,1\}^*$ e a chave-mestra $s \leftarrow_R Z_p^*$.

$\mathcal{B}_2$ faz H ser gerador de $\mathbb{G}_2$ e $P = \psi(H)$ gerador de $\mathbb{G}_1$. Calcula $H_{pub} = sH$. Entrega para $\mathcal{A}_2$ ID^*, os parâmetros do sistema $\texttt{params} = \langle p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, P, H, H_{pub}, g', H_1, H_2 \rangle$ e a chave-mestra s, sendo que H_2 é um oráculo aleatório controlado por $\mathcal{B}_2$ e $g' = e(P, H)$. Observe que H_1 é o oráculo original, gerado por **inicializa**, e que servirá para cálculo das chaves parciais, tanto para $\mathcal{A}_2$ quanto para $\mathcal{B}_2$, uma vez que ambos conhecem o valor da chave-mestra s.

$\mathcal{B}_2$ cria duas listas inicialmente vazias L_0 e L_2. L_0 armazena quáduplas na forma (ID, N, D, t), para valores de chave pública N, chave parcial secreta D e informação secreta t, relacionados com ID. Com a lista L_2, $\mathcal{B}_2$ controla suas respostas às consultas ao oráculo aleatório H_2, realizadas por $\mathcal{A}_2$ (para verificações de assinaturas).

$\mathcal{B}_2$ responde às consultas de $\mathcal{A}_2$ da seguinte maneira:

- **consultasH$_2$** sobre (M, ID, N, r): Se $\mathcal{B}_2$ não encontrar (M, ID, N, r, h_2) na lista L_2, aborta; caso contrário, responde h_2.

- **Requisição de Chave Pública** para ID: Se $ID = ID^*$, calcula e armazena $(ID^*, N^* = e(G,H), D = (1/(H_1(ID^*)+s)P, t^* = \perp)$, em L_0. Caso contrário, escolhe aleatoriamente $t \leftarrow_R Z_p^*$, calcula e armazena $(ID, N = (g')^t, D = (1/(H_1(ID)+s)P, t)$ em L_0. Em ambos os casos, $\mathcal{B}_2$ devolve N como resposta. Observe que quando $ID = ID^*$, $N^* = e(G,H) = e((t^*)P, H) = e(P,H)^{t^*}$; também $G = \alpha P = (t^*)P$.
- **Extração de Chave Secreta** de ID: Se $ID = ID^*$, $\mathcal{B}_2$ aborta, senão recupera (ID, N, D, t) de L_0. Se necessário for, executa antes uma consulta de Requisição de Chave Pública para ID. Responde (D, t).
- **Assinatura** sobre ID e M: $\mathcal{B}_2$ efetua os seguintes passos:
  1. Escolhe aleatoriamente $S \leftarrow_R \mathbb{G}_1$ e $h \leftarrow_R \mathbb{Z}_p^*$.
  2. Calcula $r = e(S, Q_{ID})N^{-h}$, onde $Q_{ID} = H_1(ID)H + H_{pub}$ e N pode ser extraído de L_0.
  3. $\mathcal{B}_2$ define $H_2(M, ID, N, r)$ como sendo h, incluindo (M, ID, N, r, h) em L_2, porém, aborta se $H_2(M, ID, N, r)$ já tiver sido definido anteriormente.
  4. Responde $\sigma = (S, h)$.

Podemos afirmar que, se $\mathcal{B}_2$ não abortar durante a simulação acima, então $\mathcal{A}_2$ tem a mesma visão que teria caso estivesse em um ataque real. Essa afirmação é verdadeira uma vez que H_2 é oráculo aleatório e os valores de h (que o simulam) são escolhidos independente e uniformemente, no tratamento de $consultas H_2$.

Ademais, $\mathcal{B}_2$ pode abortar (sem que $\mathcal{A}_2$ tenha abortado antes) apenas quando, numa consulta ao oráculo de assinatura, um mesmo valor de h é escolhido duas vezes. A probabilidade deste evento ocorrer é no máximo $q_s/2^k$, pois h_2 é escolhido ao acaso. Em outras palavras,

$$Pr[\mathcal{B}_2 \text{ não abortar}] \geq \epsilon \left(1 - \frac{q_s}{2^k}\right)$$

Agora, estamos prontos para aplicarmos o resultado do Lema da Bifurcação. A partir de $\mathcal{A}_2$, construímos um algoritmo $\mathcal{A}$' que reexecuta $\mathcal{A}_2$, usando ID* e $\texttt{params} = \langle p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, P, H, H_{pub}, g', H_1, H_2 \rangle$, até obter duas assinaturas forjadas (M^*, r, S_1, h_1) e (M^*, r, S_2, h_2), com $h_1 \neq h_2$. $\mathcal{B}_2$ então executa $\mathcal{A}$' para obter tais forjas, para a mesma mensagem M^* e mesmo r. Como as duas assinaturas são válidas, tem-se:

$$e(S_1, Q_{ID^*})(N^*)^{-h_1} = e(S_2, Q_{ID^*})(N^*)^{-h_2}$$

onde $Q_{ID^*} = H_1(ID^*)H + H_{pub} = (H_1(ID^*) + s)H$. Então, valem as igualdades:

$$e(S_1, Q_{ID^*})(e(P,H)^{t^*})^{-h_1} \quad = \quad e(S_2, Q_{ID^*})(e(P,H)^{t^*})^{-h_2}$$

$$
\begin{aligned}
e(S_1, Q_{ID^*})(e(P,H)^{-t^* h_1}) &= e(S_2, Q_{ID^*})(e(P,H)^{-t^* h_2}) \\
e(S_1, Q_{ID^*}) e(S_2, Q_{ID^*})^{-1} &= (e(P,H)^{-t^* h_1})^{-1} e(P,H)^{-t^* h_2} \\
e((S_1 - S_2), Q_{ID^*}) &= e(P,H)^{t^*(h_1 - h_2)} \\
e([t^*(h_1 - h_2)]^{-1}(S_1 - S_2), Q_{ID^*}) &= e(P,H) \\
(t^*(h_1 - h_2))^{-1}(S_1 - S_2) &= (1/(H_1(ID^*) + s)P \\
(h_1 - h_2)^{-1}(H_1(ID^*) + s)(S_1 - S_2) &= (t^*)P
\end{aligned}
$$

$\mathcal{B}_2$ calcula $(t^*)P = G$, que é a resposta da instância dada de BPI.

Para calcular a complexidade de tempo de execução de $\mathcal{B}_2$, considere os resultados:

- $\mathcal{A}_2$ forja uma assinatura em tempo t com probabilidade $\epsilon \geq 10(q_s+1)(q_s+q_{h_2})/2^k$ (para que seja aplicável o Lema da Bifurcação);
- $Pr[\mathcal{B}_2 \text{ não abortar}] \geq \epsilon(1 - q_s/2^k)$;
- o tempo de processamento das consultas a H_2 é $O(\log(q_{h_2}))$, pois envolve apenas tratamento da lista L_2.
- o número de consultas para requisição de chave pública é $O(q_{h_1})$. Como essas consultas envolvem exponenciação em $\mathbb{G}_T$ e multiplicação escalar em $\mathbb{G}_1$, sua complexidade de tempo é $O(q_{h_1}(\tau_{exp} + \tau_{mult}))$, onde τ_{exp} representa o tempo de cálculo de uma exponenciação em $\mathbb{G}_T$ e τ_{mult}, cálculo de uma multiplicação escalar em $\mathbb{G}_1$;
- o número de consultas para extração de chave secreta é $O(q_{h_1})$. Como essas consultas apenas efetuam operações sobre a lista L_0, tem-se que seu custo é $O(\log(q_{h_1}))$;
- o custo de processamento das consultas de assinatura é $O(q_s(\tau_p + \tau_{exp}))$, em que τ_p denota o custo de cálculo de um emparelhamento bilinear;
- logo, o custo total de processamento de todos os tipos de consulta é

$$
\begin{aligned}
&O(q_s(\tau_p + \tau_{exp})) + O(q_{h_1}(\tau_{exp} + \tau_{mult})) + O(\log(q_{h_1})) + O(\log(q_{h_2})) = \\
&O(q_s(\tau_p + \tau_{exp})) + O(q_{h_1}(\tau_{exp} + \tau_{mult})) + O(\log(q_{h_2}))
\end{aligned}
$$

- $q = q_{h_1}$.

Assim $\mathcal{B}_2$ resolve BPI em tempo esperado:

$$
t' \leq 120686 q_{h_2} \frac{t + O(q_s(\tau_p + \tau_{exp})) + O(q(\tau_{exp} + \tau_{mult})) + O(\log(q_{h_2}))}{\epsilon(1 - q_s/2^k)}
$$

A inequação acima, pode ser simplificada em alguns casos.

Como $\mathcal{A}_2$ é um adversário de mensagem escolhida, tipicamente pode-se usar a aproximação $q_{h_1} = O(q_s)$ e $q_{h_2} = O(q_s)$. Dependendo das escolhas dos grupos bilineares e da implementação, pode valer ainda a aproximação $O(q_s(\tau_p + \tau_{exp})) + O(q(\tau_{exp} + \tau_{mult})) + O(\log(q_{h_2})) = O(q_s\tau_p)$ de modo que

$$t' \leq 120686 q_{h_2} \frac{t + O(q_s\tau_p)}{\epsilon(1 - q_s/2^k)}$$

Conclui-se, então, a demonstração do lema. □

Teorema 6.2 *Se os problemas q-SDH e BPI são difíceis sobre o grupo bilinear $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ e as funções de* hash *H_1 e H_2 são oráculos aleatórios, então o esquema CL-PKS-Proposto é EUF-CMA, ou seja, é seguro contra adversários Tipo-I-CMA e Tipo-II-CMA.*

***Demonstração*:** A prova segue da aplicação imediata dos lemas 6.5, 6.6 e 6.7.

Para adversário Tipo-I-CMA, suponha que existe $\mathcal{A}^I$ com tempo de execução t_1 e vantagem ϵ_1 tal que

$$\epsilon_1 \left(1 - \frac{1}{2^k}\right) \left(\frac{1}{q_{h_1}}\right) \geq 10(q_s + 1)(q_s + q_{h_2}) \left(\frac{1}{2^k}\right)$$

Então, pelo lema 6.6, exite um algoritmo capaz de resolver q-SDH num tempo esperado

$$t'_1 \leq 120686 q_{h_2} \frac{t + O(q_s(\tau_p + \tau_{exp})) + O(q\tau_{exp}) + O(\log(q_{h_2}))}{\epsilon_1 \left(1 - \frac{1}{2^k}\right) \left(\frac{1}{q_{h_1}}\right) \left(1 - \frac{q_s}{2^k}\right)} + O(q^2\tau_{mult})$$

Analogamente para adversário Tipo-II-CMA, suponha que existe $\mathcal{A}^{II}$ com tempo de execução t_2 e vantagem ϵ_2 tal que

$$\epsilon_2 \left(1 - \frac{1}{2^k}\right) \left(\frac{1}{q_{h_1}}\right) \geq 10(q_s + 1)(q_s + q_{h_2}) \left(\frac{1}{2^k}\right)$$

Então, pelo lema 6.7, exite um algoritmo capaz de resolver BPI num tempo esperado

$$t'_2 \leq 120686 q_{h_2} \frac{t + O(q_s(\tau_p + \tau_{exp})) + O(q\tau_{exp}) + O(\log(q_{h_2}))}{\epsilon_2 \left(1 - \frac{1}{2^k}\right) \left(\frac{1}{q_{h_1}}\right) \left(1 - \frac{q_s}{2^k}\right)} + O(q^2\tau_{mult})$$

Conclui-se, então, a demonstração do teorema. □

6.3 Resumo

Neste capítulo foram apresentadas as demonstrações dos fatos de que CL-PKE-Proposto e CL-PKS-Proposto são seguros, sob a hipótese de dificuldade de problemas descritos na seção 3.5 e no modelo de oráculos aleatórios.

Capítulo 7

Análises sobre os Esquemas Propostos

Neste capítulo são analisadas características dos esquemas CL-PKE-Proposto e CL-PKS-Proposto, comparativamente aos demais trabalhos relacionados e estudados. Foram levados em conta os aspectos abaixo listados, respectivamente desenvolvidos nas próximas seções. Antes, porém, apresentamos algumas considerações sobre a métrica adotada para tratarmos o tópico eficiência.

- Viabilidade de Implementação
- Eficiência Computacional
- Uso de Espaço (tamanhos de chave e mensagem)
- Modelo de Segurança
- Resumo de Vantagens e Desvantagens

7.1 Métrica Considerada em Eficiência Computacional

Todos os esquemas estudados e propostos são baseados em propriedades de emparelhamentos bilineares, sobre grupos elípticos. A implementação de algoritmos que realizam operações sobre tais grupos é bastante dependente das características das curvas elípticas, que geraram esses grupos.

Para que seja possível uma comparação de eficiência computacional entre os vários esquemas estudados e propostos, faz-se necessária a fixação do grupo bilinear, isto é, do conjunto $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$, tal que $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. Em alguns dos esquemas, $\mathbb{G}_1 = \mathbb{G}_2$.

Fixado o grupo bilinear, há ainda que se fixar algoritmos e implementações desses algoritmos, para as operações algébricas como soma de pontos, multiplicação escalar, exponenciação e cálculo de emparelhamento. Isso é importante, pois muitos dos algoritmos conhecidos aproveitam características algébricas de grupos específicos, para aumentarem eficiência computacional. Também implementações específicas (sobre uma plataforma) de tais algoritmos podem influenciar nos critérios de comparação.

Em outras palavras, para um mesmo grupo bilinear, é possível encontrar pares ⟨algoritmos, implementações⟩ em que a relação de tempo computacional das operações algébricas sobre os grupos fixados muda drasticamente. Por exemplo, em uma determinada implementação, uma operação de exponenciação pode custar mais que uma operação de multiplicação escalar, enquanto que em outra implementação ocorre o inverso.

Optamos, portanto, por comparar os esquemas em função da quantidade necessária de cada uma das operações mais custosas computacionalmente. São elas: cálculo de emparelhamento bilinear, multiplicação escalar, multiexponenciação (multiplicação de pontos da curva elíptica), exponenciação de um ponto e cálculo de uma função de *hash*.

Na tabela 7.1, essas operações de maior custo estão listadas numa ordenação típica de complexidade computacional. Assim, por essa tabela tem-se que, tipicamente, um cálculo de emparelhamento é a operação mais demorada de todas, enquanto o cálculo de uma função de *hash* demanda menos tempo. As operações de soma e multiplicação de pontos são bem mais velozes que as três primeiras (emparelhamento, exponenciação e multiplicção escalar) e, em geral, podem ser desconsideradas nas análises de complexidade dos algoritmos.

As operações de *hash* também podem ser eficientes ao ponto de não serem relevantes na análise de complexidade. Entretanto, se forem implementadas técnicas semelhantes às de (BONEH; BOYEN, 2004a), para eliminação da hipótese de oráculos aleatórios, são acrescentadas exponenciações, que elevam a ordem de grandeza computacional para o cálculo de *hash* (ver sugestões de trabalhos futuros, na seção 8.2.2).

Nas tabelas comparativas apresentadas nas seções seguintes sobre eficiência computacional, são contabilizadas as quantidades de cada tipo de operação. Em alguns dos esquemas, é possível a implementação de otimizações por meio de consultas a valores pré-calculados. Nas tabelas a seguir, foram desconsideradas pré-computações.

E	Emparelhamento bilinear
X	eXponenciação nos grupos multiplicativos
M	Multiplicação escalar nos grupos aditivos
m	multiplicação de pontos nos grupos multiplicativos
S	Soma de pontos nos grupos aditivos
H	cálculo de *Hash*

Tabela 7.1: Operações sobre grupos bilineares, em ordem de complexidade (e legenda)

7.2 Análises sobre CL-PKE-Proposto

Seguem comentários e análises sobre nosso esquema CL-PKE-Proposto.

7.2.1 Viabilidade de Implementação

Levando-se em conta o resumo apresentado por (GALBRAITH; PATERSON; SMART, 2006), a respeito do uso de emparelhamentos bilineares em esquemas criptográficos, chegamos à conclusão de que CL-PKE-Proposto e demais esquemas relacionados estudados são todos passíveis de serem implementados, ou seja, podem ser implementados na prática, desde que os grupos estejam definidos sobre corpos de característica prima grande (e grau de imersão 2). Isso é devido à escolha de emparelhamentos simétricos, na definição de todos os esquemas que podem ser comparados com CL-PKE-Proposto.

Detalhes sobre a viabilidade de implementação da função de *hash* $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$ é descrita em (BONEH; FRANKLIN, 2001).

7.2.2 Eficiência Computacional

Considerando-se que as operações mais caras envolvidas nos esquemas são respectivamente as listadas na tabela 7.1, podemos obter uma comparação de tempo computacional para criptografar e decriptografar textos, com cada um dos esquemas CL-PKE. A tabela 7.2 relaciona, em ordem cronológica de publicação, os esquemas relacionados com CL-PKE-Proposto, totalizando as operações que envolvem pontos da curva.

Pela tabela 7.2, é possível observar que, em implementações típicas, CL-PKE-Proposto deve ser o mais veloz para criptografar mensagens, pois requer apenas um cálculo de emparelhamento

Esquema CL-PKE	Criptografia				Decriptografia			
	E	X	M	H	E	X	M	H
(AL-RIYAMI; PATERSON, 2003)	3	1	1	4	1	0	1	3
(AL-RIYAMI; PATERSON, 2005)	1	1	2	5	1	0	2	4
(CHENG; COMLEY, 2005)	1	1	2	4	1	0	2	3
CL-PKE-Proposto	1	1	2	3	1	0	2	2

Tabela 7.2: Quantidade de operações sobre grupos nos esquemas CL-PKE

e menor quantidade de *hash* que todos os antecessores.

Para decriptografar, o esquema de (AL-RIYAMI; PATERSON, 2003) apresenta a solução mais eficiente, entre todos. Nosso esquema é ligeiramente mais veloz que o de (CHENG; COMLEY, 2005), por usar uma função de *hash* a menos.

Salientamos que o uso de menor número de operações de *hash* ocorre não apenas nos algoritmos de criptografia e decriptografia. No próprio algoritmo de inicializações de CL-PKE-Proposto, fazem parte dos parâmetros públicos do sistema apenas três funções de *hash*, a menor quantidade dentre todos os esquemas.

Se forem implementadas técnicas semelhantes às de (BONEH; BOYEN, 2004a), para eliminação da hipótese de oráculos aleatórios, o cálculo de *hash* se torna mais caro, porém a eficiência de nosso algoritmo de decriptografia se torna praticamente equivalente à do mais veloz (ver seção 8.2.2).

7.2.3 Uso de Espaço

A tabela 7.3 lista os tamanhos de chave pública, de texto legível e de texto cifrado, requeridos pelos esquemas. Por essa tabela, pode-se observar que CL-PKE-Proposto requer chaves públicas de bom tamanho frente aos demais esquemas. E, potencialmente, produz cifras menores que seus antecessores.

Considerando-se que o tamanho da mensagem para CL-PKE-Proposto, em bits, é $m - k_0$, e n para os demais esquemas, é possível realizar uma comparação dos tamanhos da cifras produzidas, fazendo $n = m - k_0$. Para determinadas escolhas de m e k_0, com k_0 polinomial em m, o esquema proposto produz cifras de menor tamanho conforme se vê na tabela 7.4.

Para que CL-PKE-Proposto produza cifras de menor tamanho, é preciso que se escolha

Esquema CL-PKE	Tamanhos (em bits)		
	chave pública	mensagem legível	mensagem cifrada
(AL-RIYAMI; PATERSON, 2003)	$2g$	n	$g + 2n$
(AL-RIYAMI; PATERSON, 2005)	g	n	$g + 2n$
(CHENG; COMLEY, 2005)	g	n	$g + 2n$
CL-PKE-Proposto	g	$m - k_0$	$g + m$

Tabela 7.3: Tamanhos requeridos na representação de elementos nos esquemas CL-PKE (em que g é o tamanho em bits para representar um ponto de $\mathbb{G}_1$)

$k_0 < n$, limitado porém a uma relação polinomial entre k_0 e $m(= n + k_0)$. Isto é, deve valer $k_0(n) = O(n^{1/c})$, para alguma constante $c > 1$. Nessas condições, nosso esquema é mais econômico no uso de memória e de banda em canais de comunicação.

Valor de k_0	com $m = n + k_0$	Tamanho da cifra de CL-PKE-Proposto
$k_0 > n$	$m > 2n$	maior que a dos antecessores
$k_0 = n$	$m = 2n$	igual a dos antecessores
$k_0 < n$	$m < 2n$	menor que a dos antecessores

Tabela 7.4: Tamanho relativo do texto cifrado em CL-PKE-Proposto

7.2.4 Modelo de Segurança

Os esquemas estudados de CL-PKE alcançam um nível de segurança que pode ser comprovado. Cada um, entretanto, usa uma hipótese de problema difícil, nas demonstrações de segurança contra adversários Tipo-II. Na tabela 7.5, são relacionados os problemas considerados difíceis computacionalmente. Sob a hipótese da dificuldade desses problemas é que foram possíveis as demonstrações de segurança.

Conforme discutido na seção 3.5.3, é sabido que BDH é reduzido para CDH, e que GDH resolve CDH se existir um oráculo DDH. Desse modo, as reduções adotadas na demonstração de segurança para CL-PKE-Proposto usam uma hipótese mais restritiva que seus antecessores, no caso de ataques Tipo-II (o que configura uma desvantagem para nossa proposta). Entretanto, cabe lembrar que o modelo de adversários de (CHENG; COMLEY, 2005) é mais fraco

que o do nosso CL-PKE-Proposto, pois resolvemos adotar o mesmo modelo de (AL-RIYAMI; PATERSON, 2003) e (AL-RIYAMI; PATERSON, 2005) (veja seção 4.4, para mais detalhes).

Esquema CL-PKE	Problemas Reduzidos aos Adversários	
	Adversário Tipo-I	Adversário Tipo-II
(AL-RIYAMI; PATERSON, 2003)	BDH	CDH
(AL-RIYAMI; PATERSON, 2005)	BDH	CDH
(CHENG; COMLEY, 2005)	BDH	GDH
CL-PKE-Proposto	BDH	BDH

Tabela 7.5: Problemas Pressupostos Difíceis nas Demonstrações de Segurança

Acreditamos ser possível o desenvolvimento de outra seqüência de reduções para a demonstração de segurança para adversários Tipo-II, de modo que CDH (ou GDH) seja reduzido ao problema do KGC conseguir decriptografar textos em CL-PKE-Proposto, sem conhecimento da informações secretas de seus usuários.

Em (ZHANG; FENG, 2005) e em (LIBERT; QUISQUATER, 2006) é apresentada uma forma de ataque aos esquemas de (AL-RIYAMI; PATERSON, 2003) e (AL-RIYAMI; PATERSON, 2005) e respectivas correções são sugeridas. Nosso CL-PKE-Proposto não sofre o mesmo tipo de ataque, bem como o de (CHENG; COMLEY, 2005), pois ambos já embutem a sugestão de correção.

7.2.5 Resumo de Vantagens e Desvantagens

De acordo com o descrito nas seções imediamente anteriores a esta, é possível resumir os seguintes **pontos de vantagem** para CL-PKE-Proposto:

- Velocidade: em implementações típicas, CL-PKE-Proposto deve ser o mais veloz para criptografar textos.
- Tamanho de cifras e de chaves públicas: CL-PKE-Proposto requer menor tamanho de chaves públicas e pode produzir cifras menores.
- Menor número de funções de *hash*: há uma função de *hash* a menos, nos parâmetros públicos de CL-PKE-Proposto.

Notamos os seguintes **pontos de desvantagem** para CL-PKE-Proposto (ou pontos que poderiam ser aprimorados):

- Hipótese mais forte: para adversários Tipo-II, usamos a hipótese da dificuldade de BDH, contra a dificuldade de CDH ou GDH dos outros trabalhos.
- Decriptografia: em implementações típicas, nosso esquema deve ser o segundo mais veloz para decriptografar.

7.3 Análises sobre CL-PKS-Proposto

Seguem comentários e análises sobre nosso esquema proposto de para assinatura no modelo de criptografia sem certificado.

7.3.1 Viabilidade de Implementação

Levando-se em conta o resumo apresentado por (GALBRAITH; PATERSON; SMART, 2006), a respeito do uso de emparelhamentos bilineares em esquemas criptográficos, conclui-se que CL-PKS-Proposto e demais esquemas relacionados estudados podem ser implementados na prática:

- se as implementações para (HUANG et al., 2005) e (ZHANG et al., 2006), usarem grupos definidos sobre corpos de característica prima grande (e grau de imersão 2), dada a utilização de emparelhamentos simétricos;
- em CL-PKS-Proposto, sem restrições, já que o emparelhamento é assimétrico e não há necessidade de cálculo de *hash* com imagem em $\mathbb{G}_2$.

O uso de emparelhamento assimétrico por CL-PKS-Proposto torna-o mais flexível que seus antecessores, pois as implementações não ficam restritas ao universo de curvas supersingulares. Por ter sido possível a demonstração de segurança para CL-PKS-Proposto, requerindo apenas um homomorfismo (não necessariamente inversível) de G_2^* em G_1^*, há ainda a possibilidade de se implementar técnicas de compactação de valores de emparelhamentos (SCOTT; BARRETO, 2004)[1]. Essas técnicas, além de reduzirem o tamanho em bits para armazenar valores do emparelhamento, permitem maior eficiência no cálculo de exponenciações sobre esses pontos.

7.3.2 Eficiência Computacional

Nas tabelas 7.6 e 7.7, são listados os esquemas de assinatura propostos desde a introdução do modelo CL-PKS, em (AL-RIYAMI; PATERSON, 2003). O esquema de assinatura anunciado

[1] Do original inglês, *compressed pairing*.

neste último artigo não foi demonstrado seguro e, de fato, não o era: em (HUANG et al., 2005) foram exibidas uma forma de ataque e uma nova proposta que corrigia o problema original. Mantivemos a citação do esquema de assinatura de (AL-RIYAMI; PATERSON, 2003) nas tabelas, apenas com o propósito de evidenciar melhorias que surgiram a cada trabalho.

Considerando-se que as operações mais caras envolvidas nos esquemas são respectivamente as listadas na tabela 7.1, podemos obter uma comparação de tempo computacional para assinar mensagens e verificar assinaturas, com cada um dos esquemas CL-PKS.

A tabela 7.6 relaciona, em ordem cronológica de publicação, os esquemas relacionados com CL-PKS-Proposto e totaliza as operações que envolvem pontos da curva. Por essa tabela, é possível observar em nossa proposta:

Velocidade na assinatura e na verificação. Em implementações típicas, CL-PKS-Proposto deve ser o mais veloz tanto para assinar mensagens quanto para verificar assinaturas. Na assinatura, CL-PKS-Proposto é certamente mais veloz que o esquema de de (HUANG et al., 2005), pois não requer emparelhamento; comparativamente ao esquema de (ZHANG et al., 2006), nosso esquema deve ser mais rápido para assinar, pois substitui duas multiplicações escalares em $\mathbb{G}_1$ por uma exponenciação em G_T. Na verificação de assinaturas, CL-PKS-Proposto economiza três operações de cálculo de emparelhamento.

Esquema CL-PKS	Assinatura						Verificação					
	E	X	M	m	S	H	E	X	M	m	S	H
(AL-RIYAMI; PATERSON, 2003)	1	0	3	0	1	1	4	1	0	1	0	1
(HUANG et al., 2005)	2	0	2	0	1	1	4	1	0	1	0	2
(ZHANG et al., 2006)	0	0	3	0	2	2	4	0	0	2	0	3
CL-PKS-Proposto	0	1	1	0	0	1	1	1	1	1	1	2

Tabela 7.6: Quantidade de operações sobre grupos nos esquemas CL-PKS

7.3.3 Uso de Espaço

A tabela 7.7 lista o espaço de mensagens assinadas, permitindo a comparação entre tamanhos de assinaturas. CL-PKS-Proposto gera assinaturas de menor tamanho que o esquema de (ZHANG et al., 2006), pois, conforme se pode observar nessa tabela, apenas um inteiro de $\mathbb{Z}_p^*$ é usado para representar o segundo componente da assinatura. Dependendo do valor de n, isto é, do comprimento da mensagem assinada, relativamente ao parâmetro de segurança k, nosso

esquema também produz assinaturas menores que as geradas pelo esquema de (HUANG et al., 2005).

Esquema CL-PKS	Espaço de Assinatura	Espaço de Chave Pública
(AL-RIYAMI; PATERSON, 2003)	$\mathbb{G}_1 \times \{0,1\}^n$	$\mathbb{G}_1$
(HUANG et al., 2005)	$\mathbb{G}_1 \times \{0,1\}^n$	$\mathbb{G}_1$
(ZHANG et al., 2006)	$\mathbb{G}_1 \times \mathbb{G}_1$	$\mathbb{G}_1$
CL-PKS-Proposto	$G_1 \times \mathbb{Z}_p^*$	G_T

Tabela 7.7: Espaços de assinatura e de chave pública dos esquemas CL-PKS

Por meio da tabela 7.7, na coluna que relaciona o espaço de chaves públicas de cada esquema de assinatura, percebe-se que CL-PKS-Proposto usa chaves em G_T, que possuem representações mais longas que as de pontos em $\mathbb{G}_1$ (de curvas supersingulares). Além disso, curvas ordinárias ($G_1 \neq G_2$) exigem mais espaço para G_2.

Entretanto, como já dissemos, é possível a aplicação de compactação de valores de emparelhamentos usando a técnica descrita em (SCOTT; BARRETO, 2004). É importante notar que os esquemas de (HUANG et al., 2005) e de (ZHANG et al., 2006) são ambos definidos sobre emparelhamentos simétricos. Se os convertermos para esquemas com emparelhamentos assimétricos (como o é nosso CL-PKS-Proposto), eventualmente as chaves públicas terão que ser definidas em G_2. Se isso ocorrer, nosso CL-PKS-Proposto passa a ter chaves públicas de igual tamanho ou possivelmente até menor: se CL-PKS-Proposto for definido sobre curvas MNT (MIYAJI; NAKABAYASHI; TAKANO, 2001) com técnicas de compactação, produzirá chaves públicas de igual tamanho aos dos outros dois esquemas; se for definido sobre curvas BN (BARRETO; NAEHRIG, 2005), pode eventualmente ter chaves mais curtas. A investigação dessas possibilidades é sugerida como trabalho futuro, na seção 8.2.3.

7.3.4 Modelo de Segurança

Alguns dos esquemas estudados de CL-PKS alcançam um nível de segurança que pode ser comprovado. Nas demonstrações de segurança, entretanto, adotamos hipóteses diferentes dos demais trabalhos. Na tabela 7.8, são relacionados os problemas considerados difíceis computacionalmente. No caso de ataque Tipo-I, a redução adotada na demonstração de segurança para CL-PKS-Proposto usa hipótese mais restritiva que os demais (e isso é uma desvantagem). No entanto, no caso de ataque Tipo-II, nossa proposta usa uma hipótese melhor: a de que BPI é intratável. É uma hipótese melhor que a do CDH, pois CDH é reduzido ao problema BPI, que,

por sua vez, é reduzido ao problema do logaritmo discreto (para detalhes, veja seção 3.5.3).

Esquema CL-PKS	Problemas Reduzidos aos Adversários	
	Adversário Tipo-I	Adversário Tipo-II
(HUANG et al., 2005)	CDH	CDH
(ZHANG et al., 2006)	CDH	CDH
CL-PKS-Proposto	q-SDH	BPI

Tabela 7.8: Problemas Pressupostos Difíceis nas Demonstrações para CL-PKS

Com relação ao modelo de segurança para CL-PKS, valem alguns comentários adicionais. Adotamos as mesmas definições de adversários propostas por (ZHANG et al., 2006). Aqui, o adversário Tipo-I pode substituir a chave pública, sem ser inquirido sobre o valor secreto correspondente, o que ocorre no adversário Tipo-I de (HUANG et al., 2005). Logo, o modelo que adotamos para adversário Tipo-I é melhor que o deste último.

Cabe observar, entretanto, que nós impusemos que o valor secreto correspondente a uma chave pública substituída fosse requerido do adversário sempre que ele solicitasse uma assinatura sobre uma identidade com chave substituída. Esta é uma restrição ao modelo de adversários proposto por (AL-RIYAMI; PATERSON, 2003), mas lembramos que esses mesmos autores não encontraram a demonstração sob o modelo mais forte.

7.3.5 Resumo de Vantagens e Desvantagens

De acordo com o descrito nas seções anteriores, é possível resumir os seguintes **pontos de vantagem** para CL-PKS-Proposto:

- Velocidade na assinatura e na verificação: em implementações típicas, CL-PKS-Proposto deve ser o mais veloz para assinar mensagens e para verificar assinaturas.
- Assinaturas mais curtas: gera assinaturas de menor tamanho.
- Maior flexibilidade: devido ao uso de emparelhamento assimétrico.
- Hipótese menos forte: para adversários Tipo-II, usamos a hipótese da dificuldade de BPI, contra a dificuldade de CDH dos outros trabalhos.

Notamos os seguintes **pontos de desvantagem** para CL-PKS-Proposto (ou pontos que poderiam ser aprimorados):

- Hipótese mais forte: para adversários Tipo-I, usamos a hipótese da dificuldade de q-SDH, contra a dificuldade de CDH dos outros trabalhos.
- Chaves públicas maiores: nossa proposta requer chaves públicas definidas em G_T, contra chaves em $\mathbb{G}_1$ nos demais esquemas.

7.4 Resumo

Neste capítulo foram analisadas as propostas CL-PKE-Proposto e CL-PKS-Proposto, com relação à viabilidade de implementação, eficiência computacional, tamanhos de chave e de cifra/assinatura, e modelo de segurança. Para cada esquema proposto, foram relacionadas vantagens e desvantagens.

Capítulo 8

Conclusões

Neste capítulo apresentamos um resumo das contribuições desta dissertação e enumeramos sugestões de trabalhos futuros.

8.1 Resumo de Contribuições

O trabalho apresentado nesta dissertação gerou a publicação e submissão dos artigos abaixo:

An Improved Certificateless Public Key Encryption, (TERADA; GOYA, 2006a), nos anais de *"The 2006 Symposium on Cryptography and Information Security"* (SCIS 2006), p.17-20, Hiroshima, Japão, Janeiro de 2006. Também submetido para *International Journal of Security on Networks*.

A Certificateless Signature Scheme based on Bilinear Pairing Functions, (TERADA; GOYA, 2006b). Submetido para *"1st International Workshop on Security"* (IWSEC2006), Kyoto, Japão, Outubro de 2006.

Os artigos acima sintetizam respectivamente as contribuições a seguir:

1. Novo protocolo de criptografia, sob o modelo CL-PKC, que:
   - foi demonstrado seguro, sob uma noção de segurança;
   - apresenta melhorias em eficiência computacional e na utilização de memória ou banda, comparativamente a esquemas anteriores;

- é uma opção para IBE, quando não é desejável a característica de custódia de chaves (*key escrow*).

2. Novo protocolo de assinatura, sob o modelo CL-PKC, que:
   - foi demonstrado seguro, sob uma noção de segurança;
   - apresenta melhorias na utilização de memória ou banda
   - apresenta maior eficiência computacional na assinatura e na verificação;
   - mais flexível para escolha de grupos, quando comparado a esquemas anteriores;
   - é uma opção para IBS, quando não é desejável a característica de custódia de chaves (*key escrow*).

8.2 Trabalhos Futuros

Enumeramos, nas próximas subseções, sugestões ou tópicos a serem estudados em trabalhos futuros.

8.2.1 CL-PKE-Proposto2

No apêndice B, é apresentada uma segunda proposta de esquema de criptografia sob o modelo CL-PKC, que nomeamos CL-PKE-Proposto2. Acreditamos ser possível uma demonstração de segurança para a proposta, pois ela é derivada de um outro trabalho, demonstrado seguro: (SHI; LI, 2005).

Se realmente satisfizer uma noção de segurança, esse segundo esquema CL-PKE é mais eficiente que nossa primeira proposta, CL-PKE-Proposto. Isso é devido ao fato de que não são necessários cálculos de emparelhamento na criptografia, pois foi adotada técnica sugerida por (SAKAI; KASAHARA, 2003).

Paralelamente ao desenvolvimento de nossa proposta, foi apresentado o trabalho de (LIBERT; QUISQUATER, 2006), com nova sugestão de um CL-PKE, comparável, em vários aspectos, com CL-PKE-Proposto2. A segurança desse esquema foi anunciada, porém suas demonstrações de segurança não foram publicadas.

Sugerimos, como trabalho futuro, o desenvolvimento de CL-PKE-Proposto2, eventualmente melhorando características de modo a acrescentar alguma vantagem sobre o trabalho de (LIBERT; QUISQUATER, 2006).

8.2.2 Modelos Teóricos

As demonstrações de todos os trabalhos relacionados e de nossas propostas foram baseadas no modelo do oráculo aleatório, discutido na seção 3.8. Um caminho alternativo seria buscar formas de demonstrar a segurança desses esquemas pelo modelo padrão, isto é, sem a hipótese de existência de oráculos aleatórios. Existem muitas referências de trabalhos que seguem essa linha, dentre as quais podemos citar (WATERS, 2004), (BONEH; BOYEN, 2004a), (BONEH; BOYEN, 2004b) e (BONEH; BOYEN, 2004c). Essa é, portanto, uma sugestão de trabalho futuro.

O fato de nossas propostas requererem menor quantidade de funções de *hash* implicará maior vantagem em eficiência computacional se todos os esquemas relacionados também forem convertidos para o modelo sem oráculos aleatórios, com técnicas como as apresentadas em (BONEH; BOYEN, 2004a). Isso é verdade, uma vez que nesse trabalho os oráculos aleatórios são substituídos por instâncias de funções de *hash* que envolvem operações de exponenciação.

Numa seqüência de três trabalhos, (YUM; LEE, 2004a), (YUM; LEE, 2004b), (YUM; LEE, 2004c), os autores descreveram composições genéricas, capazes de gerar esquemas CL-PKC a partir de esquemas ID-PKC. Em (LIBERT; QUISQUATER, 2006), são criticadas algumas das técnicas adotadas, e são expostos argumentos que acabam por invalidar parte do trabalho de Yum e Lee. Em (LIBERT; QUISQUATER, 2006) é proposta uma nova construção genérica e segura para CL-PKE a partir de um esquema IBE seguro. Nada é feito para assinatura e, portanto, sugerimos revisão da construção genérica de um CL-PKS a partir de um IBS, apresentada em (YUM; LEE, 2004b).

Em (KANG; PARK, 2005), é descrita uma falha na demonstração apresentada em (AL-RIYAMI; PATERSON, 2005), invalidando a criação genérica de esquemas CBE, a partir de CL-PKE. Os autores argumentam não ser possível relacionar esses dois tipos de esquemas, e ficou em aberta a questão: é possível construir um novo CBE a partir de um CL-PKE? Entretanto, anteriormente em (YUM; LEE, 2004c), foram demonstradas equivalências, em várias direções, entre sistemas IBE, CL-PKE e CBE. Sugerimos revisão desse trabalho, eventualmente para detectar e corrigir falhas, mostrar uma nova construção segura e genérica, ou, ao contrário, demonstrar que de fato não existem equivalências entre os três modelos.

8.2.3 Tópicos Diversos

Acrescentamos, ainda, as seguintes sugestões de trabalhos futuros:

Emparelhamentos assimétricos. Tanto o nosso CL-PKE-Proposto quanto o CL-PKE de (LIBERT; QUISQUATER, 2006) foram descritos inicialmente com emparelhamentos

simétricos. Estudar a possibilidade de generalizá-los para uso de emparelhamentos assimétricos e apresentar respectivas demonstrações de segurança.

Tentar converter também os esquemas de (HUANG et al., 2005) e de (ZHANG et al., 2006) para emparelhamentos assimétricos. Se essa conversão só for possível de modo que as chaves públicas tenham que ser definidas em G_2 (em vez de G_1), nosso CL-PKS-Proposto passa a ter chaves públicas de igual tamanho se forem utilizadas curvas MNT e compactação de valores de emparelhamentos. Investigar também se, nesse cenário, o uso de curvas BN leva a pontos de G_T mais compactos que pontos de G_2, o que tornaria nosso CL-PKS-Proposto mais eficiente também por produzir chaves públicas menores.

CL-PKE e CL-PKS sem emparelhamentos. Em (BAEK; SAFAVI-NAINI; SUSILO, 2005) foi apresentado o primeiro esquema de criptografia sob o modelo CL-PKC, sem uso de emparelhamentos. Dar continuidade ao trabalho, aprimorando-o ou o estendendo para assinatura.

Criptografia autenticada. Todos os esquemas CL-PKE podem ser derivados para um esquema de criptografia com autenticação de origem, ou seja, CL-Auth-PKE. Trabalhos desse gênero são exemplificados com (LEE; LEE, 2004) e (CHENG; COMLEY, 2005). Sugerimos estender nosso CL-PKE-Proposto2, ou o PKE de (LIBERT; QUISQUATER, 2006), para um CL-Auth-PKE.

Criptoassinatura sem *key escrow*. Inúmeros esquemas já foram propostos em criptoassinatura, isto é, criptografia e assinatura em único passo, ou *signcryption*. Uma sugestão de trabalho futuro é tomar como ponto de partida nosso CL-PKS-Proposto e o esquema de criptoassinatura em (BARRETO et al., 2005), para gerar um novo esquema de criptoassinatura, sem custódia de chaves.

CL-KEM. Em criptografia de chave pública, quando se tem necessidade de criptografar mensagens longas, é comum usar criptografia assimétrica apenas para trocar uma chave secreta, para então usar criptografia simétrica, mais eficiente, para cifrar a mensagem longa. À primeira parte desse procedimento se dá o nome de *Key Encapsulation Mechanism* (KEM), ou mecanismo de encapsulamento de chave.

Em (BENTAHAR et al., 2005), é apresentada uma construção genérica de mecanismos CL-KEM a partir de um ID-KEM (isto é, KEM baseado em identidades). Em (LIBERT; QUISQUATER, 2006), são enumeradas algumas restrições que tornam a combinação de dois esquemas seguros, num terceiro esquema inseguro. Uma primeira sugestão de trabalho futuro, nesse tópico, é analisar se na construção de (BENTAHAR et al., 2005) não há algum equívoco, como os detectados por (LIBERT; QUISQUATER, 2006), em (YUM; LEE, 2004a) e (AL-RIYAMI; PATERSON, 2005).

Também em (CHENG; COMLEY, 2005), foi proposta uma instância de CL-KEM, sem as respectivas demonstrações de segurança. Uma segunda sugestão de trabalho futuro, em KEM, é apresentar um CL-KEM e demonstrá-lo seguro.

Assinatura com verificador designado. Uma extensão de assinatura é aquela em que a verificação só pode ser efetuada por identidades previamente autorizadas (ou designadas). Em (LIBERT; QUISQUATER, 2004) há um exemplo de IBS com verificador designado. E em (HUANG et al., 2006), o equivalente para CL-PKS. Acreditamos ser possível evoluir esses dois trabalhos.

Segurança mediada. Em (CHOW; BOYD; NIETO, 2006), é proposto um modelo de criptografia de segurança mediada sobre CL-PKC, que acrescenta a propriedade de revogação instantânea de chaves. Um esquema CL-PKE com segurança mediada é apresentado, porém faltou definir uma noção de segurança e adversários, sobre a qual o esquema pudesse ser demonstrado seguro. Também ficam em abertas quaisquer novas propostas de esquemas, como CL-PKS com segurança mediada e outros.

Modelos hierárquicos. Estender o CL-PKC para modelos hierárquicos, com vários KGC, como proposto em (AL-RIYAMI; PATERSON, 2003) e (AL-RIYAMI, 2005). Estudar escalabilidade desses modelos.

Implementações e *benchmarking*. Fixar alguns parâmetros e curvas, implementar os esquemas estudados e propostos de CL-PKE e CL-PKS. Tabular resultados.

8.3 Resumo

Neste capítulo foram enumerados as contribuições desta dissertação e sugestões de trabalhos que podem ser desenvolvidos, em frentes diversificadas:

- linha teórica;
- projeto de protocolos ou modelos novos;
- implementação.

Apêndice A

Trabalhos Relacionados aos Esquemas Propostos

Neste apêndice são descritos os trabalhos e conceitos que culminaram na definição de CL-PKC e que influenciaram a especificação de CL-PKC-Proposto e CL-PKS-Proposto. A apresentação é feita em ordem cronológica de aparição dos trabalhos e, na medida do possível, foram incluídos comentários sobre o que motivou cada um. Ao final deste apêndice, são apresentadas tabelas de comparação das formações dos esquemas.

A.1 Chaves Públicas Auto-Certificadas

A noção de criptografia de chave pública sem certificado, com a nomenclatura adotada nesta dissertação, foi apresentada originalmente em (AL-RIYAMI; PATERSON, 2003). Os autores se inspiraram no conceito de criptografia de chave pública *auto-certificada*[1] de (GIRAULT, 1991), em que cada chave de criptografia é calculada conjuntamente pelo respectivo usuário e por uma autoridade de confiança, de modo que a chave pública embute uma espécie de certificação de sua legitimidade. Trata-se de uma ligeira modificação sobre esquemas convencionais de criptografia de chave pública, conforme descrito a seguir.

Um usuário cria um par de chaves para si: uma chave secreta x e uma chave pública y. A chave pública y é entregue à autoridade de confiança, que vincula essa chave ao seu respectivo dono, por meio de um valor público chamado *testemunho*. O cálculo de um testemunho w é realizado a partir da combinação de y com a identidade ID do usuário. O valor w funciona

[1] Do original inglês, *self-certified*.

como a assinatura da autoridade de confiança, certificando que y é a chave pública do usuário identificado por ID. As relações matemáticas entre os valores são tais que:

- dados w, ID e a chave pública da autoridade de confiança, qualquer pessoa pode extrair a chave pública y para criptografar um texto endereçado a ID;
- somente a autoridade de confiança é capaz de produzir um testemunho w, a partir de y e ID.

Desse modo, o testemunho w tem o mesmo papel que certificados em sistemas convencionais de criptografia de chave pública, porém é mais *leve*, isto é, agrega menos informação que um certificado tradicional e segue embutido num valor publicamente conhecido, do qual se obtém a chave de criptografia e a respectiva garantia de que a chave é a verdadeira para ID.

Na ocasião da apresentação de seu trabalho, Girault mostrou instâncias de esquemas auto-certificados baseados nos problemas do logaritmo discreto e da fatoração de inteiros. E afirmou que o nível de segurança atingido por tais esquemas era equivalente ao obtido por sistemas convencionais de criptografia de chave pública, com a vantagem de otimizarem espaço necessário para armazenamento, quantidade de comunicações entre usuários e autoridade, e tempo de processamento nos protocolos.

Em (SAEEDNIA, 2003), entretanto, uma falha foi identificada no modelo de Girault. Autoridades de confiança não muito confiáveis, poderiam escolher parâmetros públicos de fácil fatoração ou sujeitos à aplicação do algoritmo de Pohlig-Hellman, para obtenção do logaritmo discreto. Assim, as autoridades teriam a habilidade de calcular as chaves secretas de seus usuários e decriptografar mensagens. Um tal comportamento desonesto não seria detectado pelos usuários. Assim, o nível de segurança do modelo de Girault se iguala ao do modelo de criptografia de chave pública baseado em identidade, com custódia de chaves. Saeednia discute como a falha pode ser corrigida, porém o custo da correção é a perda das vantagens da auto-certificação, isto é, das otimizações em armazenamento, comunicação e processamento.

A idéia de gerar uma chave pública misturando o identificador de usuário com a chave secreta da autoridade de confiança e com a chave secreta do usário foi retomada no trabalho de (AL-RIYAMI; PATERSON, 2003), aproveitando a concretização de (BONEH; FRANKLIN, 2001) para criptografia e decriptografia de chave pública baseada em identidades. Vamos, então, resumir o esquema de Boneh-Franklin, apresentar o de Al-Riyami e Paterson e os que os sucederam.

A.2 IBE de Boneh-Franklin

O esquema de criptografia e decriptografia no modelo de criptografia de chave pública baseada em identidades de (BONEH; FRANKLIN, 2001) usa emparelhamentos bilineares para viabilizar segurança contra ataques de texto cifrado escolhido. O uso de emparelhamentos em sistemas de criptografia foi exposto originalmente por (SAKAI; OHGISHI; KASAHARA, 2000).

O esquema completo de Boneh-Franklin é denominado FullIdent. É composto de quatro algoritmos, nos quais o emparelhamento bilinear admissível é o pilar de sustentação do esquema:

inicializa. [2] Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema $\text{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3, H_4 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{G}_1^*$

$H_2 : \mathbb{G}_2 \to \{0,1\}^n$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^n \to \{0,1\}^n$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n \times \{0,1\}^n$.

extrai. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular $Q_A = H_1(ID_A)$ e a chave secreta $d_A = sQ_A$.

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A e params, calcular o texto cifrado $\langle rP, \sigma \oplus H_2(\hat{e}(Q_A, P_{pub})^r), m \oplus H_4(\sigma) \rangle$, onde $r = H_3(m, \sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V, W \rangle \in \mathcal{C}$, a chave secreta d_A e params:

Se $U \notin \mathbb{G}_1^*$, C é rejeitado. Caso contrário, calcular $V \oplus H_2(\hat{e}(d_A, U)) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r = H_3(\sigma, m)$, verificar se $U = rP$. Se for, m é a resposta, senão C é rejeitado.

Os algoritmos **inicializa** e **extrai** são executados pelo gerador de chaves secretas (PKG, ou *Private Key Generator*). Desse modo, PKG conhece as chaves secretas de todos usuários e tem o poder de decifrar qualquer texto. A perda ou o comprometimento da chave-mestra do PKG seria desastrosa para todo o sistema, por isso, um IBE que simplesmente implementa os algoritmos acima descritos é indicado apenas para ambientes de menor porte (restrito a uma empresa, por exemplo).

[2] Do original inglês, *setup*.

Para chegar ao IBE seguro contra ataques de texto cifrado escolhido, Boneh e Franklin se utilizaram da transformação de (FUJISAKI; OKAMOTO, 1999), capaz de converter um esquema de criptografia de chave pública mais fraco (que satisfaz apenas a noção IND-CPA), num novo esquema fortalecido IND-CCA. Boneh e Franklin construíram um esquema básico que foi nomeado BasicPub. Esse esquema foi demonstrado seguro ao nível IND-CPA. Ao aplicar a transformação de Fujisaki-Okamoto, foi obtido o esquema FullIdent descrito acima. A diferença entre o esquema básico e o completo está fundamentalmente no componente W da cifra e no cálculo de r, gerados com o auxílio das duas funções de hash: H_3 e H_4.

Cabe observar que o cálculo do emparelhamento (o de mais alto custo computacional em todo o esquema) é feito sobre valores que independem da mensagem a ser cifrada: $\hat{e}(Q_A, P_{pub})$. Assim, pode ser vantajoso armazenar, para cada ID, o respectivo cálculo do emparelhamento.

A.3 CL-PKE de Al-Riyami e Paterson

O esquema de criptografia e decriptografia no modelo de criptografia de chave pública sem certificado de (AL-RIYAMI; PATERSON, 2003), que possui segurança contra ataques de texto cifrado escolhido, é denominado FullCL-PKE. É composto de sete algoritmos, nos quais o emparelhamento bilinear admissível também é a base do esquema, pois se trata de uma extensão do IBE de (BONEH; FRANKLIN, 2001):

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3, H_4 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{G}_1^*$

$H_2 : \mathbb{G}_2 \to \{0,1\}^n$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^n \to \{0,1\}^n$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n \times \{0,1\}^n$.

secreta-parcial. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular $Q_A = H_1(ID_A)$ e a chave secreta parcial $D_A = sQ_A$.

info-secreta. Selecionar ao acaso uma informação secreta $x_A \in \mathbb{Z}_q^*$, que é a informação secreta da entidade A.

secreta. Dados params, a chave secreta parcial D_A e a informação secreta x_A, a chave secreta (completa) é $S_A = x_A D_A = x_A s Q_A$.

publica. Dados params e a informação secreta x_A, a chave pública é $P_A = \langle X_A, Y_A \rangle$, onde $X_A = x_A P$ e $Y_A = x_A P_{pub} = x_A s P$.

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A, a chave pública $P_A = \langle X_A, Y_A \rangle$ e params, verificar a validade da chave pública: se $\hat{e}(X_A, P_{pub}) = \hat{e}(Y_A, P)$, a chave é válida; caso contrário, abortar a criptografia.

Calcular o texto cifrado $\langle rP, \sigma \oplus H_2(\hat{e}(Q_A, Y_A)^r), m \oplus H_4(\sigma) \rangle$, onde $r = H_3(m, \sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V, W \rangle \in \mathcal{C}$, a chave secreta S_A e params:

Calcular $V \oplus H_2(\hat{e}(S_A, U)) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r = H_3(\sigma, m)$, verificar se $U = rP$. Se for, m é a resposta, senão C é rejeitado.

Os algoritmos **inicializa** e **secreta-parcial** são executados pelo centro de geração de chaves (KGC, ou *Key Generating Centre*). Tipicamente, os algoritmos **info-secreta**, **secreta** e **publica** são executados pela entidade A. Em CL-PKE-Proposto, estes três últimos algoritmos foram aglutinados em um único, por simplicidade. Al-Riyami e Paterson separaram em três módulos para tornar evidente a possibilidade de se divulgar uma chave pública e criptografar mensagens para um usuário A, sem que a chave secreta completa de A tenha sido gerada. Tal propriedade possibilita a implementação de *workflows criptográficos*, seqüências de operações de criptografia que envolvem diferentes usuários.

Note que, no esquema de (AL-RIYAMI; PATERSON, 2003), a chave pública é composta por dois pontos de $\mathbb{G}_1$. O primeiro ponto, $x_A P$, depende de uma informação que só o usuário A conhece; o segundo ponto, $x_A P_{pub} = x_A s P$, depende do segredo de KGC. Desse modo, a chave pública embute uma certificação implícita de sua legitimidade. Ademais, a chave de criptografia é, na verdade, composta não apenas pela chave pública P_A, mas também pelo identificador do usuário receptor (ID_A). Como resultado, o esquema dispensa certificados.

Posteriormente, em (AL-RIYAMI; PATERSON, 2005) melhorias foram acrescentadas à proposta original, de modo a diminuir o tamanho das chaves públicas e tornar a criptografia mais eficiente. O novo equema é denominado FullCL-PKE* e as modificações são resumidas a seguir:

inicializa. Uma nova função de hash é adicionada: $H_5 : \mathbb{G}_1 \rightarrow \{0,1\}^n$.

secreta-parcial e info-secreta. Algoritmos idênticos aos da versão anterior.

secreta. Agora, $S_A = \langle D_A, x_A \rangle$.

publica. Agora, $P_A = x_A P$.

cript. Para verificar a validade da chave pública, agora basta conferir se $P_A \in \mathbb{G}_1^*$, e o texto cifrado é $\langle rP, \sigma \oplus H_2(\hat{e}(Q_A, P_{pub})^r) \oplus H_5(rP_A), m \oplus H_4(\sigma) \rangle$.

decript. Dados $C = \langle U, V, W \rangle \in \mathcal{C}$, a chave secreta $S_A = \langle D_A, x_A \rangle$ e params:

Calcular $V \oplus H_2(\hat{e}(D_A, U)) \oplus H_5(x_A U) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r = H_3(\sigma, m)$, verificar se $U = rP$. Se for, m é a resposta, senão C é rejeitado.

No novo esquema, a criptografia é mais eficiente, pois a validação da chave pública, que antes requeria o cálculo de dois emparelhamentos, agora é substituída pela verificação de pertinência de um ponto (chave pública) ao grupo $\mathbb{G}_1$. O comprimento da chave pública foi reduzido à metade (de dois pontos do grupo para um). Para viabilizar ambas melhorias, os autores tiveram a necessidade de acrescentar uma nova função de hash envolvendo $\mathbb{G}_1$.

A.4 CBE de Gentry

Paralelamente ao trabalho de (AL-RIYAMI; PATERSON, 2003), em (GENTRY, 2003) foi desenvolvida independentemente uma proposta similar. Gentry nomeou seu modelo de criptografia de chave pública baseada em certificado (CBE, de *Certificate-Based Encryption*). O modelo mescla um esquema convencional de criptografia de chave pública com o IBE de (BONEH; FRANKLIN, 2001), de forma que a chave de decriptografia agrega características de certificados de chave pública, com a identificação do usuário receptor, a chave-mestra da autoridade certificadora e um identificador temporal, que define um prazo de validade para a chave. O objetivo do esquema de Gentry é a simplificação do processo de revogação de chaves públicas (de PKCs convencionais). Como resultado, é obtido um esquema intermediário entre IBE e o modelo de criptografia de chave pública tradicional, sem o problema de custódia de chaves do IBE.

O esquema que possui segurança contra ataques de texto cifrado escolhido é denominado FullCBE. É composto de cinco algoritmos, fundamentados num emparelhamento bilinear admissível:

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}/q\mathbb{Z}$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3, H_4 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{G}_1^*$

$H_2 : \mathbb{G}_2 \to \{0,1\}^n$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^n \to \{0,1\}^n$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n \times \{0,1\}^n$.

A entidade A escolhe aleatoriamente sua chave secreta $t_A \in \mathbb{Z}_q^*$ e calcula sua chave pública $N_A = t_A P$, como em um PKC convencional.

certificacao. A identificação do usuário A é A_{info}, que combina um identificador $ID_A \in \{0,1\}^*$, com N_A. A autoridade calcula $P_A = H_1(P_{pub}, i, A_{info})$, para um período i. O certificado $Cert_A = sP_A$ é enviado à entidade A, que calcula sua chave de decriptografia (válida para o período i): $S_A = Cert_A + t_A P'_A$, onde $P'_A = H_1(A_{info})$.

cript. Dados um texto $m \in \mathcal{M}$, um identificador A_{info}, um período de tempo i e params:

Calcular o texto cifrado $\langle rP, \sigma \oplus H_2\left(\left(\hat{e}(P_{pub}, P_A)\hat{e}(N_A, P'_A)\right)^r\right), m \oplus H_4(\sigma)\rangle$, onde $r = H_3(m, \sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente, e P_A e P'_A são calculados como em **certificacao**.

decript. Dados $C = \langle U, V, W\rangle \in \mathcal{C}$, a chave secreta S_A e params:

Calcular $V \oplus H_2(\hat{e}(S_A, U)) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r = H_3(\sigma, m)$, verificar se $U = rP$. Se for, m é a resposta, senão C é rejeitado.

A autoridade certificadora deve emitir e transmitir regularmente um certificado novo para cada entidade A, de modo que esta última possa gerar sua chave de decriptografia. A comunicação entre A e a autoridade certificadora não precisa ser por canal seguro, como ocorre em IBE de (BONEH; FRANKLIN, 2001) e (AL-RIYAMI; PATERSON, 2003).

Em (AL-RIYAMI; PATERSON, 2005), foi apresentada uma conversão genérica de esquemas CL-PKE em esquemas CBE. Contudo, em (KANG; PARK, 2005) foi apresentada uma falha na demonstração de segurança sobre a conversão, tornando-a inválida.

A.5 CL-PKE de Cheng-Comley

Em (CHENG; COMLEY; VASIU, 2004) há idéias desenvolvidas praticamente em simultaneidade com o trabalho de (AL-RIYAMI; PATERSON, 2003), mas de forma independente. Naquele artigo, os autores estenderam o IBE de Boneh-Franklin, criando uma chave pública auxiliar à identidade de um usuário, que foi nomeada por *codinome*[3]. Esse codinome é idêntico à chave pública de (AL-RIYAMI; PATERSON, 2003) (composto por dois pontos da curva elíptica).

O conceito de codinome não apresentou vantagens em relação ao CL-PKC de Al-Riyami e Paterson, porém Cheng e Comley deram continuidade àquela linha de estudos e buscaram melhorar os resultados de (AL-RIYAMI; PATERSON, 2003) e (AL-RIYAMI; PATERSON, 2005). Chegaram a um CL-PKE descrito em (CHENG; COMLEY, 2005), com segurança contra ataques de texto cifrado escolhido, composto de cinco algoritmos:

[3]Do original inglês, *nickname*.

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3, H_4 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{G}_1^*$

$H_2 : \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \to \{0,1\}^n$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^n \to \{0,1\}^n$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n \times \{0,1\}^n$.

extrai. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular $Q_A = H_1(ID_A)$ e a chave secreta parcial $d_A = sQ_A$.

publica. Dado params, a entidade A escolhe aleatoriamente a informação secreta t_A e calcula a chave pública $N_A = t_A P$.

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A, a chave pública N_A, o texto cifrado é $\langle rP, \sigma \oplus H_2(rP, \hat{e}(Q_A, P_{pub})^r, rN_A), m \oplus H_4(\sigma) \rangle$, onde $r = H_3(m, \sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V, W \rangle \in \mathcal{C}$, os valores secretos d_A e t_A e params:

Calcular $V \oplus H_2(U, \hat{e}(U, d_A), t_A U) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r = H_3(\sigma, m)$, verificar se $U = rP$. Se for, m é a resposta, senão C é rejeitado.

Cheng e Comley optaram por reunir os algoritmos **info-secreta**, **secreta** e **publica**, de (AL-RIYAMI; PATERSON, 2003), em um único algoritmo (**publica**), pois são todos executados pela mesma entidade A. As mesmas otimizações alcançadas em (AL-RIYAMI; PATERSON, 2005) estão presentes em (CHENG; COMLEY, 2005): chave pública menor e menos cálculos de emparelhamentos. Entretanto, esse último trabalho aprimora a segunda versão de Al-Riyami e Paterson ao utilizar quatro funções de hash (uma a menos).

A.6 IBE de Galindo

Com o objetivo de melhorar a eficiência do IBE de Boneh-Franklin, o trabalho de (GALINDO, 2005) apresentou uma variação sobre (BONEH; FRANKLIN, 2001), capaz de produzir cifras de menor tamanho e com uma função de hash a menos. O truque foi utilizar a transformação de (FUJISAKI; OKAMOTO, 2000) sobre o esquema BasicPub de Boneh-Franklin, em vez da primeira versão de Fujisaki-Okamoto, desenvolvida anteriormente em (FUJISAKI; OKAMOTO, 1999).

A transformação mais recente de Fujisaki-Okamoto adota apenas uma função de hash para converter um esquema de criptografia de chave pública IND-CPA para o nível IND-CCA, mais seguro. Tal otimização foi possível ao se concatenar, em uma só cadeia, a mensagem e o valor aleatório da criptografia probabilística; na primeira versão, a mensagem e o valor aleatório eram cadeias de tamanhos iguais, tratadas paralelamente como elementos separados.

O esquema de criptografia de chave pública baseada em identidades de Galindo altera a descrição do espaço de mensagens e de texto cifrado, e potencialmente diminui o tamanho das cifras, quando comparado com o IBE de Boneh-Franklin. O trabalho de (GALINDO, 2005) foi a base para a nossa proposta descrita no capítulo 5. Os algoritmos do IBE de Galindo utilizam um emparelhamento mais genérico que envolve três grupos elípticos, em vez de dois. Para facilitar a comparação desse esquema com os demais, descrevemos a seguir o IBE de Galindo com emparelhamento simétrico:

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, n, P, P_{pub}, H_1, H_2, H_3 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^*$

$H_2 : \mathbb{G}_2 \rightarrow \{0,1\}^n$

$H_3 : \{0,1\}^{n-k_0} \times \{0,1\}^{k_0} \rightarrow \mathbb{Z}_q^*$

O espaço de mensagens agora é $\mathcal{M} = \{0,1\}^{n-k_0}$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$.

extrai. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular $Q_A = H_1(ID_A)$ e a chave secreta $d_A = sQ_A$.

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A e params, calcular o texto cifrado $\langle rP, (m \parallel \sigma) \oplus H_2(\hat{e}(Q_A, P_{pub})^r) \rangle$, onde $r = H_3(m, \sigma)$, para $\sigma \in \{0,1\}^{k_0}$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V \rangle \in \mathcal{C}$, a chave secreta d_A e params:

Calcular $V \oplus H_2(\hat{e}(d_A, U)) = (m \parallel \sigma)$. Desmembrar m e σ, para calcular $r = H_3(m, \sigma)$. Se $U = rP$, m é a resposta, senão C é rejeitado.

A.7 IBE de Sakai-Kasahara

Outra modificação sobre o IBE de (BONEH; FRANKLIN, 2001) foi apresentada em (SAKAI; KASAHARA, 2003). Certas propriedades dos emparalhamentos de Weil e Tate são

exploradas de modo que o cálculo do emparelhamento na criptografia é reduzido a uma constante, independente das chaves dos usuários e que pode ser adotada como parâmetro público do sistema. Desse modo, as implementações de IBE podem se livrar do cálculo do emparelhamento durante a criptografia (ou economizar memória e tempo de busca, caso os emparelhamentos fossem pré-calculados para cada ID).

As idéias lançadas por Sakai e Kasahara foram organizadas em (CHEN; CHENG, 2005), onde foi aplicada a transformação de (FUJISAKI; OKAMOTO, 1999) para se construir um IBE, denominado SK-IBE, eficiente e seguro contra ataques de texto cifrado escolhido. O emparelhamento adotado envolve três grupos de mesma ordem q: $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, com geradores P_1 e P_2, respectivamente para $\mathbb{G}_1$ e $\mathbb{G}_2$, onde $P_1 = \psi(P_2)$ para um isomorfismo ψ.

Todos os esquemas definidos anteriormente também podem usar emparelhamento que levam dois grupos distintos a um terceiro, todos de mesma ordem. Naqueles esquemas, adotamos $\mathbb{G}_1 = \mathbb{G}_2$ e ψ como sendo a identidade, por simplicidade. No esquema de Sakai-Kasahara, mantivemos a definição completa, com emparelhamento de grupos distintos, pois nesse caso, é possível obter vantagem sobre certas famílias de curvas elípticas (BONEH; BOYEN, 2004a), conforme descrito na seção A.8.

Nas demonstrações de segurança para SK-IBE, fica evidente o custo da eficiência alcançada: o IBE é reduzido a um problema mais genérico que BDH, isto é, a segurança de SK-IBE depende de uma hipótese mais forte, chamada k-BDHI (inversão do Diffie-Hellman bilinear (BONEH; BOYEN, 2004c)) e descrita a seguir:

k-BDHI: Para um inteiro k, x escolhido aleatoriamente de $\mathbb{Z}_q^*$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, $P_1 = \psi(P_2)$, dados $\langle P_1, P_2, xP_2, x^2P_2, \ldots, x^kP_2 \rangle$, é difícil calcular $\hat{e}(P_1, P_2)^{1/x}$.

Os algoritmos de SK-IBE são:

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \psi, \hat{e}, n, P_1, P_2, P_{pub}, g, H_1, H_2, H_3, H_4 \rangle$, onde $\mathbb{G}_1$, $\mathbb{G}_2$ e $\mathbb{G}_T$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ é um emparelhamento bilinear admissível, P_2 é gerador de $\mathbb{G}_2$, $P_1 = \psi(P_2)$, ψ é um isomorfismo, $g = \hat{e}(P_1, P_2)$, $P_{pub} = sP_1$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$

$H_2 : \mathbb{G}_T \to \{0,1\}^n$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^n \to \{0,1\}^n$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n \times \{0,1\}^n$.

extrai. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular a chave secreta $d_A = \frac{1}{s+H_1(ID_A)}P_2$. No caso em que $s + H_1(ID_A) = 0$, define-se $\frac{1}{0} = 0$, $0P_2 = O$ (ponto no infinito da curva elíptica, ou seja, o elemento neutro de $\mathbb{G}_2$) e $d_A = O$ (esse caso especial não é citado no original).

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A e params, calcular o texto cifrado $\langle rQ_A, \sigma \oplus H_2(g^r), m \oplus H_4(\sigma)\rangle$, onde $Q_A = H_1(ID_A)P_1 + P_{pub}$, $r = H_3(m,\sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V, W\rangle \in \mathcal{C}$, a chave secreta d_A e params: Se $U \notin \mathbb{G}_1^*$, C é rejeitado. Caso contrário, calcular $V \oplus H_2(\hat{e}(d_A, U)) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r = H_3(\sigma, m)$, verificar se $U = rP$. Se for, m é a resposta, senão C é rejeitado.

Observe que no novo esquema H_1 mapeia ID em $\mathbb{Z}_q^*$ (e não mais a um ponto da curva). O cálculo do emparelhamento no algoritmo de criptografia foi substituído pelo parâmetro público (constante) $g = \hat{e}(P_1, P_2)$.

Como acrescentamos a especificação $d_A = O$ no caso em que $s + H_1(ID_A) = 0$, faz-se necessária a demonstração de que o algoritmo de decriptografia recupera corretamente mensagens criptografadas para aquele ID particular que, combinado com a chave-mestra de PKG, obtém-se uma chave secreta igual ao ponto no infinito.

Suponha que $s + H_1(ID_A) = 0$ e que $d_A = O$ para uma certa entidade A identificada por ID_A. Precisamos mostrar que, para um texto cifrado $C = \langle U, V, W\rangle$, $\hat{e}(U, d_A) = \hat{e}(P_1, P_2)^r$. Como definimos $\frac{1}{0} = 0$ e $0P_2 = O$, temos que $P_2 = 0O$. Dado que $\hat{e}$ é bilinear, segue:

$$\begin{aligned}
\hat{e}(U, d_A) &= \hat{e}(rQ_A, O) \\
&= \hat{e}(H_1(ID_A)P_1 + P_{pub}, O)^r \\
&= \hat{e}((H_1(ID_A) + s)P_1, O)^r \\
&= \hat{e}(P_1, (H_1(ID_A) + s)O)^r \\
&= \hat{e}(P_1, 0O)^r \\
&= \hat{e}(P_1, P_2)^r
\end{aligned}$$

A.8 IBE de Chen-Cheng

Em (CHEN; CHENG, 2005) são sugeridas modificações sobre SK-IBE que levam a um novo esquema, nomeado CC-IBE, que é mais eficiente do ponto de vista computacional. A primeira

modificação é a fixação de um grupo particular de curvas elípticas, sobre as quais é possível explorar certas propriedades matemáticas dos emparelhamentos de Tate e Weil (instâncias conhecidas de emparelhamentos bilineares admissíveis), de modo a acelerar os cálculos. Outra alteração é o uso da segunda versão da transformação de Fujisaki-Okamoto, de forma análoga à encontrada em (GALINDO, 2005).

Uma linha de pesquisa em criptografia que tem evoluído grandemente com os estudos de emparelhamentos bilineares é a de assinaturas digitais compactas. Um trabalho de referência nessa linha é o de (BONEH; LYNN; SHACHAM, 2001), ou BLS, onde a assinatura de uma mensagem se resume à coordenada x de um ponto da curva elíptica. Um progresso em torno da técnica de BLS culminou no conceito de *emparelhamento compacto*[4], apresentado em (SCOTT; BARRETO, 2004). Nesse último trabalho, foi demonstrado como valores do emparelhamento de Tate podem ser reduzidos em tamanho, quando a curva elíptica é definida sobre $GF(p)$, com $p \geq 3$; a técnica de compressão também permite que o cálculo da exponenciação do emparelhamento compacto em $\mathbb{G}_T$ seja acelerado.

Chen e Cheng propõem o uso dessa forma de compactação, como primeira modificação sobre SK-IBE. Eles ainda incluem o valor rQ_A como um componente no cálculo de H_2 por dois motivos, de acordo com os autores: um com o objetivo de obter uma redução mais justa[5] à hipótese de segurança; outro motivo foi eliminar ambigüidades inseridas pela compactação do valor do emparelhamento. Os algoritmos de CC-IBE são descritos a seguir.

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \psi, \hat{e}, n, P_1, P_2, P_{pub}, \varphi(g^r), H_1, H_2, H_3 \rangle$, onde $\mathbb{G}_1$, $\mathbb{G}_2$ e $\mathbb{G}_T$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ é um emparelhamento bilinear admissível, P_2 é gerador de $\mathbb{G}_2$, $P_1 = \psi(P_2)$, φ é o algoritmo de compactação de emparelhamento de (SCOTT; BARRETO, 2004), $\varphi(g^r) = \varphi(\hat{e}(P_1, P_2)^r)$, $P_{pub} = sP_1$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$

$H_2 : \mathbb{G}_1 \times \mathbb{F} \to \{0,1\}^{2n}$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^{2n}$.

extrai. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular a chave secreta $d_A = \frac{1}{s+H_1(ID_A)} P_2$. No caso em que $s + H_1(ID_A) = 0$, define-se $\frac{1}{0} = 0$, $0P_2 = O$ (ponto no infinito da curva elíptica, ou seja, o elemento neutro de $\mathbb{G}_2$) e $d_A = O$ (esse caso especial não é citado no original).

[4] Do original inglês, *compressed pairing*.

[5] Do original inglês, *tightened*. Uma redução polinomial mais justa (de um problema A ao problema B) intuitivamente pode ser compreendida como um fator multiplicativo mais próximo de um, relacionando os tempos de execução para resolver A e B.

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A e params, calcular o texto cifrado $\langle rQ_A, (m \parallel \sigma) \oplus H_2(rQ_A, \varphi(g^r))\rangle$, onde $Q_A = H_1(ID_A)P_1 + P_{pub}$, $r = H_3(m, \sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V\rangle \in \mathcal{C}$, a chave secreta d_A e params, calcular $\varphi(g') = \varphi(\hat{e}(U, d_A))$ e $m' \parallel \sigma' = V \oplus H_2(U, \varphi(g'))$. Calcular $r' = H_3(\sigma', m')$. Se $U = r'(H_1(ID_A)P_1 + P_{pub})$, m' é a resposta, senão C é rejeitado.

A.9 CL-PKE de Shi-Li

Em (SHI; LI, 2005), o esquema de criptografia e decriptografia baseado em identidade de Sakai e Kasahara, SK-IBE, com segurança IND-CCA pelo fortalecimento da primeira versão da transformação de Fujisaki-Okamoto, e que foi descrito e comprovado seguro em (CHEN; CHENG, 2005), foi usado como base para produzir um novo esquema de criptografia de chave pública sem certificado. O esquema resultante é definido pelos algoritmos a seguir:

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \psi, \hat{e}, n, P_1, P_2, P_{pub}, g, H_1, H_2, H_3, H_4, H_5\rangle$, onde $\mathbb{G}_1$, $\mathbb{G}_2$ e $\mathbb{G}_T$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ é um emparelhamento bilinear admissível, P_2 é gerador de $\mathbb{G}_2$, $P_1 = \psi(P_2)$, ψ é um isomorfismo, $g = \hat{e}(P_1, P_2)$, $P_{pub} = sP_1$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$

$H_2 : \mathbb{G}_T \to \{0,1\}^n$

$H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^n \to \{0,1\}^n$

$H_5 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n \times \{0,1\}^n$.

secreta-parcial. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular a chave secreta parcial $d_A = \frac{1}{s+H_1(ID_A)}P_2$. No caso em que $s + H_1(ID_A) = 0$, define-se $\frac{1}{0} = 0$, $0P_2 = O$ (ponto no infinito da curva elíptica, ou seja, o elemento neutro de $\mathbb{G}_2$) e $d_A = O$ (esse caso especial não é citado no original).

info-secreta. Selecionar ao acaso uma informação secreta $t_A \in \mathbb{Z}_q^*$, que é a informação secreta da entidade A.

secreta. Dados params, a chave secreta parcial d_A e a informação secreta t_A, a chave secreta (completa) é $S_A = \langle d_A, t_A\rangle$.

publica. Dados params e a informação secreta t_A, a chave pública é $N_A = t_A P_2$,

cript. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A e params. Se $N_A \notin \mathbb{G}_2^*$, a chave pública é inválida e deve-se abortar a criptografia.

Calcular o texto cifrado $\langle r_1 Q_A + r_2 N_A, \sigma \oplus H_2(g^{r_1+r_2}), m \oplus H_4(\sigma) \rangle$, onde $Q_A = H_1(ID_A)P_1 + P_{pub}$, $r_1 = H_3(m, \sigma)$, $r_2 = H_5(m, \sigma)$, para $\sigma \in \{0,1\}^n$, escolhido aleatoriamente.

decript. Dados $C = \langle U, V, W \rangle \in \mathcal{C}$, a chave secreta $S_A = \langle d_A, t_A \rangle$ e params:

Calcular $V \oplus H_2(\hat{e}(U, d_A - \frac{1}{t_A} P_1)) = \sigma$ e $W \oplus H_4(\sigma) = m$. Com $r_1 = H_3(\sigma, m)$, $r_2 = H_5(\sigma, m)$, verificar se $U = r_1 Q_A + r_2 N_A$. Se for, m é a resposta, senão C é rejeitado.

A.10 CL-PKS de Huang

Em (HUANG et al., 2005), é apontada uma falha no CL-PKS de (AL-RIYAMI; PATERSON, 2003) e, em seguida, é proposto um esquema de assinatura ligeiramente modificado para corrigir a falha. O novo esquema foi demonstrado seguro; é composto pelos algoritmos:

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_{pub}, H_1, H_2 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_1 : \{0,1\}^* \to \mathbb{G}_1^*$

$H_2 : \mathbb{G}_2 \to \{0,1\}^n$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de assinaturas é $\mathcal{S} = \mathbb{G}_1^* \times \{0,1\}^n$.

parcial. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular $Q_A = H_1(ID_A)$ e a chave secreta parcial $D_A = sQ_A$.

valor-secreto. Dado params, a entidade A escolhe aleatoriamente a informação secreta x_A.

chave-secreta. Dados params, a chave parcial D_A e a informação secreta x_A, A calcula e mantém em sigilo sua chave de assinatura $S_A = x_A D_A$.

publica. Dados params, e a informação secreta x_A, A calcula a chave pública $P_A = (x_A P, x_A P_{pub})$.

assina. Dados um texto $M \in \mathcal{M}$, uma identidade ID_A, a chave secreta S_A, a assinatura é $\langle U, v \rangle$, onde:

$U = vS_A + rP$

$v = H_2(M, e(rP, P), e(S_A, P))$

r é escolhido aleatoriamente de $\mathbb{Z}_q^*$

verifica. Dadas a chave pública $P_A = (X, Y)$ e a assinatura $\langle U, v \rangle$ sobre M, realizar os seguintes testes:

- $e(X, P_{pub}) \stackrel{?}{=} e(Y, P)$
- $v \stackrel{?}{=} H_2(M, e(U, P)e(Q_A, -Y)^v, e(Q_A, Y))$

Se ambas igualdades forem satisfeitas, a assinatura é aceita; caso contrário, é rejeitada.

A.11 IBS de Barreto

Em (BARRETO et al., 2005) é descrito um esquema de assinatura IBS bastante eficiente; é composto dos algoritmos abaixo:

inicializa. Dado um parâmetro de segurança 1^k, o KGC:

1. Gera dois grupos cíclicos G_1, G_2, $G_1 \neq G_2$, e G_T de ordem prima $p > 2^k$ e um emparelhamento bilinear $e : G_1 \times G_2 \rightarrow G_T$.
   Escolhe aleatoriamente dois geradores $P \in G_1^*, Q \in G_2^*$, tais que $P = \psi(Q)$, onde $\psi()$ é um homomorfismo de G_2^* em G_1^*, eficientemente computável (isto é, de complexidade de tempo polinomial em k).
2. Calcula $g = e(P, Q) \in G_T$.
3. Escolhe aleatoriamente a chave-mestra $s \in Z_p^*$ e calcula $Q_{pub} = sQ$.
4. Escolhe funções de *hash*
   $H_1 : \{0,1\}^* \rightarrow Z_p^*$
   $H_2 : \{0,1\}^* \times \{0,1\}^* \times G_T \times G_T \rightarrow Z_p^*$.
   O espaço de mensagens é $\mathcal{M} = \{0,1\}^*$. O espaço de assinatura é $\mathcal{S} = G_1 \times Z_p^*$. A `chave-mestra` do sistema é s. Os parâmetros públicos do sistema são

$$\texttt{params} = \langle p, G_1, G_2, G_T, e(), \psi, P, Q, Q_{pub}, g, H_1, H_2 \rangle$$

gerachaves. Dada uma identidade $ID_A \in \{0,1\}^*$, `params` e a `chave-mestra` s, o KGC calcula: $S_{ID} = \frac{1}{H_1(ID_A)+s} P \in G_1^*$.

assina. Dados `params`, uma mensagem $M \in \mathcal{M} = \{0,1\}^*$, uma identidade ID_A, a chave secreta de assinatura de S_{ID}, A assina M da seguinte forma:

1. Escolhe aleatoriamente $x \in Z_p^*$.

2. Calcula
$$\begin{cases} r = g^x \in G_T \\ h = H_2(M, r) \in Z_p^* \\ S = (x + h)S_{ID} \in G_1 \end{cases}$$
3. A assinatura sobre M é $\sigma = (S, h) \in G_1 \times Z_p^*$.

verifica. Dados `params`, uma mensagem M, a assinatura $\sigma = (S, h)$, a identidade ID_A, o algoritmo gera 1, aceitando σ como autêntica, se e somente se:

$$h = H_2(M, e[S, H_1(ID_A)Q + Q_{pub}]g^{-h})$$

caso contrário, responde 0.

A.12 CL-PKS de Zhang

Em (ZHANG et al., 2006), é proposto um novo esquema de CL-PKS, eficiente e comprovado seguro; é composto pelos algoritmos:

inicializa. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_{pub}, H_1, H_2, H_3 \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem prima q, $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear admissível, P é gerador de $\mathbb{G}_1$, $P_{pub} = sP$ e H_i são funções de hash, tais que:

$H_i : \{0,1\}^* \to \mathbb{G}_1^*$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$. O espaço de assinaturas é $\mathcal{S} = \mathbb{G}_1^* \times \mathbb{G}_1^*$.

parcial. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular $Q_A = H_1(ID_A)$ e a chave secreta parcial $D_A = sQ_A$.

valor-secreto. Dado params, a entidade A escolhe aleatoriamente a informação secreta x_A.

chave-secreta. Dados params, a chave parcial D_A e a informação secreta x_A, A calcula e mantém em sigilo sua chave de assinatura S_A formada pelo par (x_A, D_A).

publica. Dados params, e a informação secreta x_A, A calcula a chave pública $P_A = x_A P$.

assina. Dados um texto $M \in \mathcal{M}$, uma identidade ID_A, a chave secreta $S_A = (x_A, D_A)$, a assinatura é $\langle U, V \rangle$, onde:

$U = rP$

$V = D_A + rH_2(M, ID_A, P_A, U) + x_A H_3(M, ID_A, P_A)$

e r é escolhido aleatoriamente de $\mathbb{Z}_q^*$

verifica. Dadas a chave pública P_A e a assinatura $\langle U, V \rangle$ sobre M, realizar o seguinte teste:

$$e(V, P) \stackrel{?}{=} e(Q_A, P_{pub})e(H_2(M, ID_A, P_A, U), U)e(H_3(M, ID_A, P_A), P_A)$$

Se a igualdade for satisfeita, a assinatura é aceita; caso contrário, é rejeitada.

A.13 Resumo da Evolução dos Esquemas Relacionados

Na tabela A.1, são relacionadas as fórmulas para geração de cifras dos vários esquemas relacionados com CL-PKE-Proposto. A relação é colocada em ordem cronológica, isto é, a primeira linha da tabela refere-se ao esquema mais antigo, enquanto as seguintes listam sucessivamente os esquemas posteriores. O propósito desta tabela é evidenciar, de forma resumida, como os componentes de formulação das cifras evoluiram de um trabalho para outro e influenciaram na concepção de CL-PKE-Proposto. Para facilitar as comparações, foram efetuadas algumas mudanças de nomes de variáveis.

Esquema	Texto cifrado C
(BONEH; FRANKLIN, 2001) (IBE)	$\langle rP, \sigma \oplus H_2(\hat{e}(Q_A, P_{pub})^r), m \oplus H_4(\sigma) \rangle$
(AL-RIYAMI; PATERSON, 2003)	$\langle rP, \sigma \oplus H_2(\hat{e}(Q_A, t_A P_{pub})^r), m \oplus H_4(\sigma) \rangle$
(GENTRY, 2003)	$\langle rP, \sigma \oplus H_2\left((\hat{e}(P_{pub}, P_A)\hat{e}(N_A, P'_A))^r\right), m \oplus H_4(\sigma) \rangle$
(AL-RIYAMI; PATERSON, 2005)	$\langle rP, \sigma \oplus H_2(\hat{e}(Q_A, P_{pub})^r) \oplus H_5(rN_A), m \oplus H_4(\sigma) \rangle$
(CHENG; COMLEY, 2005)	$\langle rP, \sigma \oplus H_2(rP, \hat{e}(Q_A, P_{pub})^r, rN_A), m \oplus H_4(\sigma) \rangle$
(GALINDO, 2005) (IBE)	$\langle tP, (m \parallel \sigma) \oplus H_2(\hat{e}(Q_A, P_{pub})^t) \rangle$
CL-PKE-Proposto	$\langle tP, (m \parallel \sigma) \oplus H_2(tP, \hat{e}(Q_A, P_{pub})^t, tN_A) \rangle$

onde:

$r = H_3(m, \sigma)$, com $m, \sigma \in \{0,1\}^n$ (FUJISAKI; OKAMOTO, 1999)

$t = H_3(m, \sigma)$, com $m \in \{0,1\}^{n-k_0}$, $\sigma \in \{0,1\}^{k_0}$ (FUJISAKI; OKAMOTO, 2000)

Tabela A.1: Relação das cifras dos esquemas relacionados com CL-PKE-Proposto.

Na tabela A.2, são listadas as fórmulas para assinaturas/verificações dos esquemas relacionados com CL-PKS-Proposto, excluindo-se o CL-PKS de (AL-RIYAMI; PATERSON, 2003) que continha erro. A relação é colocada em ordem cronológica, do esquema mais antigo ao mais recente. O propósito desta tabela é evidenciar, de forma resumida, a formulação das assinaturas e verificações.

Esquema	Assinatura	Verificação
(HUANG et al., 2005)	$\langle U = vtD + rP,$ $v = H(M, R, e(tD, P))\rangle$	$v \stackrel{?}{=} H(M, e(U, P)e(Q_i, -N)^v,$ $e(Q_i, N))$
(BARRETO et al., 2005) (IBS)	$\langle S = (r + h)D, h\rangle$	$h \stackrel{?}{=} H[M, ID, e(S, Q_{ID})g^{-h}]$
CL-PKS-Proposto	$\langle S = (r + ht)D, h\rangle$	$h \stackrel{?}{=} H[M, ID, N, e(S, Q_{ID})(N)^{-h}]$
(ZHANG et al., 2006)	$\langle U = rP, V = D + rh_2 + th_3\rangle$	$e(V, P) \stackrel{?}{=} e(h_1, P_{pub})e(h_2, U)e(h_3, N)$

onde:

$g = e(P_1, P_2)$, parâmetro público constante (SAKAI; KASAHARA, 2003)

D, t são secretos, N é chave pública, $R = e(rP, P)$, $Q_i = H_1(ID)$, $Q_{ID} = H_1(ID)Q + Q_{pub}$.

Tabela A.2: Relação das assinaturas dos esquemas relacionados com CL-PKS-Proposto.

Apêndice B

CL-PKE-Proposto2

Neste apêndice, é apresentada uma segunda proposta de esquema de criptografia e decriptografia baseada no modelo de criptografia de chave pública sem certificados. O esquema que nomeamos CL-PKE-Proposto2 é definido pelos algoritmos abaixo descritos.

inicializa2. Dado um parâmetro de segurança k, gera a chave-mestra $s \in \mathbb{Z}_q^*$ e os parâmetros públicos do sistema params $= \langle q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \psi, \hat{e}, n, k_0, P_1, P_2, P_{pub}, g, \varphi(g^r), H_1, H_2, H_3, H_4 \rangle$, onde $\mathbb{G}_1$, $\mathbb{G}_2$ e $\mathbb{G}_T$ são grupos de ordem prima q, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ é um emparelhamento bilinear admissível, P_2 é gerador de $\mathbb{G}_2$, $P_1 = \psi(P_2)$, φ é o algoritmo de compactação de emparelhamento de (SCOTT; BARRETO, 2004) com imagem $\mathbb{F}$, $\varphi(g^r) = \varphi(\hat{e}(P_1, P_2)^r)$, $P_{pub} = sP_1$ e H_i são funções de *hash*, tais que:

$H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$

$H_2 : \mathbb{G}_1 \times \mathbb{F} \to \{0,1\}^n$

$H_3 : \{0,1\}^{n-k_0} \times \{0,1\}^{k_0} \to \mathbb{Z}_q^*$

$H_4 : \{0,1\}^{n-k_0} \times \{0,1\}^{k_0} \to \mathbb{Z}_q^*$

O espaço de mensagens é $\mathcal{M} = \{0,1\}^{n-k_0}$. O espaço de textos cifrados é $\mathcal{C} = \mathbb{G}_1^* \times \{0,1\}^n$.

extrai2. Dados um identificador $ID_A \in \{0,1\}^*$, params e a chave-mestra s, calcular a chave secreta parcial $d_A = \frac{1}{s+H_1(ID_A)} P_2$. No caso em que $s + H_1(ID_A) = 0$, define-se $\frac{1}{0} = 0$, $0P_2 = O$ (ponto no infinito da curva elíptica, ou seja, o elemento neutro de $\mathbb{G}_2$) e $d_A = O$.

publica2. Dados params, a indentificação para a entidade A e a chave secreta parcial d_A, selecionar ao acaso uma informação secreta $t_A \in \mathbb{Z}_q^*$. A chave secreta (completa) é $S_A = \langle d_A, t_A \rangle$, mantida em sigilo. A chave pública é $N_A = t_A P_2$.

cript2. Dados um texto $m \in \mathcal{M}$, uma identidade ID_A e params. Se $N_A \notin \mathbb{G}_2^*$, a chave pública é inválida e deve-se abortar a criptografia; caso contrário, o algoritmo procede com os

seguintes passos:

1. Escolher $\sigma \in \{0,1\}^{k_0}$ aleatoriamente
2. Calcular
   $Q_A = H_1(ID_A)P_1 + P_{pub}$
   $r_1 = H_3(m, \sigma)$
   $r_2 = H_4(m, \sigma)$
   $U = r_1 Q_A + r_2 N_A$
   $V = (m \parallel \sigma) \oplus H_2(U, \varphi(g^{r_1+r_2}))$
3. O texto cifrado é $C = \langle U, V \rangle$.

decript2. Dados $C = \langle U, V \rangle \in \mathcal{C}$, a chave secreta $S_A = \langle d_A, t_A \rangle$ e params:

1. Calcular
   $\varphi(g') = \varphi(\hat{e}(U, d_A - \frac{1}{t_A}P_1))$
   $V \oplus H_2(U, \varphi(g')) = (m' \parallel \sigma')$
   $r'_1 = H_3(m', \sigma')$
   $r'_2 = H_4(m', \sigma')$
2. Se $(U = r'_1 Q_A + r'_2 N_A)$, então m' é a resposta, senão C é rejeitado.

B.1 Validade de CL-PKE-Proposto2

Para se verificar a validade do esquema dado, basta lembrar que o emparelhamento $\hat{e}$ é bilinear e que vale a igualdade abaixo (SAKAI; KASAHARA, 2003):

$$\hat{e}(aP_1 + dP_2, cP_1 + bP_2) = \hat{e}(aP_1, bP_2)\hat{e}(cP_1, -dP_2)$$

Portanto, tem-se:

$$\begin{aligned} g' &= \hat{e}\left(U, d_A - \frac{1}{t_A}P_1\right) \\ &= \hat{e}\left(r_1 Q_A + r_2 N_A, d_A - \frac{1}{t_A}P_1\right) \\ &= \hat{e}\left(r_1 Q_A, d_A\right)\hat{e}\left(-\frac{1}{t_A}P_1, -r_2 N_A\right) \\ &= \hat{e}\left(r_1(H_1(ID_A)P_1 + sP_1), \frac{1}{s + H_1(ID_A)}P_2\right)\hat{e}\left(\frac{1}{t_A}P_1, r_2 t_A P_2\right) \end{aligned}$$

$$
\begin{aligned}
&= \hat{e}\left(r_1(H_1(ID_A)+s)P_1, \frac{1}{s+H_1(ID_A)}P_2\right))\hat{e}\,(P_1, r_2P_2))\\
&= \hat{e}\,(r_1P_1, P_2))\hat{e}\,(P_1, r_2P_2))\\
&= \hat{e}(P_1, P_2)^{r_1}\hat{e}\,(P_1, P_2))^{r_2}\\
&= \hat{e}(P_1, P_2)^{r_1+r_2}\\
&= g^{(r_1+r_2)}
\end{aligned}
$$

Logo $H_2(U, \varphi(g')) = H_2(U, \varphi(g^{r_1+r_2}))$. Como $V = (m \parallel \sigma) \oplus H_2(U, \varphi(g^{r_1+r_2}))$, então

$$V \oplus H_2(U, \varphi(g')) = (m \parallel \sigma)$$

Portanto **decript2** recupera corretamente a mensagem cifrada por **cript2**.

B.2 Origens e Segurança de CL-PKE-Proposto2

De forma resumida, o segundo esquema proposto pode ser descrito como uma reunião de idéias apresentadas em vários trabalhos. A formulação da chave secreta parcial é idêntica à proposta do protocolo de criptografia e decriptografia baseado em identidades (IBE) de (SAKAI; KASAHARA, 2003). Este, por sua vez, em (CHEN; CHENG, 2005) foi demonstrado seguro e acrescido da função φ, para ganho de eficiência computacional e redução da representação dos pontos, por meio da técnica de *emparelhamento compacto*[1], de (SCOTT; BARRETO, 2004).

Em (SAKAI; KASAHARA, 2003) são sugeridas equações eficientes para verificações baseadas em emparelhamentos bilineares. Como resultado, foi possível alcançar esquemas cujo algoritmo de criptografia não requer cálculo de emparelhamento; apenas o algoritmo de decriptografia envolve emparelhamento, operação de alto custo computacional.

Em (SHI; LI, 2005) foi publicado um novo esquema CL-PKE derivado do IBE de (CHEN; CHENG, 2005). Nosso CL-PKE-Proposto2 acrescenta a transformação mais eficiente de (FUJISAKI; OKAMOTO, 2000) e redefine a função de *hash* H_2 sobre o esquema de (SHI; LI, 2005).

Na tabela B.1, são relacionadas as fórmulas para geração de cifras dos vários esquemas relacionados com CL-PKE-Proposto2. A relação é colocada em ordem cronológica, do esquema mais antigo ao mais recente. O propósito desta tabela é evidenciar, de forma resumida, como os componentes de formulação das cifras evoluiram de um trabalho para outro e influenciaram na concepção de CL-PKE-Proposto2. Todos esses trabalhos relacionados são descritos no apêndice A.

[1] Do original inglês, *compressed pairing*.

Esquema	Texto cifrado C
(SAKAI; KASAHARA, 2003) (IBE)	$\langle rQ_A, \sigma \oplus H_2(g^r), m \oplus H_4(\sigma) \rangle$
(SHI; LI, 2005)	$\langle rQ_A + r_2 N_A, \sigma \oplus H_2(g^{r+r_2}), m \oplus H_4(\sigma) \rangle$
(CHEN; CHENG, 2005) (IBE)	$\langle tQ_A, (m \parallel \sigma) \oplus H_2(tQ_A, \varphi(g^t)) \rangle$
CL-PKE-Proposto2	$\langle tQ_A + t_2 N_A, (m \parallel \sigma) \oplus H_2(tQ_A + t_2 N_A, \varphi(g^{t+t_2})) \rangle$

onde:

$g = \hat{e}(P_1, P_2)$, parâmetro público constante	(SAKAI; KASAHARA, 2003)
$r = H_3(m, \sigma)$, com $m, \sigma \in \{0,1\}^n$	(FUJISAKI; OKAMOTO, 1999)
$r_2 = H_5(m, \sigma)$, com $m, \sigma \in \{0,1\}^n$	(FUJISAKI; OKAMOTO, 1999)
$t = H_3(m, \sigma)$, com $m \in \{0,1\}^{n-k_0}$, $\sigma \in \{0,1\}^{k_0}$	(FUJISAKI; OKAMOTO, 2000)
$t_2 = H_4(m, \sigma)$, com $m \in \{0,1\}^{n-k_0}$, $\sigma \in \{0,1\}^{k_0}$	(FUJISAKI; OKAMOTO, 2000)

Tabela B.1: Relação das cifras dos esquemas relacionados com CL-PKE-Proposto2.

Para as demonstrações de que CL-PKE-Proposto2 é seguro contra adversários Tipo-I-CCA2 e Tipo-II-CCA2, o caminho sugerido a tomar é adotar como hipótese a dificuldade dos problemas usados em (SHI; LI, 2005), a saber, *BCAA1* e *co-BIDH*. Proceder, então, de forma semelhante ao desenvolvimento dos lemas da seção 6.1, para demonstração de segurança de CL-PKE-Proposto.

Referências

ADLEMAN, L.; MCCURLEY, K. Open problems in number theoretic complexity, ii. In: *ANTS-I Algorithmic Number Theory Symposium*. Ithaca, NY, USA: Springer-Verlag, 1994. (Lecture Notes in Computer Science, v. 877), p. 291–322.

AL-RIYAMI, S. S. *Cryptographic Schemes based on Elliptic Curve Pairings*. Tese (Tese de Doutorado) — Department of Mathematics, Royal Holloway, University of London, 2005. Disponível em: <http://www.rhul.ac.uk/mathematics/techreports>.

AL-RIYAMI, S. S.; PATERSON, K. G. Certificateless public key cryptography. In: *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security*. Taipei, Taiwan: Springer, 2003. (Lecture Notes in Computer Science, v. 2894). ISBN 3-540-20592-6. Disponível em: <http://eprint.iacr.org/2003/126>.

AL-RIYAMI, S. S.; PATERSON, K. G. Cbe from cl-pke: A generic construction and efficient schemes. In: *Public Key Cryptography - PKC 2005*. Les Diablerets, Switzerland: Springer, 2005. (Lecture Notes in Computer Science, v. 3386), p. 398–415. ISBN 3-540-24454-9.

BAEK, J.; SAFAVI-NAINI, R.; SUSILO, W. Certificateless public key encryption without pairing. In: *ISC*. Springer, 2005. (Lecture Notes in Computer Science, v. 3650), p. 134–148. Disponível em: <http://www.uow.edu.au/∼baek/publications/clpkewp_bss_final.pdf>. Acesso em: setembro de 2005.

BAO, F.; DENG, R. H.; ZHU, H. Variations of diffie-hellman problem. In: *ICICS*. Huhehaote, China: Springer, 2003. (Lecture Notes in Computer Science, v. 2836), p. 301–312.

BARRETO, P. *Curvas Elípticas e Criptografia - Conceitos e Algoritmos*. 1999. Paulo Barreto's Crypto Page. Disponível em: <http://paginas.terra.com.br/informatica/paulobarreto>.

BARRETO, P.; NAEHRIG, M. Pairing-friendly elliptic curves of prime order. In: *Selected Areas in Cryptography, 12th International Workshop, SAC 2005*. Kingston, ON, Canada: Springer, 2005, (Lecture Notes in Computer Science, v. 3897). p. 319–331.

BARRETO, P. S. L. M. et al. Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. In: ROY, B. (Ed.). *Asiacrypt 2005*. Chennai, India: Springer, 2005. (Lecture Notes in Computer Science, v. 3788), p. 515–532.

BELLARE, M. Practice-oriented provable-security. In: *ISW '97: Proceedings of the First International Workshop on Information Security*. London, UK: Springer-Verlag, 1998. p. 221–231. ISBN 3-540-64382-6.

BELLARE, M.; BOLDYREVA, A.; PALACIO, A. *An Uninstantiable Random-Oracle-Model Scheme for a Hybrid Encryption Problem*. 2003. Cryptology ePrint Archive, Report 2003/077. Disponível em: <http://eprint.iacr.org/>.

BELLARE, M. et al. Relations among notions of security for public-key encryption schemes. In: *CRYPTO '98: Proceedings of the 18th Annual International Cryptology Conference on Advances in Cryptology*. London, UK: Springer-Verlag, 1998. p. 26–45. ISBN 3-540-64892-5.

BELLARE, M.; PALACIO, A. *Towards Plaintext-Aware Public-Key Encryption without Random Oracles*. 2004. Cryptology ePrint Archive, Report 2004/221. Disponível em: <http://eprint.iacr.org/>.

BELLARE, M.; ROGAWAY, P. Random oracles are practical: A paradigm for designing efficient protocols. In: *First ACM Conference on Computer and Communications Security*. Fairfax, Virginia, USA: ACM, 1993. p. 62–73.

BENTAHAR, K. et al. *Generic Constructions of Identity-Based and Certificateless KEMs*. 2005. Cryptology ePrint Archive, Report 2005/058. Disponível em: <http://eprint.iacr.org/>.

BERBECARU, D.; LIOY, A.; MARIAN, M. On the complexity of public-key certificate validation. In: *ISC '01: Proceedings of the 4th International Conference on Information Security*. London, UK: Springer-Verlag, 2001. p. 183–203. ISBN 3-540-42662-0.

BONEH, D.; BOYEN, X. Short signatures without random oracles. In: *EUROCRYPT: Advances in Cryptology*. Springer, 2004a. (Lecture Notes in Computer Science, v. 3027), p. 56–73. Disponível em: <http://crypto.stanford.edu/∼dabo/papers/bbsigs.pdf>. Acesso em: setembro de 2005.

BONEH, D.; BOYEN, X. *Secure identity based encryption without random oracles*. 2004b. Disponível em: <citeseer.ist.psu.edu/boneh04secure.html>.

BONEH, D.; BOYEN, X. Efficient selective-ID secure identity-based encryption without random oracles. In: *EUROCRYPT: Advances in Cryptology*. Springer, 2004c. (Lecture Notes in Computer Science, v. 3027), p. 223–238. Disponível em: <http://crypto.stanford.edu/∼dabo/papers/bbibe.pdf>. Acesso em: setembro de 2005.

BONEH, D.; FRANKLIN, M. K. Identity-based encryption from the weil pairing. In: *CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference on*

Advances in Cryptology. London, UK: Springer-Verlag, 2001. p. 213–229. ISBN 3-540-42456-3. Disponível em: <http://eprint.iacr.org/2001/090>.

BONEH, D.; LYNN, B.; SHACHAM, H. Short signatures from the weil pairing. In: *ASIACRYPT '01: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security*. London, UK: Springer-Verlag, 2001. p. 514–532. ISBN 3-540-42987-5.

CANETTI, R.; GOLDREICH, O.; HALEVI, S. *The Random Oracle Methodology, Revisited*. 1998. Cryptology ePrint Archive, Report 1998/011. Disponível em: <http://eprint.iacr.org/>.

CERTICOM. *An Elliptic Curve Cryptography (ECC) Primer*. 2004. The Certicom *"Catch the Curve"* White Paper Series. Disponível em: <http://www.certicom.com/download/aid-317/WP-ECCprimer.pdf>. Acesso em: julho de 2006.

CHA, J. C.; CHEON, J. H. An identity-based signature from gap diffie-hellman groups. In: *PKC '03: Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography*. London, UK: Springer-Verlag, 2003. p. 18–30. ISBN 3-540-00324-X.

CHEN, L.; CHENG, Z. *Security Proof of Sakai-Kasahara's Identity-Based Encryption Scheme*. 2005. Cryptology ePrint Archive, Report 2005/226. Disponível em: <http://eprint.iacr.org/>. Acesso em: agosto de 2005.

CHENG, Z.; COMLEY, R. *Efficient Certificateless Public Key Encryption*. 2005. Cryptology ePrint Archive, Report 2005/012. Disponível em: <http://eprint.iacr.org/>.

CHENG, Z.; COMLEY, R.; VASIU, L. Remove key escrow from the identity-based encryption system. In: *IFIP TCS*. Toulouse, France: Kluwer, 2004. p. 37–50. Disponível em: <http://www.cs.mdx.ac.uk/staffpages/m_cheng/link/ibe_springer_full.pdf>. Acesso em: agosto de 2005.

CHEON, J. H.; LEE, D. H. *Diffie-Hellman Problems and Bilinear Maps*. 2002. Cryptology ePrint Archive, Report 2002/117. Disponível em: <http://eprint.iacr.org/>.

CHOW, S. S. M.; BOYD, C.; NIETO, J. M. G. Security-mediated certificateless cryptography. In: *Public Key Cryptography PKC 2006*. New York, NY, USA: Springer, 2006. (Lecture Notes in Computer Science, v. 3958), p. 508–524.

CRAMER, R.; SHOUP, V. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, v. 33, n. 1, p. 167–226, 2004. ISSN 0097-5397.

DENT, A. W.; KUDLA, C. *On Proofs of Security for Certificateless Cryptosystems*. 2005. Cryptology ePrint Archive, Report 2005/348. Disponível em: <http://eprint.iacr.org/>. Acesso em: setembro de 2005.

DIFFIE, W.; HELLMAN, M. E. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22, n. 6, p. 644–654, 1976.

ELLISON, C.; SCHNEIER, B. Ten risks of PKI: What you're not being told about public-key infrastructure. *Computer Security Journal*, v. 16, n. 1, p. 1–7, 2000. ISSN 0277-0865.

FUJISAKI, E.; OKAMOTO, T. Secure integration of asymmetric and symmetric encryption schemes. In: *CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*. London, UK: Springer-Verlag, 1999. p. 537–554. ISBN 3-540-66347-9.

FUJISAKI, E.; OKAMOTO, T. How to enhance the security of public-key encryption at minimum cost. *IEICE Trans. Fundamentals*, E83-A, n. 1, p. 24–32, 2000.

GALBRAITH, S.; PATERSON, K.; SMART, N. *Pairings for Cryptographers*. 2006. Cryptology ePrint Archive, Report 2006/165. Disponível em: <http://eprint.iacr.org/>. Acesso em: maio de 2006.

GALINDO, D. *Boneh-Franklin Identity Based Encryption Revisited*. 2005. Cryptology ePrint Archive, Report 2005/117. Disponível em: <http://eprint.iacr.org/>.

GAREY, M.; JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY, USA: Freeman, 1979.

GENTRY, C. *Certificate-Based Encryption and the Certificate Revocation Problem*. 2003. Cryptology ePrint Archive, Report 2003/183. Disponível em: <http://eprint.iacr.org/>.

GIRAULT, M. Self-certified public keys. In: *EuroCrypt91*. Brighton, UK: Springer, 1991. p. 490–497. LCNS vol.547.

GOLDREICH, O. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge, UK: Cambridge University Press, 2004. Disponível em: <http://www.wisdom.weizmann.ac.il/∼oded/foc-vol2.html (rascunho)>.

GOLDWASSER, S.; BELLARE, M. *Lecture Notes on Cryptography*. 2001. Disponível em: <http://www.cs.ucsd.edu/users/mihir/papers/gb.html>.

GOLDWASSER, S.; MICALI, S. Probabilistic encryption. *Journal of Computer and Systems Sciences*, v. 28, n. 2, p. 270–299, 1984.

GORDON, D. M. Discrete logarithms in GF(p) using the number field sieve. *SIAM J. Disc. Math.*, v. 6, n. 1, p. 124–138, fev. 1993.

GUTMAN, P. Pki: It's not dead, just resting. *Computer*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 35, n. 8, p. 41–49, 2002. ISSN 0018-9162.

HUANG, X. et al. On the security of certificateless signature schemes from asiacrypt 2003. In: DESMEDT, Y. et al. (Ed.). *CANS*. Xiamen, China: Springer, 2005. (Lecture Notes in Computer Science, v. 3810), p. 13–25. ISBN 3-540-30849-0.

HUANG, X. et al. Certificateless designated verifier signature schemes. In: *AINA 2006, 20th International Conference on Advanced Information Networking and Applications*. Vienna, Austria: IEEE Computer Society, 2006. p. 15–19.

JOUX, A.; NGUYEN, K. *Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups*. 2001. Cryptology ePrint Archive, Report 2001/003. Disponível em: <http://eprint.iacr.org/>.

KANG, B. G.; PARK, J. H. *Is it possible to have CBE from CL-PKE?* 2005. Cryptology ePrint Archive, Report 2005/431. Disponível em: <http://eprint.iacr.org/>. Acesso em: março de 2006.

KOBLITZ, N. Elliptic curve cryptosytems. *Mathematics of Computation*, v. 48, n. 177, p. 203–209, 1987.

KOBLITZ, N. *A course in number theory and cryptography, 2.ed.* New York, NY, USA: Springer-Verlag, 1994.

KOBLITZ, N.; MENEZES, A. *Another Look at "Provable Security"*. 2004. Cryptology ePrint Archive, Report 2004/152. Disponível em: <http://eprint.iacr.org/>. Acesso em: julho de 2004.

KOHNFELDER, L. *A Method for Certification*. Cambridge, MA, 1978.

LEE, Y.-R.; LEE, H.-S. *An Authenticated Certificateless Public Key Encryption Scheme*. 2004. Cryptology ePrint Archive, Report 2004/150. Disponível em: <http://eprint.iacr.org/>.

LEMPEL, A. Cryptology in transition: a survey. *Computing Surveys*, v. 11, p. 285–304, dez. 1979.

LIBERT, B.; QUISQUATER, J. *Identity Based Undeniable Signatures*. 2004. B. Libert and J.-J. Quisquater, Identity Based Undeniable Signatures, Topics in Cryptology CT-RSA'04, LNCS 2964, pp. 112-125, Springer, 2004.

LIBERT, B.; QUISQUATER, J.-J. On constructing certificateless cryptosystems from identity based encryption. In: *Public Key Cryptography 2006 (PKC'06)*. New York, NY, USA: Springer-Verlag, 2006. (Lecture Notes in Computer Science, v. 3958), p. 474–490.

LUCCHESI, C. L. *Introdução à Criptografia*. São Paulo, SP: Quarta Escola de Computação, Instituto de Matemática e Estatística, USP, 1984.

MAURER, U. M.; WOLF, S. The relationship between breaking the diffie-hellman protocol and computing discrete logarithms. *SIAM Journal on Computing*, v. 28, n. 5, p. 1689–1721, 1999. Disponível em: <citeseer.ist.psu.edu/maurer98relationship.html>.

MENEZES, A.; VANSTONE, S.; OKAMOTO, T. Reducing elliptic curve logarithms to logarithms in a finite field. In: *STOC '91: Proceedings of the twenty-third annual ACM*

symposium on Theory of computing. New York, NY, USA: ACM Press, 1991. p. 80–89. ISBN 0-89791-397-3.

MENEZES, A. J.; VANSTONE, S. A.; OORSCHOT, P. C. V. *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, Inc., 1996. Disponível em: <http://www.cacr.math.uwaterloo.ca/hac/>.

MILLER, V. S. Use of elliptic curves in cryptography. In: WILLIAMS, H. C. (Ed.). *CRYPTO 85*. Santa Barbara, California, USA: Springer, 1985. (Lecture Notes in Computer Science, v. 218), p. 417–426.

MIYAJI; NAKABAYASHI; TAKANO. New explicit conditions of elliptic curve traces for FR-reduction. *TIEICE: IEICE Transactions on Communications/Electronics/Information and Systems*, 2001. Disponível em: <citeseer.ist.psu.edu/miyaji01new.html>.

MORIN, P. *The Role of Reductions in Cryptography*. 1996. Ano incerto. Disponível em: <http://cg.scs.carleton.ca/∼morin/>. Acesso em: junho de 2004.

OKAMOTO, T.; POINTCHEVAL, D. The gap-problems: A new class of problems for the security of cryptographic schemes. In: *PKC '01: Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography*. London, UK: Springer-Verlag, 2001. p. 104–118. ISBN 3-540-41658-7.

POINTCHEVAL, D.; STERN, J. Security arguments for digital signatures and blind signatures. *Journal of Cryptology: the journal of the International Association for Cryptologic Research*, v. 13, n. 3, p. 361–396, 2000.

RIVEST, R. L. *Cryptography - Chapter 13 of Handbook of Theoretical Computer Science*. Elsevier ed. J. Van Leeuwen, 1990. 717–755 p. Disponível em: <http://theory.lcs.mit.edu/∼rivest/publications.html>.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. M. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, v. 21, n. 2, p. 120–126, 1978.

SAEEDNIA, S. A note on girault's self-certified model. *Inf. Process. Lett.*, Elsevier North-Holland, Inc., Amsterdam, The Netherlands, The Netherlands, v. 86, n. 6, p. 323–327, 2003. ISSN 0020-0190.

SAKAI, R.; KASAHARA, M. *ID based Cryptosystems with Pairing on Elliptic Curve*. 2003. Cryptology ePrint Archive, Report 2003/054. Disponível em: <http://eprint.iacr.org/>. Acesso em: agosto de 2005.

SAKAI, R.; OHGISHI, K.; KASAHARA, M. Cryptosystems based on pairing. In: *Symposium on Cryptography and Information Security (SCIS2000)*. Okinawa, Japan: Inst. of Electronics, Information and Communication Engineers, 2000. p. 26–28.

SCOTT, M.; BARRETO, P. S. L. M. Compressed pairings. In: . Springer, 2004. (Lecture Notes in Computer Science, v. 3152), p. 140–156. Disponível em: <http://eprint.iacr.org/2004/032>. Acesso em: agosto de 2005.

SHAMIR, A. On the cryptocomplexity of knapsack systems. In: *STOC '79: Proceedings of the eleventh annual ACM symposium on Theory of computing*. New York, NY, USA: ACM Press, 1979. p. 118–129.

SHAMIR, A. Identity-based cryptosystems and signature schemes. In: *Proceedings of CRYPTO 84 on Advances in cryptology*. New York, NY, USA: Springer-Verlag New York, Inc., 1984. p. 47–53. ISBN 0-387-15658-5.

SHI, Y.; LI, J. *Provable Efficient Certificateless Public Key Encryption*. 2005. Cryptology ePrint Archive, Report 2005/287. Disponível em: <http://eprint.iacr.org/>. Acesso em: agosto de 2005.

SHOUP, V. Lower bounds for discrete logarithms and related problems. *Lecture Notes in Computer Science*, v. 1233, p. 256–266, 1997. Disponível em: <citeseer.ist.psu.edu/shoup97lower.html>.

SILVERMAN, J. H.; SUZUKI, J. Elliptic curve discrete logarithms and the index calculus. In: *ASIACRYPT '98: Proceedings of the International Conference on the Theory and Applications of Cryptology and Information Security*. London, UK: Springer-Verlag, 1998. p. 110–125. ISBN 3-540-65109-8.

SMART, N. P. The discrete logarithm problem on elliptic curves of trace one. *J. Cryptology*, v. 12, n. 3, p. 193–196, October 1999. ISSN 0933-2790.

TERADA, R. *Segurança de Dados - Criptografia em Redes de Computador*. São Paulo, SP: Editora Edgard Blücher, 2000.

TERADA, R.; GOYA, D. H. *An Improved Certificateless Public Key Encryption*. Hiroshima, Japan: Inst. of Electronics, Information and Communication Engineers, 2006a. 17-20 p. The 2006 Symposium on Cryptography and Information Security, SCIS 2006. Também submetido para *International Journal of Security on Networks*.

TERADA, R.; GOYA, D. H. *A Certificateless Signature Scheme based on Bilinear Pairing Functions*. 2006b. Submetido para *"1st International Workshop on Security"* (IWSEC2006), Kyoto, Japão, Outubro de 2006.

WATERS, B. R. *Efficient Identity-Based Encryption Without Random Oracles*. 2004. Cryptology ePrint Archive, Report 2004/180. Disponível em: <http://eprint.iacr.org/>.

WIN, E. D.; PRENEEL, B. *Elliptic Curve Publik Key Cryptossystems - an Introduction*. 1998. 131-141 p. LCNS vol.1528. Disponível em: <ftp://ftp.esat.kuleuven.ac.be/pub/cosic/dewin/coursetext.ps.gz>. Acesso em: julho de 2004.

YACOBI, Y. *A Note on the Bilinear Diffie-Hellman Assumption*. 2002. Cryptology ePrint Archive, Report 2002/113. Disponível em: <http://eprint.iacr.org/>.

YUM, D. H.; LEE, P. J. Generic construction of certificateless encryption. In: *ICCSA 2004*. Assisi, Italy: Springer, 2004a. (Lecture Notes in Computer Science, v. 3043), p. 802–811.

YUM, D. H.; LEE, P. J. Generic construction of certificateless signature. In: *ACISP 2004*. Sydney, Australia: Springer-Verlag, 2004b. (Lecture Notes in Computer Science, v. 3108), p. 200–211.

YUM, D. H.; LEE, P. J. Identity-based cryptography in public key management. In: *EuroPKI 2004*. Samos Island, Greece: Springer-Verlag, 2004c. (Lecture Notes in Computer Science, v. 3093), p. 71–84.

ZHANG, Z.; FENG, D. *On the Security of a Certificateless Public-Key Encryption*. 2005. Cryptology ePrint Archive, Report 2005/426. Disponível em: <http://eprint.iacr.org/>. Acesso em: março de 2006.

ZHANG, Z. et al. Certificateless public key signature: Security model and efficient construction. In: *4th. International Conference on Applied Cryptography and Network Security*. Singapore: Springer, 2006. (Lecture Notes in Computer Science, v. 3989).