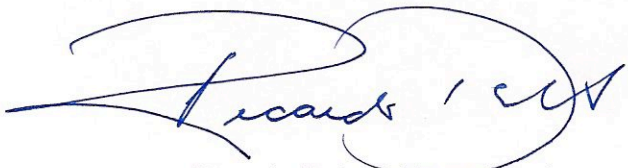
Assinaturas de Chave Pública sem Certificados

Este exemplar corresponde à redação da Dissertação apresentada para a Banca Examinadora antes da defesa da Dissertação.

Campinas, 26 de abril de 2009.



Ricardo Dahab (Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Instituto de Computação
Universidade Estadual de Campinas

Assinaturas de Chave Pública sem Certificados

Rafael Dantas de Castro[1]

Abril de 2009

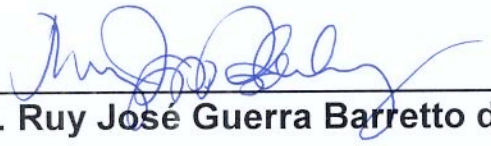
Banca Examinadora:

- Ricardo Dahab (Orientador)
- Ruy J. Guerra B. de Queiroz (UFPE)
- Julio C. López Hernández (UNICAMP)
- Jeroen van de Graaf (UFOP) - suplente
- Jacques Wainer (UNICAMP) - suplente

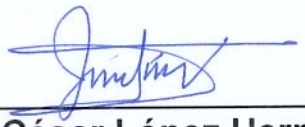
[1]Suporte financeiro de: Bolsa da FAPESP (processo 2006/06146-3) 2007,

TERMO DE APROVAÇÃO

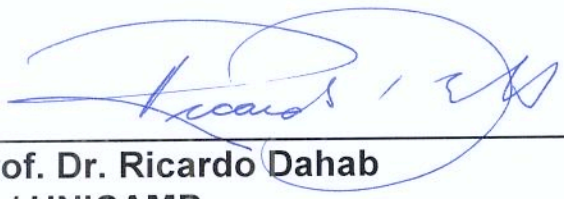
Dissertação Defendida e Aprovada em 15 de dezembro de 2008, pela Banca examinadora composta pelos Professores Doutores:



Prof. Dr. Ruy José Guerra Barretto de Queiroz
CIN / UFPE.



Prof. Dr. Julio César López Hernández
IC / UNICAMP.



Prof. Dr. Ricardo Dahab
IC / UNICAMP.

Resumo

Nesta tese é apresentada uma visão do campo de Assinaturas de Chave Pública sem Certificados *(CL-PKC)*, resultado final de pesquisa empreendida durante o mestrado do autor na Universidade Estadual de Campinas. A abordagem aqui contida tem como principal base a *Segurança Demonstrável*; analisamos os modelos de segurança de CL-PKC, os esquemas já propostos na literatura, e suas demonstrações de segurança. Contribuímos também alguns novos resultados para a área, especificamente:

- estudo da aplicabilidade do Lema da Bifurcação à esquemas de CL-PKC (§ 4.3.3);
- algumas pequenas otimizações a esquemas seguros (§ 4.4.1);
- demonstração de segurança para um esquema cuja segurança ainda era um problema em aberto (§ 4.4.2);
- explicação de falhas em demonstrações de segurança de alguns esquemas (§ 4.4.4, § 4.4.5, § 4.4.7);
- um ataque desconhecido a um esquema anteriormente suposto seguro (§ 4.4.4);
- proposta de um modelo de segurança para agregação de assinaturas no modelo CL-PKC (§ 5.4);
- proposta de um novo esquema de assinaturas CL-PKC que permite agregação, assim como sua prova de segurança (§ 5.2).

Abstract

In this thesis is presented a broad view of the field of Certificateless Public Key Signatures *(CL-PKC)*, final result of the research undertaken by the author during his masters at the Campinas State University. Our approach is strongly based on the ideas behind *Provable Security*; we analyze the security models for CL-PKC, all schemes available in the literature, and their security proofs. We also contribute a few novel results, namely:

- applicability of the *Forking Lemma* in the CL-PKC paradigm (§ 4.3.3);
- small optimizations to secure schemes (§ 4.4.1);
- security proof for a scheme whose security was an open problem (§ 4.4.2);
- investigation of the problems in the security proofs of a few schemes (§ 4.4.4, § 4.4.5, § 4.4.7);
- an attack on a scheme previously thought to be secure (§ 4.4.4);
- proposal of a security model for aggregation of signatures in the CL-PKC paradigm (§ 5.4);
- proposal of a new scheme for Certificateless Signatures that allows aggregation, along with its security proof (§ 5.2).

Agradecimentos

Eu gostaria de agradecer primeiramente a meus pais, Ayala e Castro, que me proporcionaram inúmeras oportunidades ao longo da minha vida e me ofereceram um apoio inestimável. tudo que eles fizeram por mim culmina neste trabalho aqui contido e, se Deus quiser, continuarão me acompanhando por muitos e muitos novos desafios. Quero agradecer ao meu orientador, Dahab, pelo apoio e confiança que tornaram este trabalho possível. Quero agradecer também a tantos outros mestres e companheiros que me acompanharam e de quem muito aprendi, sejam professores como meu querido Lucchesi, ou companheiros como a Cândida, o Alberto, meus colegas de laboratório Augusto, Diego e Leo, meus colegas de maratona, meus colegas de turma. E, finalmente, quero agradecer à minha família postiça, sem a qual esses anos em Campinas teriam sido impossíveis: Renato, Rajiv, Fábio, Betha, Leandro, Paulinha, Mari, Drika, Lucas, Paulão: muito obrigado.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Neste trabalho estudamos assinaturas digitais sem certificados. O paradigma de criptografia de chave pública sem certificados *(CL-PKC)*[1] é uma interessante nova proposta, inicialmente apresentada em [ARP03], que segue o paradigma da certificação implícita de chaves públicas. A idéia básica de CL-PKC é conseguir um compromisso entre a simplicidade encontrada na Criptografia Baseada em Identidades *(IBC)*[2] e o nível de segurança da Criptografia de Chave Pública tradicional (ou *explicitamente certificada*).

Tradicionalmente certificados são necessários para a autenticação de chaves públicas: como não podemos supor que a distribuição das chaves é feita de maneira segura, é necessário que elas sejam autenticadas de alguma forma, como através de certificados emitidos por uma autoridade (supostamente) confiável que deve se encarregar de verificar a identidade do dono da chave. A confiança depositada na autoridade estende-se então à chave. Este sistema, no entanto, gera uma série de problemas administrativos e de infraestrutura.

A principal característica de IBC é a ausência de chaves públicas (e, conseqüentemente, de certificados), o que traz uma simplicidade imensa para o funcionamento do sistema como um todo. Infelizmente a contra-partida desta simplicidade é o excesso de confiança que deve ser depositada na autoridade central, que conhece as chaves privadas de todos os usuários, podendo portanto personificá-los a qualquer momento sem ser detectada. No mundo das chaves públicas explicitamente certificadas, por outro lado, não se exige um nível tão alto de confiança: Autoridades Certificadoras podem agir de maneira desonesta, mas esta ação sempre pode ser detectada (por exemplo, pela existência de dois certificados válidos para uma mesma identidade) e a autoridade pode ser punida.

Idealmente, desejaríamos um sistema simples como IBC mas que atingisse níveis altos de segurança: isto, infelizmente, não é possível. De fato, não existe nada de especial em

[1] Do inglês *Certificateless Public-Key Cryptography.*
[2] Do inglês *Identity-Based Cryptography.*

relação às identidades dos usuários que permita, por exemplo, a Alice computar *apenas* a sua chave privada mas não a de Bob. O que Al-Riyami e Paterson propôem, de maneira semelhante a proposta anteriores como a de Girault [Gir91], é encontrar um meio-termo, apresentando um paradigma o mais simples possível (ainda que não tão simples quanto IBC) mas que preserve as propriedades de segurança desejadas. Acontece que não pode-se dispensar totalmente as chaves públicas: elas são exatamente a "coisa especial" que diferencia as identidades de dois usuários. Pode-se, no entanto, dispensar os certificados utilizando a idéia de *certificação implícita*, onde as chaves públicas dos usuários não são autenticadas (explicitamente) por objetos independentes que devem ser disponibilizados junto às chaves, mas sim (implicitamente) pelo seu uso correto.

Mais concretamente, gostaríamos de obter um sistema onde, se Alice encontra uma chave pública que ela acredita ser de Bob e uma assinatura relativa a esta chave numa mensagem conhecida, a corretude da assinatura prova a autenticidade da chave pública, e nenhum certificado é necessário. Em outras palavras, nenhum adversário é capaz de gerar um par *(chave_pública,assinatura)* e atribuí-los a um outro usuário qualquer. Note-se que esquemas tradicionais de assinaturas digitais (explicitamente certificados) não gozam desta propriedade.

É importante ressaltar que CL-PKC não traz novas funcionalidades para assinaturas digitais: consegue-se sim, devido à ausência de certificados, uma grande simplificação do modelo e um potencial ganho de eficiência do sistema como um todo.

Contribuições desta tese

Este documento reúne uma série de trabalhos realizados ao longo do mestrado do autor, que compôem uma visão atualizada da criptografia de chave pública sem certificados voltada para esquemas de assinatura. Revisamos todas as propostas de esquemas disponíveis na literatura, analisando sua segurança e eficiência, apontamos algumas vulnerabilidades anteriormente desconhecidas e, em alguns casos, propomos correções apropriadas; finalmente, propomos um novo esquema altamente eficiente e que conta com uma propriedade adicional bem interessante: a possibilidade de agregar assinaturas[3].

Duas contribuições relevantes também foram obtidas pelo autor e não estão descritas nesta tese:

- em co-autoria com Augusto Devegili e Ricardo Dahab foi ministrado um mini-curso de introdução à segurança demonstrável no SBSeg 2007. Para acompanhar o mini-curso, foi publicado o capítulo de um livro.
- em co-autoria com Diego Aranha, Ricardo Dahab e Julio Lopez foi proposto um esquema de cifrassinaturas para CL-PKC em um short paper no SBSeg 2008. Este

[3]Mais sobre agregação de assinaturas na seção §5.1

esquema ainda é tema de pesquisa ativa, pois ainda não conseguimos uma demonstração de sua segurança.

Organização deste documento

O próximo capítulo traz um tratamento geral da idéia de certificação ímplicita de chaves públicas, da qual a Criptografia Baseada em Identidade e a Criptografia de Chave Pública sem Certificados são exemplos. Abordamos os primórdios da idéia de certificação implícita, passando por Girault, que formalizou o conceito em [Gir91], evoluindo até propostas recentes, como a própria CL-PKC.

No terceiro capítulo fazemos uma extensa revisão bibliográfica, abordando todos os esquemas de assinaturas digitais sem certificados de que temos conhecimento. Nossa discussão será balizada principalmente pela segurança e eficiência dos esquemas. Como os esquemas apresentados utilizam emparelhamentos bilineares, o capítulo inclui uma seção introdutória aos emparelhamentos e suas propriedades mais interessantes para a criptografia. Este capítulo traz alguns resultados originais de nossa autoria, como uma versão mais eficiente e demonstravelmente segura de um dos esquemas e um ataque a um outro esquema que acreditava-se ser seguro.

O quarto capítulo é dedicado à segurança de esquemas de assinaturas sem certificados. Discutimos variados aspectos de modelos de segurança e detalhes das demonstrações. Fazemos novamente uma extensa revisão bibliográfica, desta vez centrada nas técnicas de demonstração utilizadas em vários dos esquemas. Apresentamos também alguns resultados originais, especificamente:

- mostramos que a suposição de que um adversário deve conhecer a chave secreta correspondente à chave pública que utiliza para montar um ataque é falsa; vários dos esquemas utilizam esta suposição em suas demonstrações e mostramos que ela não se sustenta através de um ataque a um deles;
- caracterizamos esquemas em que a utilização da Técnica de Reexecução de Oráculos é válida; novamente, muitos esquemas a utilizam em suas demonstrações, mas nenhum estudo formal da sua aplicabilidade havia sido feito até então.
- apresentamos demonstrações de segurança para alguns esquemas que não eram demonstravelmente seguros até então (ou cujas provas originais estavam incorretas/imprecisas).

No quinto capítulo apresentamos a principal contribuição desta dissertação, um esquema de assinaturas sem certificados bastante eficiente, demonstravelmente seguro, e que aceita agregação de assinaturas. Este esquema tem um procedimento de agregação bastante eficiente, mantendo constante o tamanho da assinatura mesmo que se agregue um número grande de assinaturas individuais.

O sexto capítulo traz algumas considerações finais, a lista de publicações geradas durante o mestrado do autor e sugestões para pesquisas futuras na área de assinaturas digitais sem certificados.

Capítulo 2

Criptografia com certificação implícita de chaves públicas

Neste capítulo discutimos a idéia seminal de criptografia de chave pública [DH76]. Revisamos os principais conceitos envolvidos nesta mudança de paradigmas e os principais problemas envolvidos na sua adoção, em especial, aqueles relacionados com o gerenciamento de certificados. Abordamos então as principais alternativas à criptografia de chave pública "tradicional" (ou de *certificação explícita*) que envolvem diferentes tipos de certificação implícita: criptografia baseada em identidades (§2.2), chaves públicas "auto-certificadas" (§2.3) e, finalmente, o principal foco deste trabalho, a criptografia de chave pública sem certificados (§2.4).

2.1 Criptografia de chave pública

Certamente a grande revolução na Criptografia dos últimos 40 anos foi gerada pelo artigo "New Directions in Cryptography", de Diffie & Hellman [DH76]. Este artigo seminal trouxe a idéia de criptografia de chave pública, ou criptografia assimétrica, onde comunicação segura não depende mais de um segredo compartilhado: se anteriormente duas partes que desejavam se comunicar precisavam ter estabelecido previamente algum segredo compartilhado, agora cada usuário poderia publicar um valor (chamado sua chave pública) que permitiria a qualquer outra pessoa enviar-lhe mensagens de maneira segura. A revolução proporcionada por esta simples, porém brilhante, idéia veio na hora certa e teve um crescimento sem precedentes: exatamente no momento em que os computadores estavam aumentando rapidamente de poder, tornando-se capazes de fazer eficientemente os extensos cálculos que a criptografia de chave pública parecia necessitar; além disso, a utilização de computadores como infra-estrutura de comunicação começava a se difundir, e os problemas que a criptografia de chave pública se propunha a resolver se tornavam

cada vez mais importantes. Neste momento, a necessidade e a capacidade evoluíram conjuntamente para potencializar o desenvolvimento da criptografia de chave pública.

Tradicionalmente, se duas pessoas, digamos Alice e Bob, desejavam se comunicar de maneira segura, eles precisavam de alguma maneira estabelecer um segredo compartilhado pelos dois; a segurança da comunicação posterior entre eles derivaria diretamente deste segredo inicial. Aparentemente, o problema que Diffie & Hellman abordam é paradoxal: como permitir que Alice e Bob se comuniquem de maneira segura sem terem estabelecido qualquer segredo previamente? Eles respondem a esta pergunta com a noção de uma *função unidirecional com segredo*:

Definição 1. *Uma função* $f_r : \{0,1\}^* \to \{0,1\}^*$ *é dita* unidirecional com segredo *se existe um par* $(r, s) \in \{0,1\}^* \times \{0,1\}^*$ *tal que as seguintes condições valem:*

1. ***Eficiência.*** *Existe um algoritmo* F *de tempo polinomial tal que* $F(x, r) = f_r(x)$.
2. ***Dificuldade de inversão.*** *Para todo algoritmo probabilístico* A' *de tempo polinomial, todo polinômio positivo* $p(\cdot)$*, todo* n *suficientemente grande e* X_n *uma cadeia aleatória de tamanho* n*, a probabilidade de* A' *inverter* f_r *é desprezível:*

$$\Pr[A'(r, f_r(X_n), 1^n) \in f_r^{-1}(f_r(X_n))] < \frac{1}{p(n)}\,.$$

3. ***Facilidade de inversão com segredo.*** *Existe um algoritmo polinomial determinístico, denotado por* F^{-1}*, que utiliza o segredo* s *para inverter* f_r*; ou seja,* $F^{-1}(F(x, r), s) = x$.

Intuitivamente, uma função f_r é unidirecional com segredo se ela é facilmente calculável mas só pode ser invertida com o conhecimento de um *segredo* s. O problema de comunicação segura estava então resolvido: se Bob deseja receber mensagens de maneira confidencial ele pode escolher uma função f_r que obedeça à Definição 1 e colocar (f, r) num diretório público. Quando Alice quiser enviar-lhe qualquer mensagem m, ela pode calcular $C = f_r(m)$ (utilizando o algoritmo F e a informação pública r) e ter certeza que apenas Bob poderá inverter a função, pois para isso é necessário conhecer s. Neste cenário, o par (f, r) é chamado de *chave pública* de Bob, e s é sua *chave secreta.*

Assinaturas digitais

A idéia anterior também pode ser utilizada para construir esquemas de assinaturas digitais. Informalmente, o que se espera de uma assinatura digital é o mesmo que de sua contraparte tradicional: i.e., que possa ser gerada facilmente pelo usuário legítimo, que seja de difícil falsificação e que possa ser facilmente verificada.

Perceba que se a função unidirecional com segredo f da seção anterior for bijetora, um esquema de assinaturas é trivialmente obtido utilizando F^{-1} para gerar assinaturas e F para verificá-las: como apenas Bob conhece s, só ele poderia calcular eficientemente a assinatura $\sigma = f^{-1}(m) = F^{-1}(m, s)$ para uma mensagem m arbitrária; por outro lado, todos podem a partir das informações públicas verificar se $f(\sigma) \stackrel{?}{=} m$.

2.1.1 O problema da autenticidade de chaves públicas.

Infelizmente, somente a existência da criptografia de chave pública não é suficiente para resolver o problema da comunicação segura e das assinaturas digitais. Se agora Alice e Bob não precisam ter compartilhado um segredo em algum momento do passado, de alguma maneira Alice precisa ter certeza de que, ao utilizar uma chave pública que ela pensa pertencer a Bob, a chave é realmente legítima: em outras palavras, surgiu o problema de autenticidade da chave pública.

Obviamente, se Alice não tiver certeza da autenticidade da chave pública que ela pensa ser de Bob, não pode haver qualquer garantia em relação à segurança da comunicação entre os dois; um adversário malicioso pode ter dado a própria chave a Alice fingindo ser Bob. Pelo mesmo motivo, não se pode obter não-repúdio de assinaturas: Bob pode sempre negar ser o dono verdadeiro da chave pública que Alice usou para verificar sua assinatura. A maneira que se encontrou para superar este tipo de limitação foi o uso de uma entidade confiável, uma *autoridade certificadora* (CA)[1] responsável por verificar que uma dada chave pública realmente pertence a Bob e gerar um certificado que pode ser anexado à chave. Este certificado é, em geral, uma assinatura da CA na chave pública de Bob e, se Alice confiar na CA, é suficiente para provar que aquela chave realmente pertence a Bob; adversários não poderiam gerar um certificado semelhante para enganá-la.

Chamamos este primeiro sistema de *explicitamente certificado*, pois existe um objeto (o certificado emitido pela CA) cuja única função é garantir a autenticidade da chave pública.

2.1.2 PKIs

Uma Infraestrutura de Chaves Públicas *(PKI)*[2] é um arcabouço de software, hardware e pessoal administrativo, destinado a gerenciar o ciclo de vida de certificados digitais: sua criação, distribuição, validação e revogação. O custo da manutenção e operação de tais estruturas é, muitas vezes, proibitivo para operações de pequena escala. Além disso, o custo computacional adicional para geração e verificação de assinaturas é substancial.

[1] Do inglês *Certificate Authority.*
[2] Do inglês *Public-key Infrastructure.*

Em primeiro lugar, a quantidade de dados a ser armazenada e transmitida aumenta bastante: enquanto uma chave pública RSA-1024 típica tem aproximadamente 1KB, um certificado X.509 simples tem por volta de quatro vezes isso. Adicionalmente, a validade do certificado tem que ser verificada, o que geralmente envolve a verificação de uma assinatura, ou mais no caso de PKIs de múltiplos níveis. Levando em conta que a chave pública sendo autenticada provavelmente será utilizada para verificar uma assinatura ou cifrar uma chave simétrica, verificar o certificado aproximadamente dobra o custo da operação que se deseja realizar. Uma boa referência para este assunto é [HP01].

2.2 Criptografia Baseada em Identidades

A idéia de criptografia baseada em identidades *(IBC)*[3]surgiu em 1984, no artigo [Sha84]. Neste trabalho, Shamir discute a idéia de usar a identidade dos usuários como sua chave pública, tornando assim desnecessária toda e qualquer infra-estrutura para a distribuição confiável de chaves públicas. Ele instancia esta idéia propondo um esquema de assinaturas baseado em identidades *(IBS)*[4], mas deixa como um problema em aberto a possibilidade de se construir esquemas de ciframento baseados em identidade *(IBE)*[5]. Apenas em 2000 conseguiu-se uma resposta satisfatória para este problema: independentemente, Boneh & Franklin ([BF01]), e Sakai & Kasahara ([SK03]), propuseram esquemas de ciframento baseados em identidade.

Qualquer combinação de informações que identifique unicamente um usuário pode ser utilizada como sua "identidade" (nome, endereço eletrônico, etc.), desde que seja de conhecimento público. Como a "chave pública" de um usuário é sua identidade, na verdade não há chaves públicas enquanto objetos independentes; portanto também não há qualquer tipo de certificado.

A autoridade confiável aqui é chamada de Autoridade de Confiança *(TA)*[6], e tem muito mais poder do que no caso simples de criptografia de chave pública: a TA é responsável por gerar as chaves privadas de todos os usuários do sistema. Perceba que as chaves secretas precisam realmente ser computadas por uma autoridade confiável pois não existe nada de especial em relação à identidade de um usuário: por exemplo, se Alice fosse capaz de, por si só, calcular a chave secreta correspondente à identidade "Alice", ela também seria capaz de calcular as chaves de "Bob", "Charlie", etc. Deve existir então uma TA que conhece alguma informação secreta que lhe permite o cálculo de chaves secretas.

Como confia-se que a TA certifica-se da identidade dos usuários antes de distribuir cha-

[3]Do inglês *Identity-Based Cryptography*.
[4]Do inglês *Identity-Based Signatures*.
[5]Do inglês *Identity-Based Encryption*.
[6]Do inglês *Trust Authority*.

ves secretas, a identidade de um usuário é *implicitamente certificada* através do uso correto da sua chave privada, pois qualquer um que confie na TA acredita que somente Alice conseguiria obter a chave privada correspondente à sua identidade. Logo, a existência de uma assinatura correta prova, ao mesmo tempo, que Alice possui a chave privada correta (pois *alguém* a conhece e gerou a assinatura e, como a TA é confiável, só Alice poderia obtê-la) e que Alice assinou a mensagem em questão. Um sistema de IBC funciona então da seguinte forma:

1. A TA inicializa o sistema. São gerados os parâmetros do sistema, mpk[7], e o segredo principal, msk[8]. Os parâmetros do sistema devem ser distribuídos de maneira autenticada a todos os potenciais usuários. O segredo principal deve ser conhecido apenas pela TA.

2. Cada usuário que deseja obter sua chave privada comunica-se com a TA, comprovando sua identidade, e recebe a chave privada correspondente. Perceba que a chave privada tem que ser transmitida pela TA ao usuário de maneira sigilosa (e autenticada).

3. Qualquer pessoa pode usar os parâmetros públicos distribuídos pela TA para verificar assinaturas, e/ou cifrar mensagens, desde que conheça a identidade do outro usuário com quem irá interagir.

Se, por um lado IBC traz uma imensa simplicidade ao cenário de criptografia de chave pública, por outro lado, o nível de confiança depositada na TA é muito grande, pois ela é capaz de personificar qualquer usuário do sistema. Isto é uma forma de custódia de chaves e é o principal empecilho para a adoção em larga escala de criptografia baseada em identidade; enquanto este nível de confiança pode ser razoável em ambientes militares ou corporativos, certamente não o é no caso geral.

Para ilustrar o conceito de assinaturas baseadas em identidade apresentamos aqui o esquema original proposto por Shamir em [Sha84]:

`Inicializar.`

Gere um par de chaves RSA $(\langle n, e\rangle, d)$, onde:

- $n = p \cdot q$ é o produto de dois primos grandes;
- $ed \equiv 1 \pmod{\phi(n)}$.

Além disso, a TA escolhe funções de hash H_1 e H_2;

[7] Do inglês *master public key*.
[8] Do inglês *master secret key*.

Os parâmetros públicos são $(\langle n, e\rangle, H_1, H_2)$; d deve ser mantido em segredo.

`Extração de Chave Secreta.`
Seja ID a identidade do usuário; a chave secreta x_{ID} é dada por:

$$x_{ID} = H_1(ID)^d \mod n.$$

`Assinar.`
Seja m a mensagem a ser assinada; o usuário ID, de posse de sua chave secreta x_{ID}, escolhe aleatoriamente $k \leftarrow \mathbb{Z}_n^*$ e calcula:

$$\begin{aligned} r &= k^e \mod n, \\ \sigma &= x_{ID} k^{H_2(r,m)} \mod n. \end{aligned}$$

A assinatura é (σ, r).
`Verificar.`
Seja (σ, r) a assinatura de ID na mensagem m;
Aceite a assinatura se e somente se,

$$\sigma^e \equiv H_1(ID) r^{H_2(r,m)} \pmod n$$

Perceba que a chave secreta do usuário é basicamente uma assinatura (RSA) da TA em sua identidade. Com esta assinatura em mãos, o usuário consegue combiná-la (aqui, de maneira aleatorizada) com a mensagem de maneira segura (e posteriormente verificável) de forma a gerar uma assinatura. A grande maioria dos esquemas de assinaturas baseados em identidades (assim como os sem certificados) são baseados em variações desta idéia.

2.3 Chaves públicas autenticadas implicitamente

Podemos encarar a criptografia baseada em identidades como um sistema de certificação implícita de chaves públicas, onde o usuário tem controle total sobre sua chave pública (sua identidade) e, portanto, a autoridade confiável tem controle total sobre a chave privada correspondente. Por que, então, não tentar chegar a um meio termo? Isto é, podemos sacrificar um pouco da liberdade e simplicidade que se obtém do uso de criptografia baseada em identidades em troca de um nível mais alto de segurança (especificamente, diminuindo o nível de confiança que se deposita na TA)?

A resposta é afirmativa e algumas propostas nesta linha surgiram na última década. Inicialmente, faremos uma análise unificada destas propostas sob denominação de *criptografia com certificação implícita de chaves públicas*, e depois analisaremos mais a fundo uma das propostas, de criptografia de chave pública sem certificados.

O primeiro artigo a usar explicitamente a idéia de certificação implícita foi [Gir91], onde Girault propôs a idéia de *chaves públicas auto-certificadas*. O resumo deste artigo diz[9]:

> *"Introduzimos a noção, e damos dois exemplos, de chaves públicas auto-certificadas, i.e. chaves públicas que não precisam ser acompanhadas de um certificado separado para serem autenticadas por outros usuários. O truque é fazer com que a chave pública seja calculada em conjunto pela autoridade confiável e pelo usuário, de maneira que o certificado esteja "embutido" na chave e, portanto, não precise ser enviado separadamente."*

Mais à frente, Girault também destaca o fato de que, ao contrário de esquemas baseados em identidade, a chave secreta aqui é de conhecimento exclusivo do usuário.

A idéia de esquemas com certificação implícita, portanto, é que o uso correto de um par de chaves implicitamente demonstra a sua autenticidade; ou seja, o fato de uma assinatura ser aceita prova simultaneamente que a chave pública utilizada na verificação é autêntica e que a assinatura é válida.

Para facilitar a discussão e tornar mais precisos os requisitos de segurança, Girault propôs classificar a segurança provida pelos diferentes esquemas em três níveis:

Nível-Girault 1. A autoridade confiável conhece, ou é capaz de facilmente calcular, a chave secreta dos usuários.

Nível-Girault 2. A autoridade confiável não conhece e não é capaz de calcular chaves secretas, mas ainda assim consegue personificar usuários gerando falsos "certificados" (ou chaves públicas) válidos.

Nível-Girault 3. Qualquer fraude por parte da autoridade confiável é detectável.

O nível de segurança 3 é o mais desejável, e é atingido pelos esquemas comuns baseados em certificação explícita. É importante notar que, mesmo nestes esquemas, a autoridade é capaz de agir de forma maliciosa e personificar usuários; o grande diferencial é que tais ataques podem ser detectados (pela existência de dois certificados válidos para uma mesma identidade) e a autoridade pode ser punida por seus atos. Os esquemas de chave pública auto-certificada que Girault apresenta vão também atingir nível 3 de segurança sem necessitar no entanto de certificação explícita: ele apresenta dois esquemas para gerar chaves auto-certificadas, baseados respectivamente no RSA e no ElGamal, e protocolos de identificação e estabelecimento de chaves simétricas utilizando estas chaves.

[9] Tradução livre do inglês.

2.3.1 Esquema de Girault para chaves auto-certificadas

Para exemplificar a abordagem de Girault, apresentaremos aqui o primeiro dos esquemas propostos por ele, baseado no RSA. Alice e Bob desejam estabelecer uma chave simétrica compartilhada; ambos confiam numa dada autoridade central que inicializa o sistema da seguinte forma:

`Inicialização.` Gere dois primos grandes, p e q. Seja $n = p \cdot q$. Gere um par de inteiros aleatórios, e e d, tais que $e \cdot d \equiv 1 \pmod{\phi(n)}$. Escolha ainda g um gerador de $\mathbb{Z}_n$. A chave pública da autoridade é (n, e, g) e a chave privada é d.

Ou seja, basicamente a autoridade confiável gera um par de chaves RSA. Agora, para gerar suas respectivas chaves públicas, Alice e Bob devem interagir com a autoridade da seguinte forma:

`Geração de Chaves.` O usuário (digamos, Alice) escolhe a chave secreta $s \xleftarrow{R} \mathbb{Z}_n^*$ e calcula $v = g^{-s} \mod n$. Alice envia v para a autoridade e executa uma prova de conhecimento zero para mostrar à autoridade que conhece o valor s (sem revelá-lo)[10]. A autoridade então calcula a chave pública de Alice como uma assinatura RSA da diferença entre v e sua identidade I:

$$P = (g^{-s} - I)^d \mod n.$$

Perceba que Alice pode validar sua chave verificando a equação

$$P^e + I \equiv g^{-s} \pmod{n}. \tag{2.1}$$

Este protocolo de geração de chaves tem algumas peculiaridades interessantes. A primeira delas é que, diferentemente de IBC ou criptografia de chave pública tradicional, a chave é gerada de maneira interativa, entre usuário e autoridade confiável; mais ainda, a geração de chaves públicas válidas (que verifiquem a equação (2.1)) só pode ser feita pela autoridade certificadora, pois envolve a geração de uma assinatura RSA. Sendo assim, a mera existência de duas chaves públicas válidas é evidência de comportamento indevido por parte da autoridade.

Chaves geradas desta forma podem ser utilizadas, por exemplo, para executar um estabelecimento de chaves similar ao método Diffie-Hellman, só que autenticado. Sejam (I, s, P) e (I', s', P') as informações de Alice e Bob respectivamente. Alice e Bob podem então calcular a chave compartilhada K da seguinte forma:

$$K = (P^e + I)^{s'} = (P'^e + I')^s \pmod{n}.$$

[10] No próprio artigo, Girault sugere um protocolo para fazer esta prova mas não abordaremos o protocolo aqui.

A vantagem é que somente Alice e Bob são capazes de calcular estas chaves, sem que certificados precisem ser explicitamente verificados.

O paradigma de certificação implícita de chaves públicas não visava trazer novas funcionalidades ao mundo da criptografia assimétrica: esperava-se meramente uma diminuição dos custos de diversos protocolos (pela ausência de certificação explícita) e uma simplificação dos modelos. Na próxima seção introduziremos a noção de Criptografia de Chave Pública sem Certificados, uma variação da idéia de certificação implícita que é o principal foco de estudo desta dissertação.

2.4 Criptografia de Chave Pública sem Certificados

A idéia de criptografia de chave pública sem certificados *(CL-PKC)* foi proposta por Al-Riyami e Paterson em [ARP03]. Seu principal objetivo foi atenuar o problema da custódia de chaves, presente nos sistemas de criptografia baseada em identidade *(IBC)*, mas ao mesmo tempo evitar a complexidade de administrar uma infraestrutura de chaves públicas completa. Em sistemas CL-PKC a autoridade confiável é conhecida como Centro de Geração de Chaves *(KGC)*[11]; mas, ao contrário da sua contraparte em esquemas baseados em identidade, a KGC não conhece as chaves privadas dos usuários por inteiro. Em CL-PKC, as chaves privadas dos usuários são formadas por dois componentes: uma *chave parcial* gerada pela KGC e relacionada à identidade do usuário; e um *valor secreto* gerado pelo próprio usuário e que ninguém mais deve conhecer. Ambas as componentes têm que ser conhecidas para que se possa decifrar mensagens e gerar assinaturas digitais.

Infelizmente, alguma forma de divulgação de chaves públicas torna-se necessária neste cenário, fazendo com que o sistema, na prática, não seja tão simples quanto um sistema de IBC. Mas, como no paradigma de chaves públicas auto-certificadas, não é necessária qualquer certificação explícita. Antes de falarmos de assinaturas sem certificados, principal foco deste trabalho, discutiremos um pouco criptossistemas de chave pública sem certificados, motivação inicial deste novo paradigma.

Definição 2. (Esquemas de ciframento de chave pública sem certificados). *Formalmente, um criptossistema de chave pública sem certificados é uma tupla de 7 algoritmos aleatorizados de tempo polinomial:*

> `Inicializar.` Este algoritmo é executado pela KGC para inicializar o sistema, recebendo um parâmetro de segurança 1^k como entrada e retornando um par de chaves-mestras, (mpk, msk). Implicitamente consideramos que a chave pública mpk contém descrições dos parâmetros do sistema, tais como a descrição dos espaços

[11] Do inglês *Key Generation Center*.

$\mathcal{M}$, de mensagens em claro, e $\mathcal{C}$, de textos cifrados; assumimos também que *mpk* está publicamente disponível de forma autenticada e que apenas a KGC conhece *msk*.

`Extrair chave privada parcial.` Este algoritmo é executado pela KGC, recebendo as chaves *msk* e *mpk*, além do identificador $ID_A \in \{0,1\}^*$ de uma entidade A, como entrada. Ele deve retornar a chave privada parcial D_A que deve ser entregue a A por um canal seguro.

`Definir valor secreto.` Este algoritmo é executado pela entidade A e recebe como entrada a chave-mestra pública *mpk* e o identificador ID_A da entidade A, e deve gerar o valor secreto x_A de A.

`Definir chave privada.` Este algoritmo é executado pela entidade A e recebe como entrada *mpk*, ID_A, a chave parcial D_A e o valor secreto x_A como entrada, e produz a chave privada completa S_A da entidade A.

`Definir chave pública.` Este algoritmo é executado pela entidade A e recebe como entrada *mpk*, ID_A, e o valor secreto da entidade A, x_A, como entrada e gera a chave pública P_A de A.

`Cifrar.` Este algoritmo recebe como entrada *mpk*, uma mensagem $M \in \mathcal{M}$, a identidade ID_A e a chave pública P_A de A e retorna um texto cifrado $C \in \mathcal{C}$ ou o símbolo $\perp$, indicando falha de ciframento.

`Decifrar.` Este algoritmo recebe como entrada *mpk*, um texto cifrado $C \in \mathcal{C}$, e a chave secreta S_A de A. Retorna uma mensagem $M \in \mathcal{M}$ ou o símbolo $\perp$, indicando falha no deciframento.

A grande motivação inicial para a proposta de CL-PKC foram esquemas de ciframento pois, neste cenário, algumas propriedades interessantes de criptografia baseada em identidades (notadamente, a capacidade de cifrar mensagens "para o futuro") poderiam ser mantidas, eliminando-se, no entanto, a custódia de chaves. Perceba que na Definição 2 o algoritmo de ciframento só recebe como entrada os parâmetros públicos do sistema, a identidade e a chave pública do destinatário e a mensagem a ser cifrada. Além disso, a definição da chave pública não depende de qualquer informação privada fornecida pela KGC. Sendo assim, podemos atrelar dados temporais às identidades dos usuários do sistema, e definir que a KGC só gerará as chaves privadas correspondentes no momento

adequado. Sendo assim, Alice poderia, por exemplo, cifrar uma mensagem para identidade "Bob‖14/10/2008", usando a chave pública de Bob, e Bob só receberia a chave privada adequada da KGC no dia 14 de outubro de 2008. Alice teria certeza, assim, de que a mensagem só poderia ser lida por Bob no dia desejado.

Veja que aqui a interação entre o usuário e a KGC ocorre na geração da chave privada, enquanto no esquema de Girault esta interação ocorria durante a geração da chave pública. Este fato tem duas conseqüências muito importantes:

1. Criptossistemas sem certificados permitem ciframento "para o futuro", como descrito anteriormente.

2. Protocolos de criptografia sem certificados, nesta forma canônica, só podem atingir o nível de segurança 2 de Girault, pois a chave pública dos usuários é completamente independente da KGC. Logo, um juiz deparado com a existência de duas chaves públicas "válidas" não é capaz de decidir quem agiu maliciosamente.

Já no artigo original Al-Riyami e Paterson perceberam este problema em potencial e propuseram a noção de encapsulamento de chave pública para elevar a segurança ao nível 3: pode-se considerar que chaves parciais são geradas somente depois que o usuário escolheu sua chave pública e criar uma dependência entre as duas. Por exemplo, pode-se considerar que a "identidade" do usuário é, na verdade, o hash da sua identidade verdadeira concatenada com o valor de sua chave pública: assim cada chave parcial estaria diretamente associada a uma única chave pública. Novas chaves públicas criadas pelo usuário não poderiam ser válidas (pois o usuário não conheceria a chave parcial correta) e, portanto, comportamento malicioso por parte da KGC poderia ser detectado.

Assinaturas sem certificados

Apesar do principal foco de CL-PKC estar no ciframento, já no primeiro artigo a delinear o paradigma [ARP03] os autores definem um esquema de assinaturas sem certificados (posteriormente mostrado inseguro). A definição formal de um esquema de assinaturas sem certificados *(CLS)*[12] é bastante parecida com a de criptossistemas e é dada abaixo.

Definição 3. (Esquemas de assinaturas de chave pública sem certificados). *Formalmente, um esquema de assinaturas de chave pública sem certificados é uma tupla de sete algoritmos aleatorizados de tempo polinomial, onde os cinco primeiros são idênticos aos da Definição 2, e os dois últimos são substituídos pelos seguintes:*

> `Assinar.` Este algoritmo recebe como entrada *mpk*, uma mensagem $M \in \mathcal{M}$, a identidade ID_A e a chave privada S_A de A e retorna uma assinatura $\sigma \in \mathcal{S}$.

`Verificar.` Este algoritmo recebe como entrada mpk, uma mensagem $M \in \mathcal{M}$, uma assinatura $\sigma \in \mathcal{S}$, a identidade ID_A e a chave pública P_A de A. Retorna `ACEITA`, `REJEITA` ou o símbolo $\perp$, indicando falha na verificação.

Naturalmente, esperamos que a verificação de uma assinatura corretamente gerada seja aceita. Perceba que não existe análogo à idéia de "ciframento para o futuro" no caso de assinaturas: para que uma assinatura seja gerada, o usuário tem que ter acesso à totalidade de suas informações secretas. Portanto, considerando a técnica de encapsulamento de chaves públicas, um esquema de assinaturas sem certificados tem, em princípio, funcionalidade equivalente a esquemas baseados na técnica de chaves auto-certificadas.

Uma definição equivalente de assinaturas sem certificados

Ao longo deste texto usaremos uma definição equivalente para esquemas de assinatura sem certificados, com cinco algoritmos em vez de sete. Esta simplificação da definição foi proposta em [HWZD06], onde os autores também mostraram que a nova definição é equivalente a definição original, de sete algoritmos.

`Inicializar.` Em geral este algoritmo é executado pela KGC para inicializar o sistema, recebendo como entrada um parâmetro de segurança 1^k e retornando um par de chaves-mestras, (mpk, msk). Implicitamente assumimos que a chave pública mpk contém descrições dos parâmetros do sistema, como a descrição dos espaços $\mathcal{M}$, de mensagens em claro, e $\mathcal{C}$, de textos cifrado; assumimos também que mpk está publicamente disponível de forma autenticada e que apenas a KGC conhece msk.

`Extrair chave privada parcial.` Este algoritmo é executado pela KGC, recebendo as chaves msk e mpk, além do identificador de uma entidade A, $ID_A \in \{0,1\}^*$, como entrada. Ele deve retornar a chave privada parcial D_A, que deve ser entregue a A por um canal seguro.

`Definir chaves do usuário.` Este algoritmo recebe mpk, ID_A, e gera o valor secreto x_A da entidade A, e sua respectiva chave pública P_A.

`Assinar.` Este algoritmo recebe como entrada mpk, uma mensagem $M \in \mathcal{M}$, a identidade ID_A e as chaves parcial, D_A, e secreta, x_A, de A e retorna uma assinatura $\sigma \in \mathcal{S}$.

`Verificar.` Este algoritmo recebe como entrada mpk, uma mensagem $M \in \mathcal{M}$, uma assinatura $\sigma \in \mathcal{S}$, a identidade ID_A e a chave pública P_A de A. Retorna `ACEITA`,

`REJEITA` ou o símbolo $\perp$, indicando falha na verificação.

2.5 Conclusão

Neste capítulo fizemos uma revisão dos conceitos mais relevantes à compreensão da seqüência de idéias que levou à criptografia de chaves públicas sem certificados, abordadas sob a ótica de certificação implícita de chaves públicas. Discutimos as idéias básicas por trás da criptografia de chave pública, como foi originalmente proposta por Diffie & Hellman [DH76], e destacamos alguns dos principais problemas envolvidos na sua utilização; especialmente, nos concentramos no problema da autenticidade de chaves públicas. Abordamos então algumas propostas que visam diminuir os custos envolvidos na certificação de chaves, relativizando o nível de liberdade que o usuário tem na escolha de suas chaves e o nível de confiança que deve ser depositada na autoridade central. Discutimos criptografia baseada em identidades (§2.2), a idéia de Girault para chaves públicas auto-certificadas (§2.3), chegando ao principal assunto deste trabalho, a criptografia de chave pública sem certificados (§2.4)

Capítulo 3

Assinaturas digitais sem certificados

Neste capítulo faremos uma revisão bastante completa da literatura de assinaturas digitais sem certificados. Como praticamente todos os esquemas propostos até o momento utilizam emparelhamentos bilineares, a próxima seção (§3.1) traz os fundamentos deste tema. Em seguida (§3.2) discutimos rapidamente todos os esquemas de CLS que foram propostos na literatura, tentando destacar as principais relações entre eles, sua eficiência relativa e se ainda são considerados seguros ou não.

3.1 Emparelhamentos bilineares

Curvas Elípticas foram propostas como uma opção viável para a implementação de criptografia em meados dos anos 80 [Mil85]. O grupo formado pelos pontos de uma curva, juntamente com uma operação de soma padrão, fornece a estrutura necessária para a instanciação do problema do logaritmo discreto e, conseqüentemente, para a adaptação de várias primitivas criptográficas baseadas na dificuldade deste problema.

Formalmente, uma curva elíptica sempre pode ser descrita por uma equação na forma de Weierstrass, $y^2 = x^3 + ax + b$. Os pontos da curva $E : F_q$ são a união das soluções $(x, y) \in F_q \times F_q$ para a equação, mais o ponto no infinito $\mathcal{O}$. Estes pontos formam um grupo sob uma operação de soma, onde o ponto especial $\mathcal{O}$ é o elemento neutro.

A grande vantagem na utilização de ECC (Criptografia de Curvas Elípticas)[1] está no maior nível de segurança que o DLP parece ter neste ambiente: enquanto algoritmos subexponenciais são conhecidos para o cálculo de logaritmos discretos em $\mathbb{Z}_p$, não se conhecem algoritmos mais eficientes que os "genéricos" (exponenciais) para grupos construídos sob curvas elípticas. Isto implica que grupos significativamente menores podem ser utilizados em ECC, levando a menores chaves e, em geral, a um ganho considerável de eficiência em

[1] Do inglês *Elliptic Curve Cryptography*.

comparação à utilização de $\mathbb{Z}_p$.

Em 1991 um pequeno golpe foi desferido contra a utilização de ECC: Menezes, Vanstone e Okamoto [MVO91] propuseram uma maneira de reduzir o DLP em um tipo particular de curvas elípticas (as chamadas *curvas super-singulares*) para o DLP numa extensão do corpo finito subjacente. Como a segurança do DLP na curva é significativamente maior do que no corpo, este ataque representava uma séria ameaça para criptossistemas baseados em ECC. Resumidamente o ataque se baseia na construção de um isomorfismo entre o subgrupo de pontos numa curva $E : F_q$ gerados por um P arbitrário, i.e. $< P >$, e o subgrupo das n-ésimas raízes da unidade de F_{q^k}, onde n denota a ordem de P. Para construir este isomorfismo os autores utilizam o emparelhamento de Weil.

Sendo assim o primeiro uso de emparelhamentos bilineares em criptografia foi "destrutivo": o emparelhamento de Weil foi proposto como parte de um ataque bastante preocupante a ECC. Percebeu-se, no entanto, que este ataque não se estende a curvas genéricas (pois o cálculo do emparelhamento de Weil não é polinomial no caso geral), servindo mais como uma advertência contra a utilização de curvas super-singulares em construções criptográficas baseadas em ECC. Levou-se bastante tempo, no entanto, até que alguém percebesse o potencial *construtivo* que a estrutura de emparelhamentos bilineares traz para a criptografia.

Somente em [Jou00] foi proposta a primeira aplicação construtiva de emparelhamentos bilineares, especificamente uma versão do protocolo de estabelecimento de chaves de Diffie-Hellman com três participantes. Percebeu-se assim o grande potencial trazido pelos emparelhamentos: pouco depois começaram a surgir propostas de criptossistemas baseados em identidade [BF01, SK03], assinaturas curtas [BLS01], agregadas [BGLS03], entre muitas outras estruturas que utilizavam emparelhamentos, e estes se tornaram um dos principais focos de pesquisa dos últimos anos.

Um *emparelhamento bilinear* $e : G_1 \times G_2 \rightarrow G_T$ é um mapa que leva um par ordenado de elementos em $G_1 \times G_2$ para um elemento do grupo G_T, com as três propriedades abaixo. Geralmente escrevemos os grupos G_1 e G_2 de forma aditiva com elemento neutro ∞, e G_T de forma multiplicativa com elemento neutro 1.

1. **Bilinearidade.** Para quaisquer $P, P_1, P_2 \in G_1$, e $Q, Q_1, Q_2 \in G_2$, temos que:

$$\begin{aligned} e(P_1 + P_2, Q) &= e(P_1, Q) \cdot e(P_2, Q), \\ e(P, Q_1 + Q_2) &= e(P, Q_1) \cdot e(P, Q_2). \end{aligned}$$

2. **Não Degeneração.** Se $\forall Q \in G_2, e(P, Q) = 1 \Rightarrow P = \infty$.
   Analogamente, se $\forall P \in G_1, e(P, Q) = 1 \Rightarrow Q = \infty$.

3. **Computabilidade.** Para todos $P \in G_1$, $Q \in G_2$ existe algoritmo eficiente que calcula $e(P, Q)$.

Adicionalmente, no caso especial em que $G_1 = G_2$, o emparelhamento é dito *simétrico*.

A bilinearidade dos emparelhamentos é uma propriedade que traz uma gama bastante interessante de possibilidades a serem exploradas pela criatividade dos pesquisadores: como já citamos, uma série de novas primitivas criptográficas foram propostas baseadas nesta construção matemática. Em primeiro lugar é necessário saber se existem emparelhamentos eficientemente calculáveis: citamos anteriormente o emparelhamento de Weil, mas devemos destacar também o emparelhamento de Tate. Para maiores informações sobre o cálculo eficiente de emparelhamentos bilineares referimos o leitor a [Sco05]. Nas próximas seções emparelhamentos bilineares são amplamente utilizados para construir esquemas de assinaturas digitais sem certificados.

3.2 Esquemas de assinatura sem certificados

Nesta seção faremos uma extensa revisão das propostas de esquemas de assinaturas digitais sem certificados presentes na literatura. Discutimos a segurança e a eficiência desses esquemas, e resumimos os resultados na tabela 3.1. Destacamos que esta não é uma comparação precisa, pois algumas outras operações (como cálculo de hash para pontos na extensão da curva) têm custos não-desprezíveis, mas é uma boa primeira classificação.

3.2.1 Al-Riyami & Paterson [2003]

No artigo onde Al-Riyami & Paterson [ARP03] propuseram a idéia de criptografia de chave pública sem certificados eles focaram principalmente na idéia de ciframento. Mas, ainda assim, eles apresentam um simples esquema de assinaturas descrito a seguir:

`Inicializar.`

Sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$;
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível;

seja $P \in \mathbb{G}$ um gerador arbitrário de $\mathbb{G}$; escolha as seguintes funções de hash:

$H_1 : \{0,1\}^* \rightarrow \mathbb{G}$; $H_2 : \{0,1\}^* \times \mathbb{G}_T \rightarrow \mathbb{Z}_p^*$;

escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;

retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Calcule $Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;

retorne a chave parcial $D_{ID_i} = sQ_{ID_i}$.

`GerarChavesUsuário`(ID_i).

escolha $x_{ID_i} \xleftarrow{R} \mathbb{Z}_p^*$; calcule $X_{ID_i} = x_{ID_i}P$ e $Y_{ID_i} = x_{ID_i}P_{pub}$;

retorne $\langle (X_{ID_i}, Y_{ID_i}), x_{ID_i} \rangle$.

`Assinar.`

Escolha $a \xleftarrow{R} \mathbb{Z}_p^*$; calcule $r = e(aP, P) \in \mathbb{G}_T$;

faça $h = H_2(M, r)$;

calcule $\sigma = (v \cdot x_{ID_i})D_{ID_i} + aP \in \mathbb{G}$;

retorne $\langle \sigma, h \rangle$.

`Verificar.`

Seja $\langle X_{ID_i}, Y_{ID_i} \rangle$ a chave pública do usuário;

verifica se $e(X_{ID_i}, P_{pub}) \stackrel{?}{=} e(Y_{ID_i}, P)$, abortando se forem diferentes;

seja $Q_{ID_i} = H_1(ID_i)$;

calcule $r = e(\sigma, P)e(Q_{ID_i}, -Y_{ID_i})^h$;

verifica se $h = H_2(M, r)$.

Se forem iguais, `ACEITA`;
caso contrário, `REJEITA`.

É um esquema bastante simples que compartilha os procedimentos de geração de chaves e de inicialização do sistema com o esquema de ciframento proposto no mesmo artigo. Uma característica interessante deste esquema é a necessidade de verificação da validade da chave pública durante a verificação de assinaturas ($e(X_{ID_i}, P_{pub}) \stackrel{?}{=} e(Y_{ID_i}, P)$); por causa destes dois emparelhamentos, o seu custo de verificação é de quatro emparelhamentos no total[2]. Muitos dos esquemas propostos após este mantêm esta estrutura de chaves públicas e incorrem em custos semelhantes de verificação. O objetivo desta verificação é "garantir" que o usuário conhece o valor secreto x_{ID_i} que deu origem a X_{ID_i} e Y_{ID_i}, pois supõe-se que é difícil calcular um par válido sem conhecer o x_{ID_i} correto. Isto, porém, não foi demonstrado.

Outra observação, e a grande causa da insegurança do esquema, é que nenhuma informação referente à chave pública do esquema está contida no hash ($h = H_2(M, r)$) que é assinado: este fato foi explorado por Huang et al. [HSMZ05] para propor um ataque. Informalmente, a vulnerabilidade do esquema vem do fato de que um adversário malicioso pode gerar uma "assinatura" aleatória e depois calcular qual chave pública torna a equação de verificação verdadeira para uma dada mensagem escolhida. O adversário então substitui a chave pública do usuário e o ataque terá sucesso. Mais especificamente, o adversário segue este procedimento para gerar assinaturas arbitrárias:

[2]Perceba que, possivelmente, esta é uma operação que pode ser feita uma única vez para cada nova chave pública, atenuando assim o seu custo.

1. Escolha $\sigma \xleftarrow{R} \mathbb{G}$;
2. calcule $R = e(\sigma, P)e(Q_{ID_i}, -P_{pub})$, onde $Q_{ID_i} = H_1(ID_i)$;
3. calcule $h = H_2(M, R)$;
4. seja $x_{ID_i} = h^{-1} \pmod{p}$;
5. calcule $X_{ID_i} = x_{ID_i}P$ e $Y_{ID_i} = x_{ID_i}P_{pub}$;
6. substitua a chave pública do usuário por $\langle X_{ID_i}, Y_{ID_i} \rangle$;
7. $\langle \sigma, h \rangle$ será então uma assinatura válida em M.

A assinatura gerada pelo procedimento acima é aceita pelo algoritmo de verificação, uma vez que as chaves públicas certamente são válidas e:

- Seja $R' = e(\sigma, P)e(Q_{ID_i}, -Y_{ID_i})^h$;
- teremos $h = H_2(M, R')$ pois

$$\begin{aligned} R' &= e(\sigma, P)e(Q_{ID_i}, -Y_{ID_i})^h \\ &= e(\sigma, P)e(Q_{ID_i}, -hx_{ID_i}P_{pub}) \\ &= e(\sigma, P)e(Q_{ID_i}, -hh^{-1}P_{pub}) \\ &= e(\sigma, P)e(Q_{ID_i}, -P_{pub}) \\ &= R. \end{aligned}$$

Em [HSMZ05], os autores propuseram utilizar o valor $e(x_{ID_i}D_{ID_i}, P) = e(Q_{ID_i}, Y_{ID_i})$ no hash do procedimento de assinatura (e, logicamente, de verificação). Sendo assim, h passaria a ser calculado como $h = H_2(M, R', e(Q_{ID_i}, Y_{ID_i}))$ e o ataque acima não poderia mais ser realizado. Isto torna a geração e a verificação de assinaturas mais ineficiente, uma vez que mais um emparelhamento deve ser calculado em cada etapa, mas os autores dão uma demonstração de segurança desta versão do esquema no artigo (o esquema original de Al-Riyami e Paterson não tinha qualquer demonstração de segurança).

Em [CD07b], nós mostramos que o motivo pelo qual a correção acima funciona é justamente forçar que qualquer adversário em potencial escolha a chave pública alvo *antes* de calcular a assinatura que pretende falsificar, porque qualquer mudança à chave pública altera de maneira imprevisível (por causa do hash) a assinatura. Nós mostramos ainda que, nestas condições, a técnica de reexecução de oráculos §4.3.2 pode ser utilizada para demonstrar a segurança do esquema; perceba que a demonstração de [HSMZ05] já utilizava a técnica sem justificar por que ela poderia ser utilizada no seu esquema, mas não no de [ARP03]. Falaremos mais sobre a técnica de reexecução de oráculos em §4.3; aqui apresentaremos apenas a versão final do esquema, como proposta em [CD07b].

Os algoritmos de inicialização e geração de chaves permanecem iguais[3]; mudamos apenas a geração e verificação de assinaturas:

[3]Bem, na verdade agora a função H_2 tem que ser definida como $H_2 : \{0,1\}^* \times \mathbb{G}_T \times \mathbb{G} \to \mathbb{Z}_p^*$.

`Assinar.`

Escolha $a \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;
calcule $r = e(aP, P) \in \mathbb{G}_T$;
faça $h = H_2(M, r, X_{ID_i})$;
calcule $\sigma = vx_{ID_i} D_{ID_i} + aP \in \mathbb{G}$;
retorne $\langle \sigma, h \rangle$.

`Verificar.`

Seja $\langle X_{ID_i}, Y_{ID_i} \rangle$ a chave pública do usuário;

verifica se $e(X_{ID_i}, P_{pub}) \stackrel{?}{=} e(Y_{ID_i}, P)$, abortando se forem diferentes;
seja $Q_{ID_i} = H_1(ID_i)$;
calcule $r = e(\sigma, P)e(Q_{ID_i}, -Y_{ID_i})^h$;
verifica se $h = H_2(M, r, X_{ID_i})$;

se forem iguais, `ACEITA`;
caso contrário, `REJEITA`.

Veja que esta versão do esquema mantém a mesma eficiência do esquema original, mas é demonstravelmente seguro [CD07b], §4.4.1.

3.2.2 Li, Chen & Sun [2005]

Em [LCS05] mais um esquema de assinaturas sem certificado foi proposto. Este esquema tem várias semelhanças com o Al-Riyami/Paterson original, inclusive um passo idêntico de verificação da chave pública, e requerendo 4 emparelhamentos para a verificação de assinaturas. O grande diferencial deste esquema, e a razão de ter sido proposto, é a possibilidade de se derivar um esquema de assinaturas delegáveis[4] a partir dele. Vejamos primeiro o esquema básico, omitindo os algoritmos de inicialização e geração de chaves, idênticos aos do Al-Riyami/Paterson.

`Assinar.`

Escolha $r \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;
calcule $U = rQ_{ID_i}$;
faça $h = H_2(M, U)$;
calcule $\sigma = (r + h)x_{ID_i} D_{ID_i} \in \mathbb{G}$;
retorne $\langle \sigma, U \rangle$.

`Verificar.`

[4] Do inglês *Proxy Signatures*.

Seja $\langle X_{ID_i}, Y_{ID_i} \rangle$ a chave pública do usuário;

verifica se $e(X_{ID_i}, P_{pub}) \stackrel{?}{=} e(Y_{ID_i}, P)$, abortando se forem diferentes;

seja $Q_{ID_i} = H_1(ID_i)$;

calcule $h = H_2(M, U)$;

verifique se $e(P, \sigma) = e(Y_{ID_i}, hQ_{ID_i} + U)$

Se forem iguais, `ACEITA`;
caso contrário, `REJEITA`.

No artigo original não foi dada qualquer prova de segurança do esquema. Em [CD07b] nós argumentamos que, utilizando os resultados apresentados no artigo (i.e., colocando a chave pública no hash de mensagem, fazendo $h = H_2(M, U, X_{ID_i})$), podemos provar que o esquema é seguro. Apresentaremos esta demonstração em §4.4.2, onde faremos uma revisão geral dos resultados conhecidos em relação à segurança demonstrável de esquemas CLS. Este esquema é de interesse, basicamente, por causa da possibilidade de derivar um esquema de assinaturas delegáveis, uma vez que sua versão simples é quase idêntica ao Al-Riyami/Paterson original.

3.2.3 Gorantla & Saxena [2005]

Em [GS05] Gorantla & Saxena propuseram o seguinte CLS:

`Inicializar`.

sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível.

Seja $P \in \mathbb{G}$ um gerador arbitrário de $\mathbb{G}$; escolha as seguintes funções de hash:

$H_1 : \{0,1\}^* \rightarrow \mathbb{G}$; $H_2 : \{0,1\}^* \times \mathbb{G} \rightarrow \mathbb{Z}_p^*$;

Escolha $s \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;

retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Calcule $Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;

retorne a chave parcial $D_{ID_i} = sQ_{ID_i}$.

`GerarChavesUsuário`(ID_i).

escolha $x_{ID_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $P_{ID_i} = x_{ID_i} P_{pub}$;

retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Seja $Q_{ID_i} = H_1(ID_i)$;
escolha $r \xleftarrow{R} \mathbb{Z}_p^*$;
calcule $U = rQ_{ID_i} + P_{pub} \in \mathbb{G}$
faça $h = H_2(M, U)$;
calcule $\sigma = (r + h)x_{ID_i}D_{ID_i}$;
retorne $\langle \sigma, U \rangle$.

`Verificar.`

Seja $Q_{ID_i} = H_1(ID_i)$;
calcule $h = H_2(M, U)$
`ACEITA` se e somente se

$$e(P, \sigma)e(P_{ID_i}, P_{pub}) = e(P_{ID_i}, U + hQ_{ID_i}).$$

Este esquema é baseado no IBS de Cha & Cheon [CC03] e foi uma primeira tentativa de diminuir a quantidade de emparelhamentos necessários para a verificação de assinaturas, de quatro para três. Na verdade, os autores não perceberam, mas como apontado em [CPK06] a verificação pode ser feita com apenas dois emparelhamentos, pois

$$e(P, \sigma)e(P_{ID_i}, P_{pub}) = e(P_{ID_i}, U + hQ_{ID_i}) \longleftrightarrow e(P, V) = e(P_{ID_i}, U + hQ_{ID_i} - P_{pub}). \quad (3.1)$$

Ainda como mostrado em [CPK06], a equação acima deixa a vulnerabilidade do esquema bastante clara: como não há um passo de verificação da validade da chave pública, não há certeza de que o usuário calculou $P_{ID_i} = xP_{pub}$ para um x conhecido: de fato, se ele calculou $P_{ID_i} = xP$, a equação (3.1) torna-se:

$$e(P, V) = e(P, x(U + hQ_{ID_i} - P_{pub})) \longrightarrow V = x(U + hQ_{ID_i} - P_{pub}).$$

Logo, forjar assinaturas torna-se trivial:

1. escolha $r, x \xleftarrow{R} \mathbb{Z}_p^*$;
2. calcule $P_{ID_i} = xP$;
3. calcule $U = rQ_{ID_i} + P_{pub}$;
4. calcule $h = H_2(M, U)$;
5. calcule $\sigma = x(U + hQ_{ID_i} + P_{pub})$.

Os autores de [CPK06] sugerem que adicionar um passo de verificação da chave pública como o do Al-Riyami/Paterson pode ser suficiente para deixar o esquema seguro, mas não fazem qualquer análise formal desta alegação. Mas, mesmo que o esquema torne-se seguro, o seu principal atrativo, a maior eficiência, seria perdido.

3.2.4 Yap, Heng & Goi [2006]

Este esquema, proposto em [YHG06], foi mais uma tentativa de diminuir o número de emparelhamentos necessários à verificação de assinaturas, de quatro para dois. Curiosamente, o esquema foi o primeiro CLS a contar com uma prova de segurança, mas foi quebrado pouco tempo depois, independentemente, em [Par06] e em [ZF06]. A inicialização e a geração de chaves do esquema são exatamente como no Gorantla/Saxena, mostrado na seção anterior. Para geração e verificação de assinaturas, os algoritmos propostos são os seguintes:

`Assinar.`
Seja $Q_{ID_i} = H_1(ID_i)$;
escolha $r \xleftarrow{R} \mathbb{Z}_p^*$;
calcule $U = rQ_{ID_i} \in \mathbb{G}$;
faça $h = H_2(M, U)$;
calcule $\sigma = (r + h)x_{ID_i}D_{ID_i}$;
retorne $\langle \sigma, U \rangle$.

`Verificar.`
Seja $Q_{ID_i} = H_1(ID_i)$;
calcule $h = H_2(M, U)$
`ACEITA` se e somente se

$$e(P, \sigma) = e(P_{pub} + P_{ID_i}, U + hQ_{ID_i}).$$

É fácil perceber a forte semelhança entre este esquema e o Gorantla/Saxena. Esperava-se, no entanto, que a existência de uma demonstração de segurança para o Yap/Heng/Goi implicasse a sua segurança; mas a prova está errada, como mostrou o ataque proposto por Park em [Par06] (essencialmente o mesmo que o proposto por Zhang e Feng em [ZF06]):

1. Escolha $x \xleftarrow{R} \mathbb{Z}_p^*$;
2. calcule a nova chave pública $P'_{ID_i} = xP - P_{pub}$;
3. para assinar qualquer mensagem M, calcule $\langle \sigma, U \rangle$ como segue:

$$U = rQ_{ID_i},\ h = H_2(M, U),\ \sigma = (r + h)xQ_{ID_i}.$$

Este procedimento gera uma falsificação válida. Discutiremos posteriormente, no capítulo dedicado à segurança de esquemas de CLS, quais exatamente são os problemas existentes na demonstração de segurança de [YHG06], mostrando dois grandes equívocos cometidos pelos autores que levam diretamente ao ataque mostrado acima.

3.2.5 Zhang, Wong, Xu & Feng [2006]

Neste artigo, [ZWXF06], os autores fazem uma análise um pouco mais elaborada da segurança de esquemas de assinatura sem certificados e propõem um CLS demonstravelmente seguro.

`Inicializar.`

sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível.

Seja $P \in \mathbb{G}$ um gerador arbitrário de $\mathbb{G}$; escolha as seguintes funções de hash:

$H_1, H_2, H_3 : \{0,1\}^* \rightarrow \mathbb{G}$;

Escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;

retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Calcule $Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;

retorne a chave parcial $D_{ID_i} = sQ_{ID_i}$.

`GerarChavesUsuário`(ID_i).

escolha $x_{ID_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $P_{ID_i} = x_{ID_i}P$;

retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Escolha $r \overset{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $U = rP \in \mathbb{G}_T$;

faça $h_2 = H_2(M, ID_i, P_{ID_i}, U)$, e $h_3 = H_2(M, ID_i, P_{ID_i})$;

calcule $\sigma = D_{ID_i} + rh_2 + xh_3 \in \mathbb{G}$;

retorne $\langle \sigma, h \rangle$.

`Verificar.`

Sejam $h_2 = H_2(M, ID_i, P_{ID_i}, U)$, e $h_3 = H_2(M, ID_i, P_{ID_i})$;

seja $Q_{ID_i} = H_1(ID_i)$;

`ACEITA` se e somente se

$$e(P, \sigma) = e(Q_{ID_i}, P_{pub})e(h_2, U)e(h_3, P_{ID_i}).$$

Este esquema também necessita de quatro emparelhamentos para a verificação de assinaturas mas é o primeiro CLS seguro a não usar a estrutura de chave públicas do Al-Riyami/Paterson, não necessitando portanto da verificação de validade da mesma.

Veja que, apesar de interessante já que propõe uma maneira diferente de construir chaves públicas, pode-se argumentar que este esquema é, na verdade, *menos* eficiente que o Al-Riyami/Paterson, pois o passo de verificação de chaves públicas daquele só precisa ser executado uma vez para cada chave, enquanto o ZDXF *sempre* precisa de quatro emparelhamentos.

3.2.6 Goya & Terada [2006]

Este esquema foi proposto em [Goy06] e é uma adaptação do IBS de Barreto et al. [BLMQ05] para a criptografia sem certificados. Ele representou um grande avanço pois, assim como o seu "pai" baseado em identidades, necessitava apenas de um emparelhamento para verificação de assinaturas. Infelizmente, em [CD07b] mostramos que o esquema é inseguro.

`Inicializar.`

sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível.

Seja $P \in \mathbb{G}$ um gerador de $\mathbb{G}$; escolha as funções de hash:

$H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$; $H_2 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_T \times \mathbb{G}_T \rightarrow \mathbb{Z}_p^*$;

calcule $g = e(P, P)$;

escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;

retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Seja $Q_{ID_i} = H_1(ID_i) \in \mathbb{Z}_p^*$;

retorne a chave parcial $D_{ID_i} = \frac{1}{Q_{ID_i}+s} P \in \mathbb{G}$.

`GerarChavesUsuário`(ID_i).

escolha $x_{ID_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $P_{ID_i} = g_{ID_i}^x \in \mathbb{G}_T$;

retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Escolha $r \overset{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $U = g^r \in \mathbb{G}_T$;

faça $h_2 = H_2(M, ID_i, P_{ID_i}, U)$;

calcule $\sigma = (r + hx_{ID_i}) D_{ID_i} \in \mathbb{G}$;

retorne $\langle \sigma, h \rangle$.

`Verificar.`

Seja $Q_{ID_i} = H_1(ID_i) \in \mathbb{Z}_p^*$;
seja $U' = e(\sigma, Q_{ID_i}P + P_{pub})(P_{ID_i})^{-h}$;
`ACEITA` se e somente se

$$h = H_2(M, ID_i, P_{ID_i}, U').$$

É fácil observar que este CLS é radicalmente diferente dos apresentados anteriormente: desde a geração de chaves parciais até a sua extrema eficiência, ele difere profundamente de tudo que foi apresentado até então. Infelizmente, ele é mais um exemplo de esquema que tem uma análise de segurança imprecisa e que foi posteriormente provado inseguro: em [CD07b] mostramos um ataque capaz de forjar mensagens arbitrárias.

1. Escolha $x_{ID_i} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ e calcule a nova chave pública:

$$P_{ID_i} = (e(P, P_{pub})g^{H_1(ID_i)})^{x_{ID_i}} = (g^s g^{H_1(ID_i)})^{x_{ID_i}} = g^{x_{ID_i}(s+H_1(ID_i))};$$

2. substitua a chave pública de ID_i por P_{ID_i};
3. seja M a mensagem cuja assinatura se quer forjar;
4. escolha $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$;
5. calcule $U = P_{ID_i}^r$;
6. calcule $h = H_2(M, ID_i, P_{ID_i}, U)$;
7. calcule $\sigma = (r + h)x_{ID_i}P$;
8. retorne $\langle \sigma, h \rangle$.

A falsificação obtida é válida, como observa-se a seguir:

$$\begin{aligned} U' &= e[\sigma, H_1(ID_i)P + P_{pub}](P_{ID_i})^{-h} \\ &= e[(r+h)x_{ID_i}P, (H_1(ID_i) + s)P]g^{x_{ID_i}(H_1(ID_i)+s)(-h)} \\ &= g^{(r+h)x_{ID_i}(H_1(ID_i)+s)}g^{-hx_{ID_i}(H_1(ID_i)+s)} \\ &= g^{rx_{ID_i}(H_1(ID_i)+s)} \\ &= U. \end{aligned}$$

No capítulo referente à segurança de CLS discutimos com mais detalhes os problemas na prova de segurança deste esquema, e que levam ao ataque proposto acima.

3.2.7 Liu, Au & Susilo [2006]

O esquema proposto por Liu, Au & Susilo em [LAS06] é uma adaptação do IBS de Paterson & Schuldt [PS06]. A grande característica peculiar de ambos os esquemas é o fato de serem demonstravelmente seguros no modelo padrão, sem precisar recorrer a

oráculos aleatórios. Por isso, é de se esperar que este esquema seja um pouco menos eficiente que as outras construções discutidas nesta seção.

`Inicializar.`

Sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$;
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível;

seja $P \in \mathbb{G}$ um gerador de $\mathbb{G}$; escolha as funções de hash resistentes a colisões:

$H_u : \{0,1\}^* \rightarrow \{0,1\}^{n_u}$; $H_m : \mathbb{G} \rightarrow \{0,1\}^{n_m}$,

onde $n_u, n_m \in \mathbb{Z}$ são parâmetros do sistema;
escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;
escolha $Q \overset{R}{\leftarrow} \mathbb{G}$;
escolha $u', m' \overset{R}{\leftarrow} \mathbb{G}$;
escolha $\hat{u}_i \overset{R}{\leftarrow} \mathbb{G}$, para $i = 1, \ldots, n_u$;
escolha $\hat{m}_i \overset{R}{\leftarrow} \mathbb{G}$, para $i = 1, \ldots, n_m$;
sejam $\hat{U} = \{\hat{u}_i\}$ e $\hat{M} = \{\hat{m}_i\}$;
retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub}, Q, u', \hat{U}, m', \hat{M} \rangle$ e $msk = sQ$.

`ExtrairChaveParcial`(ID_i).

Seja $Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;
retorne a chave parcial $D_{ID_i} = sQ_{ID_i} \in \mathbb{G}$.

`GerarChavesUsuário`(ID_i).

escolha $x_{ID_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$;
calcule $P_{ID_i} = x_{ID_i} P \in \mathbb{G}$;
retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Seja $y_{ID_i} = H_2(P_{ID_i})$;
calcule $S_{ID_i} = \frac{1}{x_{ID_i} + y_{ID_i}} D_{ID_i}$;
escolha $r \overset{R}{\leftarrow} \mathbb{Z}_p^*$;
calcule $U = rQ_{ID_i} \in \mathbb{G}$;
faça $h = H_3(M, U)$;
calcule $\sigma = (r + h) S_{ID_i} \in \mathbb{G}$;
retorne $\langle \sigma, h \rangle$.

`Verificar.`

Sejam:

$Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;

$y_{ID_i} = H_2(P_{ID_i})$;
$h = H_3(M, U)$;

`ACEITA` se e somente se

$$e(\sigma, P_{ID_i} + y_{ID_i}P) = e(U + hQ_{ID_i}, P_{pub}).$$

3.2.8 Choi, Park, Hwang & Lee [2007]

No artigo [CPHL07], os autores propõem dois novos esquemas de assinatura sem certificados, ambos extremamente eficientes. As demonstrações de segurança destes esquemas, no entanto, sofrem dos problemas expostos em [CD07b] o que torna a sua robustez, atualmente, uma questão em aberto. O primeiro dos esquemas requer o cálculo de dois emparelhamentos para a verificação de assinaturas e está descrito a seguir:

`Inicializar.`

Sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$;
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível;

seja $P \in \mathbb{G}$ um gerador de $\mathbb{G}$; escolha as funções de hash:

$H_1 : \{0,1\}^* \rightarrow \mathbb{G}$; $H_2 : \mathbb{G} \rightarrow \mathbb{Z}_p^*$; $H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$;

escolha $s \xleftarrow{R} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;
retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Seja $Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;
retorne a chave parcial $D_{ID_i} = sQ_{ID_i} \in \mathbb{G}$.

`GerarChavesUsuário`(ID_i).

Escolha $x_{ID_i} \xleftarrow{R} \mathbb{Z}_p^*$;
calcule $P_{ID_i} = x_{ID_i}P \in \mathbb{G}$;
retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Seja $y_{ID_i} = H_2(P_{ID_i})$;
calcule $S_{ID_i} = \frac{1}{x_{ID_i} + y_{ID_i}} D_{ID_i}$;
escolha $r \xleftarrow{R} \mathbb{Z}_p^*$;
calcule $U = rQ_{ID_i} \in \mathbb{G}$;
faça $h = H_3(M, U)$;
calcule $\sigma = (r + h)S_{ID_i} \in \mathbb{G}$;
retorne $\langle \sigma, h \rangle$.

`Verificar.`

Sejam:

$Q_{ID_i} = H_1(ID_i) \in \mathbb{G}$;
$y_{ID_i} = H_2(P_{ID_i})$;
$h = H_3(M, U)$;

`ACEITA` se e somente se

$$e(\sigma, P_{ID_i} + y_{ID_i}P) = e(U + hQ_{ID_i}, P_{pub}).$$

Este esquema diverge fortemente dos outros esquemas propostos anteriormente, especialmente pela maneira como a "chave privada completa" S_{ID_i} é calculada, baseada em uma das versões das assinaturas curtas de Boneh & Boyen [BB04]. Os autores propõem ainda um segundo esquema, este necessitando apenas de um emparelhamento para a verificação de assinaturas:

`Inicializar.`

Sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$
$\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível;

seja $P \in \mathbb{G}$ um gerador de $\mathbb{G}$; escolha as funções de hash:

$H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$; $H_2 : \mathbb{G} \rightarrow \mathbb{Z}_p^*$; $H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$;

escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;

retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Seja $q_{ID_i} = H_1(ID_i) \in \mathbb{Z}_p^*$;

retorne a chave parcial $D_{ID_i} = \frac{1}{s+q_{ID_i}} P \in \mathbb{G}$.

`GerarChavesUsuário`(ID_i).

Seja $q_{ID_i} = H_1(ID_i)$;

calcule $Q_{ID_i} = P_{pub} + q_{ID_i}P$;

escolha $x_{ID_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $P_{ID_i} = x_{ID_i} Q_{ID_i} \in \mathbb{G}$;

retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Seja $y_{ID_i} = H_2(P_{ID_i})$;

calcule $S_{ID_i} = \frac{1}{x_{ID_i}+y_{ID_i}} D_{ID_i}$;

escolha $r \overset{R}{\leftarrow} \mathbb{Z}_p^*$;

calcule $U = g^r = e(P, P)^r \in \mathbb{G}_T$;
faça $h = H_3(M, U)$;
calcule $\sigma = (r + h)S_{ID_i} \in \mathbb{G}$;
retorne $\langle \sigma, h \rangle$.

`Verificar.`

Sejam:

$Q_{ID_i} = P_{pub} + H_1(ID_i)P \in \mathbb{Z}_p^*$;
$y_{ID_i} = H_2(P_{ID_i})$;
$h = H_3(M, U)$;

`ACEITA` se e somente se

$$e(\sigma, P_{ID_i} + y_{ID_i} Q_{ID_i}) = U g^h.$$

A diferença entre esta segunda versão do esquema e a primeira é que aqui as assinaturas Boneh/Boyen também são usadas para a geração da chave privada parcial.

3.2.9 Du & Wen [2007]

O esquema de Du & Wen [DW07] é bastante similar ao esquema mais eficiente de Choi, Park, Hwang & Lee, mas levemente simplificado. A segurança de ambos os esquemas parece estar relacionada, e este também tem o mesmo problema na demonstração apontado em [CD07b] e discutido em §4.2.1 desta dissertação.

`Inicializar.`

Sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

$p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$
$\exists\, e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ emparelhamento bilinear admissível;

seja $P \in \mathbb{G}$ um gerador de $\mathbb{G}$; escolha as funções de hash:

$H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$; $H_2 : \{0,1\}^* \times \mathbb{G} \to \mathbb{Z}_p^*$;

escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;
retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).

Seja $q_{ID_i} = H_1(ID_i) \in \mathbb{Z}_p^*$;
retorne a chave parcial $D_{ID_i} = \frac{1}{s + q_{ID_i}} P \in \mathbb{G}$.

`GerarChavesUsuário`(ID_i).

Seja $q_{ID_i} = H_1(ID_i)$;
calcule $Q_{ID_i} = P_{pub} + q_{ID_i} P$;

escolha $x_{ID_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;
calcule $P_{ID_i} = x_{ID_i} Q_{ID_i} \in \mathbb{G}$;
retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`

Seja $h = H_2(M, P_{ID_i})$;
calcule $\sigma = \frac{1}{(x_{ID_i}+h)} D_{ID_i} \in \mathbb{G}$;
retorne $\langle \sigma \rangle$.

`Verificar.`

Sejam:

$$Q_{ID_i} = P_{pub} + H_1(ID_i)P \in \mathbb{Z}_p^*;$$
$$h = H_2(M, P_{ID_i});$$

`ACEITA` se e somente se

$$e(\sigma, P_{ID_i} + hQ_{ID_i}) = g.$$

3.2.10 Comparação dos esquemas

A tabela 3.1 apresenta uma comparação resumida da performance dos esquemas apresentados.

Esquema	Custo/Assinatura	Custo/Verificação	Status
Al-Riyami & Paterson [ARP03]	1	4	*Quebrado*
Al-Riyami & Paterson 2 [HSMZ05]	2	5	**OK**
Al-Riyami & Paterson 3 [CD07b]	1	4	**OK**
Li, Chen & Sun [LCS05]	0	4	**OK**
Gorantla & Saxena [GS05]	0	2	*Quebrado*
Yap, Heng & Goi [YHG06]	0	2	*Quebrado*
Zhang et al. [ZWXF06]	0	4	**OK**
Goya & Terada [Goy06]	0	1	*Quebrado*
Liu, Au & Susilo [LAS06]	0	6	**OK**
Choi et al. - 1 [CPHL07]	0	1	*Incerto*
Choi et al. - 2 [CPHL07]	0	2	*Incerto*
Du & Wen [DW07]	0	1	*Incerto*

Tabela 3.1: Comparação de performance entre esquemas de assinatura sem certificado

3.3 Conclusão

Este capítulo fez uma revisão de emparelhamentos bilineares (§3.1), conceito matemático fundamental na construção de praticamente qualquer esquema de criptografia de chave

pública sem certificados, seguida de uma revisão bibliográfica bastante extensa (§3.2), cobrindo todos os esquemas de assinaturas digitais sem certificados de que temos conhecimento. Tentamos apresentar esta revisão bibliográfica de maneira a destacar as principais semelhanças entre os diversos esquemas e a sua eficiência relativa. No próximo capítulo, onde tratamos especificamente da segurança de esquemas de CLS, faremos uma revisão mais aprofundada das propriedades de segurança destes esquemas.

Capítulo 4

Segurança de assinaturas digitais sem certificados

Neste capítulo estudaremos a segurança de esquemas de assinatura sem certificados, revisando e aprimorando modelos apropriados para as peculiaridades deste paradigma. Iniciaremos com um histórico da área de "Segurança Demonstrável" (§4.1), onde pretendemos destacar a importância crucial que esta representa para o avanço da criptografia e apresentar, situando historicamente, as principais ferramentas que usaremos na discussão que segue. Na seqüência (§4.2) discutiremos especificamente modelos e formalizações para a criptografia de chave pública sem certificados nos aprofundando naqueles que nos permitem analisar os esquemas de assinaturas apresentados no restante deste documento. A seção seguinte (§4.3) é dedicada a uma análise mais completa da Técnica de Reexecução de Oráculos. Encerramos o capítulo com uma revisão dos esquemas apresentados no capítulo anterior, agora focando nas propriedades de segurança dos mesmos (§4.4).

4.1 Segurança de assinaturas digitais

Nesta seção faremos uma breve retrospectiva dos principais conceitos relevantes ao desenvolvimento da área de segurança demonstrável dentro da criptografia.

4.1.1 Shannon e a Teoria da Informação

Historicamente, a criptografia era a *arte* de projetar protocolos e algoritmos que provessem segurança a um conjunto de entidades em suas comunicações. O primeiro e fundamental passo dado para formalizar esta "arte" veio com Shannon e sua análise de criptossistemas (simétricos) baseada na Teoria da Informação [Sha49]. Com este trabalho Shannon formalizou pela primeira vez o que significava um criptossistema ser "seguro" e analisou em

que condições isso seria possível. Este trabalho levou, entre outras coisas, à definição da noção de *sigilo perfeito*, que traduz formalmente a intuição de que um texto cifrado não deve fornecer *qualquer* informação sobre a mensagem original que foi cifrada.

Definição 4. (Sigilo Perfeito, Segurança Incondicional) *Um criptossistema possui sigilo perfeito se e somente se* $\Pr[y|m] = \Pr[y]$ *para toda mensagem* m *e todo texto cifrado* y*, i.e.,* $\Pr[y|m]$ *deve ser independente de* m*.*

Esta certamente é uma definição *forte* de segurança, exigindo que a distribuição de textos cifrados seja completamente independente da distribuição das mensagens: toda mensagem é igualmente provável de ser cifrada como qualquer texto cifrado válido. Shannon provou ainda que, para que o sigilo perfeito seja atingido, chaves tão grandes quanto a mensagem a ser cifrada têm que ser utilizadas, e que, em tal situação, mesmo o simples *one-time pad*[1] provê sigilo perfeito.

4.1.2 Criptografia assimétrica e noções fortes de segurança

Infelizmente as idéias de Shannon não são diretamente aplicáveis ao campo da criptografia assimétrica [DH76]: em um criptossistema assimétrico, os parâmetros públicos *sempre* carregam informação suficiente para quebrar todo o esquema; a segurança do criptossistema se baseia na suposição de que computar esta informação seja difícil. Em outras palavras, a chave pública sempre determina univocamente a chave privada e por isso o segredo perfeito não pode ser atingido neste cenário. O que se busca então é a noção de *segurança computacional*, e estudar a segurança de criptossistemas de chave pública quer dizer estudar o custo computacional de quebrar o sistema. A ferramenta mais apropriada encontrada para analisar esta noção de "custo computacional" foi a Teoria da Complexidade e, com base em idéias como complexidade assintótica e reduções polinomiais entre problemas, um trabalho de fundamentação da teoria da criptografia foi realizado ao longo dos anos 80 (e.g. em [Yao82], [GM84], [GMR88] e [MRS88]). Um dos conceitos mais seminais desenvolvidos ao longo destes trabalhos foi o de noções fortes de segurança que adaptavam a idéia de segredo perfeito para o cenário onde os adversários têm poder computacional limitado. As duas mais importantes, e que sobrevivem até hoje, foram propostas por Goldwasser & Micali em [GM82]:

Definição 5. (Segurança Semântica) *Seja* f *uma função definida no espaço de mensagens. Informalmente,* $f(m)$ *representa informação sobre* m*. Sejam os adversários* A_i *máquinas de Turing probabilísticas de tempo polinomial. Sejam os três jogos a seguir:*

[1] Para cifrar utilizando o *one-time pad* o usuário calcula simplesmente o ou-exclusivo da mensagem com a chave: se a chave for do tamanho da mensagem esta operação provê sigilo perfeito; senão, a mesma chave tem que ser repetida várias vezes para cifrar a mensagem inteira e o esquema torna-se potencialmente inseguro.

- ***Jogo 1.*** *Escolha* $m \xleftarrow{r} M$. *Neste jogo,* A_1 *tem que calcular o valor de* $f(m)$ *sem conhecer* m.
- ***Jogo 2.*** *Escolha* $m \xleftarrow{r} M$. *Calcule um* $\alpha \leftarrow E(m)$ *e forneça ao adversário. Neste jogo,* A_2 *tem que calcular o valor de* $f(m)$ *conhecendo* $E(m)$.
- ***Jogo 3.*** *Deixe* A_3 *escolher uma função* f^* *definida em* M. *Escolha* $m \xleftarrow{r} M$, *calcule um* $\alpha \leftarrow E(m)$ *e forneça ao adversário.* A_3 *tem que calcular o valor de* $f^*(m)$.

Seja A_i^* *o evento em que o adversário vence o jogo* i. *Um criptossistema* $\mathcal{C}$ *é semanticamente seguro se*

$$\Pr[A_3^*] < \Pr[A_1^*] + k^{-c},$$

ou seja, o conhecimento da mensagem cifrada e a possibilidade de escolher a função que se quer calcular não devem trazer vantagem significativa para o adversário.

A noção de segurança semântica traduz o fato de que deveria ser "difícil", para tais adversários polinomialmente limitados, calcular o valor de qualquer $f(m)$ dado o texto cifrado $E(m)$: é uma tradução do conceito de "sigilo perfeito" de Shannon para o ambiente onde todos os participantes estão limitados por um número polinomial de passos.

Definição 6. (Segurança Polinomial) *Sejam* A_i *máquinas de Turing de tempo polinomial. Definimos então o seguinte jogo:*

1. $(m_0, m_1) \leftarrow A_1(params)$;
2. $i \xleftarrow{R} \{0,1\}$; $C \leftarrow E_{pk}(m_i)$;
3. $i' \leftarrow A_2(params, m_0, m_1, C)$,

onde denotamos por $E_k(m)$ *o ciframento da mensagem* m *utilizando a chave (pública)* pk. *Seja* A^* *o evento em que o adversário calcula corretamente* i, *i.e.,* $i' = i$. *O criptossistema* $\mathcal{C}$ *é* polinomialmente seguro *se, para todos os adversários* $\mathcal{A} = (A_1, A_2)$ *e todo* $c > 0$,

$$\Pr[A^*] < \frac{1}{2} + k^{-c}.$$

A nocão de Segurança Polinomial também é conhecida por *Indistinguibilidade de Textos Cifrados*. Esta definição de segurança diz que, dado um par de mensagens e uma delas cifrada, um adversário não deve ser capaz de distinguir, em tempo polinomial, a qual das mensagens em claro o texto cifrado corresponde.

As duas definições de segurança são bastante fortes e poderia restar a dúvida de qual é a mais apropriada: a mais intuitiva (segurança semântica) ou a mais fácil de se definir e utilizar (segurança polinomial). Felizmente, os resultados de [GM82] e [MRS88] implicam que ambas as definições são equivalentes.

4.1.3 Noções fortes de segurança para esquemas de assinatura

Seguindo-se aos primeiros trabalhos de segurança demonstrável que tratavam de criptossistemas, o trabalho seminal que trata da segurança de esquemas de assinatura veio em [GMR88], onde são definidos os principais modelos de segurança para assinaturas e um esquema seguro no mais forte destes modelos é proposto. Revisaremos parcialmente aqui a taxonomia proposta neste trabalho.

Pode-se distinguir inicialmente dois tipos básicos de ataque:

- **Ataques Apenas-com-Chave,** onde o adversário conhece apenas a chave pública do usuário;
- **Ataques de Mensagem,** onde o adversário tem a possibilidade de examinar assinaturas correspondentes a mensagens conhecidas ou escolhidas por ele, de maneira a facilitar a quebra do esquema.

Subdivide-se ainda os ataques de mensagem de acordo com o controle que o adversário tem sobre a escolhas das mensagens:

- **Ataques de Mensagem Conhecida.** O adversário recebe as assinaturas de uma lista de mensagens $\mathbb{M} = m_0, m_1, \ldots, m_n$, mas não tem direito de escolher as mensagens na lista.
- **Ataques Genéricos de Mensagem Escolhida.** Aqui o adversário também recebe assinaturas referentes a uma lista de mensagens $\mathbb{M}$ mas ele tem o direito de escolher as mensagens que fazem parte da lista *de forma independente da chave pública do usuário alvo do ataque,* ou seja, de forma genérica.
- **Ataques Dirigidos de Mensagem Escolhida.** Ataques semelhantes aos genéricos, onde o adversário pode escolher as mensagens que fazem parte de $\mathbb{M}$ com base na chave pública do usuário-alvo, ou seja, o ataque é dirigido a um certo usuário U^*.
- **Ataques Adaptativos de Mensagem Escolhida.** Este é o tipo mais geral de ataque, onde o adversário pode utilizar o usuário U^* como um oráculo, obtendo dele assinaturas em mensagens arbitrárias escolhidas ao longo do ataque de forma adaptativa, ou seja, o adversário pode requisitar assinaturas em mensagens que dependem de respostas anteriores de U^*.

Fica claro pela classificação acima que o tipo mais "poderoso" de ataque é o adaptativo; logo, seria *desejável* obter esquemas de assinaturas seguros até contra adversários deste tipo.

A próxima classificação proposta por Goldwasser, Micali & Rivest está relacionada com a seguinte questão: o que significa exatamente quebrar um esquema de assinaturas?

- **Quebra Total.** O adversário consegue calcular a informação secreta do usuário U^*.
- **Falsificação Universal.** O adversário consegue encontrar um algoritmo de assina-

tura equivalente ao de U^*, mesmo sem conseguir recuperar as informações secretas deste.

- **Falsificação Seletiva.** O advesário consegue falsificar a assinatura de uma mensagem escolhida *a priori*.
- **Falsificação Existencial.** O adversário consegue falsificar a assinatura de pelo menos uma mensagem, ainda que não consiga ter qualquer controle sobre que mensagem é assinada (pode ser uma mensagem aleatória ou que não faça qualquer sentido).

Percebe-se então que a *gravidade* do ataque diminui de uma quebra total para uma falsificação existencial e que um esquema *existencialmente seguro*[2] é seguro em todos os outros sentidos.

Baseado nesta taxonomia, a noção padrão de segurança para um esquema de assinaturas é, desde então, a de um esquema *existencialmente inforjável sob ataque adaptativo de mensagem escolhida* (EU-CMA) [3].

Definição 7. (EU-CMA). *Um esquema de assinaturas $\mathcal{S}$ é* existencialmente inforjável sob ataque adaptativo de mensagem escolhida *se qualquer adversário $\mathcal{A}$ tem chance desprezível de vencer o seguinte jogo.*

1. *O desafiante $\mathcal{D}$ gera os parâmetros do sistema* params *e a chave pública PK^* do usuário-alvo U^*;*

2. *$\mathcal{D}$ executa $\mathcal{A}$ com $(params, PK^*)$ como entrada;*
   - *$\mathcal{A}$ tem acesso a um oráculo $\mathcal{O}^S$ que calcula assinaturas em mensagens arbitrárias sob a chave pública PK^*.*

3. *Após um tempo polinomial, $\mathcal{A}$ retorna um par (σ, M).*

$\mathcal{A}$ vence o jogo se o algoritmo de verificação do esquema de assinaturas aceita σ como assinatura de U^* em M.

Usaremos esta noção de segurança daqui para a frente, apenas adaptando o jogo acima a diferentes cenários, especificamente à criptografia de chave pública sem certificados.

[2] Seguro contra falsificações existenciais.
[3] Do inglês *existentially unforgeable under chosen-message attack*

4.1.4 O paradigma do oráculo aleatório

O paradigma do oráculo aleatório é uma heurística para a análise da segurança de protocolos criptográficos proposta por Bellare & Rogaway em [BR93]. Desde então, análises de segurança no modelo do oráculo aleatório tornaram-se bastante difundidas e o seu significado cada vez mais controverso. O paradigma proposto por Bellare e Rogaway consiste em, dado um protocolo $\mathcal{P}$ cuja segurança deseja-se analisar, estudar o protocolo $\mathcal{P}^{\mathcal{O}}$ onde todos os participantes têm acesso a um oráculo $\mathcal{O}$ que se comporta como uma função aleatória[4]. Os autores argumentam então que, caso o protocolo $\mathcal{P}^{\mathcal{O}}$ possa ser demonstrado seguro neste "modelo do oráculo aleatório", o protocolo $\mathcal{P}^{H}$, onde invocações ao oráculo $\mathcal{O}$ são substituídas por chamadas à função H cuja descrição é pública, pode ser considerado seguro *para uma escolha adequada da função* H. Seguindo este paradigma então, os passos para se projetar um esquema de assinatura seriam:

1. projeta-se um esquema $\mathcal{S}^{\mathcal{O}}$ contando com a existência de um oráculo aleatório;
2. demonstra-se que $\mathcal{S}^{\mathcal{O}}$ é seguro (no modelo do oráculo aleatório) em relação a uma certa noção de segurança, p.e. demonstra-se que $\mathcal{S}^{\mathcal{O}}$ é EU-CMA.
3. define-se então o esquema $\mathcal{S}^{H}$, onde invocações a $\mathcal{O}$ são simuladas pela função H, de conhecimento público (provavelmente derivada de uma função de hash).

Bellare e Rogaway argumentam então que, dada uma boa escolha de H, a segurança de $\mathcal{S}^{H}$ segue da segurança de $\mathcal{S}^{\mathcal{O}}$.

O problema é que certamente o modelo do oráculo aleatório não é uma descrição "fiel" da realidade: de fato, as simplificações que propõe são irrealizáveis pois, por "melhor" que seja a função H, ela nunca poderá se comportar como uma função aleatória; por outro lado, costuma-se argumentar que uma prova de segurança, ainda que num modelo simplificado, é melhor que nenhuma prova de segurança e, para a maioria das situações os protocolos mais eficientes que se consegue são demonstravelmente seguros apenas no modelo oráculo aleatório.

Acreditava-se, no entanto, que esta distância entre o modelo do oráculo aleatório e o "modelo padrão" não poderia ser explorada de maneira a gerar vulnerabilidades. Essa crença foi derrubada em [CGH98], onde os autores provam que nenhuma família de funções pode ser uma "boa" implementação de um oráculo aleatório: eles dão exemplos de um criptossistema e um esquema de assinaturas demonstravelmente seguros no modelo do oráculo aleatório, porém inseguros quando instanciados com qualquer família de funções. Estes exemplos são bastante artificiais, mas ainda assim serviram como um alerta para a comunidade: depois deste resultado, outros ([CGH04], [BBP04], [HK07]) se seguiram,

[4]Uma função escolhida uniformemente dentre todas as possíveis funções de tamanho apropriado.

sendo que este último apresenta uma "separação"[5] entre o modelo do oráculo aleatório e o modelo padrão bastante realista.

A tendência dos pesquisadores tem sido então pensar no paradigma do oráculo aleatório como um passo inicial no projeto de protocolos para problemas novos e pouco estudados: em última instância deve-se tentar obter esquemas seguros no modelo padrão, mas os primeiros esquemas propostos para um novo cenário podem ser construídos no modelo do oráculo aleatório, até que se obtenha conhecimento suficiente sobre o problema para que se possa tentar construir esquemas seguros no modelo padrão. De fato é isto que vem acontecendo com a criptografia de chave pública sem certificados: a demonstração de segurança do esquema proposto aqui será feita no modelo do oráculo aleatório, mas alguns esquemas de CLS seguros no modelo padrão já começaram a surgir, sendo o mais eficiente deles ([LAS06]) aproximadamente duas vezes mais lento que o esquema proposto neste trabalho.

4.1.5 Demonstrações por seqüências de jogos

Demonstrações de segurança de esquemas criptográficos podem se tornar bastante complexas, especialmente quando se prova segurança contra adversários ativos/adaptativos. Isto é um problema grave pois torna estas demonstrações ao mesmo tempo difíceis de verificar e propensas a erros.

Provavelmente o caso mais célebre de um erro numa tal demonstração foi encontrado por Victor Shoup [Sho01] na demonstração de segurança do criptossistema OAEP proposto por Bellare e Rogaway em [BR94]. Este caso é particularmente emblemático porque o OAEP foi um resultado bastante importante, sendo incluído em diversos padrões nos anos subseqüentes à sua proposta, em grande parte por ser "demonstravelmente seguro". Ainda assim passaram-se sete anos até que Shoup descobrisse um erro na sua "demonstração" de segurança[6].

Isto mostra o quanto demonstrações de segurança de esquemas criptográficos, geralmente apresentadas como jogos, podem ser difíceis de verificar; maneiras de diminuir esta dificuldade de verificação vêm sendo propostas nos últimos anos e neste trabalho utilizamos a idéia de apresentar as demonstrações como *seqüências de jogos*. A idéia aqui, apresentada mais profundamente em [Sho04] e [BR04], é definir a segurança do nosso esquema primeiro como um jogo *ideal*, que claramente reflete a noção de segurança em relação à qual construímos a demonstração. Em seguida apresentamos mudanças sim-

[5]Uma separação entre o modelo do oráculo aleatório *(ROM)* e o modelo padrão é um esquema que é demonstravelmente seguro no ROM mas é inseguro no modelo padrão.

[6]Por coincidência, o RSA-OAEP, a versão do esquema mais largamente utilizada na prática, pode ser demonstrado seguro devido a propriedades específicas da função RSA. Mas, em geral, o OAEP não é uma primitiva segura.

ples e facilmente justificáveis a este jogo, que mantêm as probabilidades de sucesso e os tempos de execução dos algoritmos relacionados aos do jogo original. Geramos assim uma seqüência de jogos, cada um relacionado ao anterior por uma mudança simples e facilmente verificável, de maneira que o jogo final desta seqüência descreve um ataque ao nosso esquema: fica assim mais fácil de verificar a relação entre um ataque ao esquema que estamos analisando e um ataque a um esquema ideal pois cada passo que transforma um no outro pode ser analisado individualmente.

4.2 Segurança em criptografia de chave pública sem certificados

A grande peculiaridade da criptografia de chave pública sem certificados, em relação aos tipos tradicionais de ataque, é que em CL-PKC as chaves públicas não são explicitamente certificadas; logo precisamos sempre levar em consideração a possibilidade de que elas tenham sido substituídas por algum adversário malicioso. Por outro lado, precisamos supor que a KGC não faz este tipo de substituição pois, caso contrário, qualquer esquema seria trivialmente quebrado. Isto nos força a considerar dois tipos diferentes de ataques:

- **Tipo I.** Qualquer adversário malicioso que tem acesso apenas a informações públicas mas pode substituir a chave pública de qualquer usuário.
- **Tipo II.** A própria KGC que, apesar de não poder substituir a chave pública do usuário que sofrerá o ataque, conhece sua chave parcial privada.

Sendo assim, demonstrações de segurança de esquemas de CL-PKC geralmente vão conter duas provas separadas, uma para cada tipo de adversário. Definiremos agora, formalmente, os jogos que definirão a segurança contra estes dois adversários para esquemas de assinaturas sem certificados.

Em ambos os jogos um adversário $\mathcal{A}$ tentará quebrar a segurança EU-CMA do esquema. Seja k o parâmetro de segurança. O jogo prossegue da seguinte forma:

1. Um desafiante $\mathcal{D}$ gera os parâmetros do sistema (mpk, msk) de forma que sua distribuição seja indistinguível da que seria obtida se $(mpk, msk) \leftarrow \texttt{Inicializar}(1^k)$.

2. $\mathcal{A}$ recebe como entrada mpk e (possivelmente) informação extra aux. Durante sua execução, $\mathcal{A}$ tem acesso a alguns oráculos, descritos a seguir. Se $\mathcal{A}$ não abortar, ele deve gerar uma falsificação (ID^*, M^*, ς^*).

O adversário vence o jogo se $\texttt{Verificar}(mpk, ID^*, P_{ID^*}, M^*, \varsigma^*) = \texttt{ACEITA}$ e o par (ID^*, M^*) não foi consultado ao oráculo de assinatura. Um esquema CLS é seguro se qualquer adversário $\mathcal{A}$ tem chance no máximo desprezível de vencer este jogo. Resta-nos apenas definir os oráculos aos quais adversários têm acesso:

- **RevelaChavePública.** O adversário fornece uma identidade ID_i e o desafiante retorna a chave pública correspondente P_{ID_i}, gerando-a se necessário.
- **RevelaChaveParcial.** O adversário fornece uma identidade ID_i e o desafiante retorna a chave parcial correspondente D_{ID_i}, gerando-a se necessário.
- **RevelaValorSecreto.** O adversário fornece uma identidade ID_i e o desafiante retorna o valor secreto x_{ID_i} correspondendo à chave pública P_{ID_i}. Se a chave pública foi substituída (pelo adversário) e o valor secreto é desconhecido, o desafiante pode retornar $\perp$.
- **SubstituiChavePública.** O adversário fornece uma identidade ID_i e a nova chave pública P'_{ID_i}. O desafiante faz com que P'_{ID_i} passe a ser a chave pública de ID_i. *Opcionalmente*, o adversário pode fornecer também o valor secreto x_{ID_i} ao desafiante.
- **Assina.** O adversário fornece a identidade ID_i e a mensagem M_i. O desafiante retorna uma assinatura σ de ID_i em M_i. A semântica exata desta assinatura no caso em que a chave pública de ID_i foi substituída será discutida posteriormente.

A segurança de assinaturas sem certificados é expressa então por dois jogos bastante similares, respectivamente contra $\mathcal{A}_I$ e $\mathcal{A}_{II}$ definidos a seguir:

Definição 8 (Segurança contra adversários do Tipo I)**.** *Seja $\mathcal{D}_I$ um algoritmo desafiante e k um parâmetro de segurança :*

1. *$\mathcal{D}_I$ gera (mpk, msk) de acordo com a distribuição correta.*
2. *$\mathcal{D}_I$ executa $\mathcal{A}_I(1^k, mpk)$; durante sua execução $\mathcal{A}_I$ tem acesso a todos os oráculos definidos acima (que são simulados por $\mathcal{D}_I$).*
3. *$\mathcal{A}_I$ retorna (ID^*, M^*, ς^*).*

$\mathcal{A}_I$ vence o jogo se:

- *`Verifica`$(mpk, ID^*, P_{ID^*}, M^*, \varsigma^*) = ACEITA$;*
- *a consulta `Assina`(ID^*, M^*) nunca foi feita;*
- *e a consulta `RevelaChaveParcial`(ID^*) também nunca foi feita.*

Um esquema é dito seguro contra adversários de Tipo I se nenhum $\mathcal{A}_I$ tem chance não-desprezível de ganhar este jogo.

Definição 9 (Segurança contra adversários do Tipo II)**.** *Seja $\mathcal{D}_{II}$ um algoritmo desafiante e k um parâmetro de segurança :*

1. *$\mathcal{D}_{II}$ gera (mpk, msk) de acordo com a distribuição correta.*
2. *$\mathcal{D}_{II}$ executa $\mathcal{A}_{II}(1^k, mpk, msk)$; durante sua execução $\mathcal{A}_{II}$ tem acesso a todos os oráculos definidos acima (que são simulados por $\mathcal{D}_{II}$).*

3. $\mathcal{A}_{II}$ *retorna* (ID^*, M^*, ς^*).

$\mathcal{A}_{II}$ *vence o jogo se:*

- `Verifica`$(mpk, ID^*, P_{ID^*}, M^*, \varsigma^*) = ACEITA$;
- *a consulta* `Assina`(ID^*, M^*) *nunca foi feita;*
- *a consulta* `SubstituiChavePública`$(ID^*, .)$ *nunca foi feita;*
- *a consulta* `RevelaValorSecreto`(ID^*) *também nunca foi feita.*

Um esquema é dito seguro contra adversários de Tipo II se nenhum $\mathcal{A}_{II}$ *tem chance não-desprezível de ganhar este jogo.*

Falta-nos apenas definir precisamente o funcionamento do oráculo de assinatura. Na definição original de segurança de [ARP03], é exigido que o oráculo de assinaturas retorne assinaturas válidas *mesmo que a chave pública do usuário tenha sido substituída* e o valor secreto correspondente seja desconhecido. Este tipo de oráculo é chamado de oráculo forte pois dá bastante poder ao adversário. Uma relaxação desta condição bastante condizente com a realidade de um ataque como este é a de que o oráculo deve retornar assinaturas válidas apenas para a chave pública original ou apenas para valores secretos fornecidos pelo adversário (oráculo fraco). Estas versões mais fracas são mais próximas da realidade; afinal, um usuário do sistema não seria capaz de (nem estaria disposto a) gerar assinaturas relacionadas a chave públicas cujo componente secreto não conhece.

Existe então uma pequena controvérsia em relação a qual oráculo é o mais apropriado: o que dá mais poder ao adversário (e potencialmente uma garantia maior de segurança), ou o que é mais próximo da realidade e, por ser mais simples, potencialmente permite esquemas mais eficientes. Em geral, não existe resposta única para todas as situações, mas este trabalho apresenta um argumento interessante para o uso de oráculos fortes: para demonstrar a segurança da agregação de assinaturas no nosso esquema é essencial que o nosso oráculo de assinaturas seja um oráculo forte. Discutiremos esta questão mais a fundo durante a análise de segurança do esquema, mas já adiantamos que nas demonstrações que seguem utilizaremos principalmente oráculos de assinaturas *fortes*.

4.2.1 Um detalhe sobre a substituição de chaves públicas

As demonstrações de segurança de vários esquemas de assinatura sem certificados fazem a suposição de que, ao substituir uma chave pública PK_i por PK'_i, o adversário sempre conhece o valor secreto x'_i correspondente à nova chave PK'_i; e que, portanto, o desafiante pode recuperar este valor ao final da simulação do ataque. Esta suposição aparece, com leves variações, nas demonstrações de segurança dos esquemas propostos em [HSMZ05], [YHG06], [Goy06], [CPHL07] e [DW07].

Argumentamos em [CD07b] que esta suposição é injustificável e pode levar a vulnerabilidades em esquemas que se acreditava serem demonstravelmente seguros; afinal, o que

esta suposição implica, no fundo, é que a única possível estratégia para um adversário calcular novas chaves públicas PK'_i seria primeiro gerar o valor secreto x'_i correspondente e depois gerar PK'_i de acordo com a descrição do esquema. Isto certamente não é verdade e mostramos que esta suposição pode ser explorada analisando a demonstração de segurança de [Goy06] e apresentando um ataque onde o adversário substitui a chave pública do usuário-alvo por uma calculada de forma "indireta" e cujo valor secreto ele não é capaz de calcular. Discutiremos os detalhes em §4.4.4.

4.2.2 Segurança de agregação de assinaturas sem certificados

Os modelos acima claramente não levam em consideração a possibilidade de agregação de assinaturas[7]. Exceto por alguns detalhes que discutiremos a seguir, a extensão destes modelos para permitir a agregação de assinaturas é bastante direta: a falsificação que deve ser gerada pelo adversário $\mathcal{A}$ é agora representada pela tupla $\langle \mathbb{U}, \mathbb{M}, \gamma \rangle$, onde:

- $\mathbb{U} = \{u_0, u_1, \ldots, u_n\}$ é a lista de usuários;
- $\mathbb{M} = \{m_0, m_1, \ldots, m_n\}$ é a lista de mensagens;
- γ é a assinatura agregada.

$\mathcal{A}$ vencerá o jogo se $\texttt{VerificaAgg}(\gamma, \mathbb{U}, \mathbb{M}) = \texttt{ACEITA}$ e houver pelo menos um par $(u_i, m_j), u_i \in \mathbb{U}, m_j \in \mathbb{M}$ que não foi consultado ao oráculo de assinaturas. A definição dos jogos para o cenário de assinaturas agregadas ficam então completamente análogas às definições 8 e 9, alterando apenas o tratamento da saída de $\mathcal{A}$.

Talvez a diferença mais relevante entre os oráculos para o caso de uma assinatura e o caso de assinaturas agregadas é a possibilidade de adversários Tipo-II substituírem a chave pública de outros usuários que não o usuário-alvo da falsificação. No caso de assinaturas simples esta possibilidade não traz qualquer poder adicional ao adversário, mas isso não é verdade no caso de assinaturas agregadas: é possível que $\mathcal{A}_{II}$ substitua a chave pública de usuários envolvidos no agregado, mas que não são o usuário-alvo. Como na nossa definição exigimos a existência de ao menos uma assinatura "falsificada" no agregado, é possível que $\mathcal{A}_{II}$ fosse capaz de gerar falsificações $\langle \mathbb{U}, \mathbb{M}, \gamma \rangle$ tais que:

- A chave pública de $u^* \in \mathbb{U}$ não foi substituída;
- (u^*, m^*) não foi consultado ao oráculo de assinaturas;
- $\exists u_i \in \mathbb{U}$ tais que $\mathcal{A}_{II}$ substituiu a chave pública de u_i.

Nesta situação $\mathcal{A}_{II}$ claramente tem mais poder do que na situação em que ele não pode substituir chaves públicas de $u_i \neq u^*$. O quanto este tipo de ataque é prático é discutível: afinal, por que a KGC iria substituir a chave de alguns usuários e não de outros? Na verdade este modelo de ataque retrata o modelo onde a KGC poderia, por exemplo, criar "usuários-fantasmas" que a ajudariam a montar um ataque: na situação de assinaturas

[7] §5.2.1 contém uma breve introdução a assinaturas agregadas

únicas isso não traz poder algum para a KGC; já na situação de assinaturas agregadas isto traz sim poder adicional para a KGC e, por isso, consideramos oráculos de substituição de chave pública que permitem esta operação.

4.2.3 KGCs maliciosas

Em [ACL+06] os autores fazem uma observação interessante sobre os modelos de segurança de CL-PKC: eles chamam atenção para o fato de que, invariavelmente, assume-se que os parâmetros do sistema são criados corretamente. Note que, apesar de em geral supormos que a KGC não substitui chaves públicas de usuários, a princípio existe a possibilidade de que os parâmetros do sistema sejam gerados de forma a permitir à KGC montar ataques como, por exemplo, calcular chaves secretas de usuários. Dependendo do esquema, é possível que parâmetros gerados de forma maliciosa permitam à KGC atacar qualquer usuário em potencial do sistema ou que ela seja obrigada a escolher uma identidade específica para atacar: i.e., sabendo que um certo "Keyser Soze" será usuário do sistema a KGC poderia gerar os parâmetros do sistema de forma a ser capaz de atacar apenas este usuário.

O primeiro tipo de ataque (quebra para qualquer usuário) é claramente mais grave do que o segundo, especialmente porque a técnica de encapsulamento de chaves públicas torna o ataque direcionado a uma certa identidade inviável: já que a identidade dos usuários é na verdade a concatenação de suas identidades com suas chaves públicas, arbitrariamente escolhidas, a KGC não tem como saber de antemão que valor será usado pelo usuário que deseja atacar. A quebra geral, por outro lado, torna o sistema completamente vulnerável e, mesmo com o encapsulamento da chave, não se consegue atingir o nível 3 de Girault.

Em [ACL+06] são apresentados argumentos de que duas construções genéricas, uma para ciframento, outra para assinaturas, são seguras mesmo neste modelo em que a KGC pode ser "ligeiramente" maliciosa. Consideramos que apenas a quebra total é um problema sério para esquemas de CLS, já que supomos sempre que, quando um nível de segurança maior é necessário, o encapsulamento de chaves é utilizado. Ao revisarmos a segurança dos esquemas de CLS, no final deste capítulo, destacaremos a resistência ou não de cada esquema a este tipo de ataque.

4.3 Do uso da técnica de reexecução de oráculos em CLS

Um dos grandes avanços no campo de segurança demonstrável nos anos 90 foi a Técnica de Reexecução de Oráculos, apresentada por Pointcheval & Stern em [PS96]. Esta técnica

permitiu a construção de demonstrações de segurança para diversos esquemas de assinaturas, como por exemplo o Schnorr e leves variações do ElGamal. Nesta seção discutimos sua relevência e sua (não tão trivial) aplicabilidade a esquemas de assinaturas sem certificados.

4.3.1 A heurística de Fiat & Shamir

No Crypto'86, Fiat & Shamir apresentaram as noções comuns de esquemas de autenticação, identificação e assinatura sob uma ótica que facilitou bastante o entendimento da relação entre elas [FS86]. Imaginemos a seguinte situação: dois usuários, A e B participam de um protocolo que visa prover algum tipo de autenticação entre as partes. Fiat & Shamir identificam então três níveis de proteção:

1. **Esquemas de autenticação.** A é capaz de provar a B que é A, mas ninguém mais consegue provar a B que é A.

2. **Esquemas de identificação.** A é capaz de provar a B que é A, mas B não consegue provar a ninguém mais que é A.

3. **Esquemas de assinatura.** A é capaz de provar a B que é A, mas B não consegue provar nem a si mesmo que é A.

Esquemas de autenticação implicam um certo nível de confiança entre A e B, pois B poderia potencialmente utilizar uma execução válida do protocolo (entre A e B) para provar a outros usuários que é A. Esquemas de identificação não permitem este tipo de problema, mas B poderia potencialmente gerar uma execução válida do protocolo sem interagir com A: ou seja, um esquema de identificação não pode ser usado por B para provar a outras pessoas (um juiz, por exemplo) que B de fato interagiu com A. Esquemas de assinaturas podem ser utilizados quando uma prova como esta é necessária, pois B não poderia gerá-la sozinho.

Com esta generalização em mente, Fiat & Shamir propõem uma heurística para construir esquemas de assinaturas a partir de *esquemas de identificação interativos de 3 passos*.

Definição 10 (*Esquemas de identificação de 3 passos*)**.** *Um esquema de identificação de 3 passos é um protocolo interativo entre dois usuários, o provador A e o verificador B, estruturado como um protocolo de desafio-resposta como a seguir:*

1. *Seja PK_A a informação pública (certificado, chave pública, etc.) de A. A (possivelmente) gera alguma informação aleatória r_A e envia o par (PK_A, r_A) para B.*

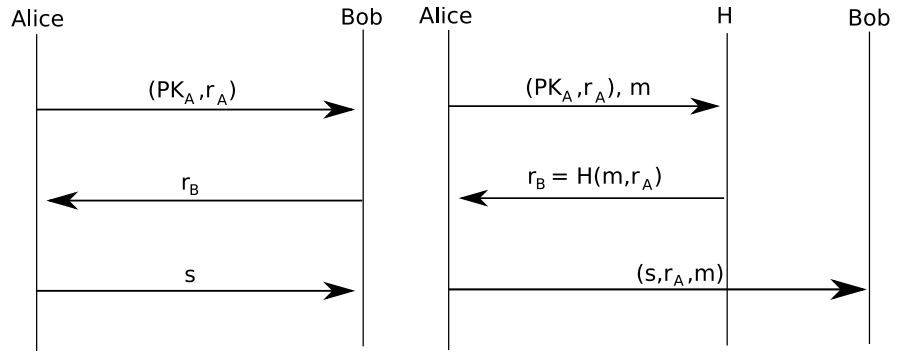
2. *B verifica as informações públicas de A, gera um desafio r_B e o envia de volta para A.*

3. A calcula uma resposta σ_A apropriada para o desafio r_B e a retorna para B.

Qualquer usuário que não conheça as informações secretas de A tem probabilidade desprezível de não ser detectado por B.

A figura 4.1(a) ilustra melhor o conceito de identificação em 3 passos.

A idéia de Fiat & Shamir foi então utilizar um esquema de identificação em 3 passos e substituir a interação com B pelo uso de uma função pseudo-aleatória H: como a saída de H é independente da entrada, ela simula a interação com B (que deveria gerar um desafio aleatório) de maneira posteriormente verificável por outros usuários (já que todos têm acesso à descrição de H), gerando assim um esquema de assinaturas (ilustrado na figura 4.1(b)). Este procedimento para gerar esquemas de assinatura a partir de protocolos de identificação ficou conhecido como *heurística de Fiat/Shamir*.



(a) Identificação em 3 passos (b) Assinaturas Fiat/Shamir

Figura 4.1: Ilustração da heurística de Fiat & Shamir.

Esquemas de assinaturas construídos com a heurística de Fiat/Shamir costumam gerar assinaturas que são triplas (r, h, σ), onde r é derivado da informação aleatória gerada por A no primeiro passo do protocolo de identificação, h é $H(m, r)$ (simulando a resposta de B) e σ é derivado da resposta final de A. Chamamos esquemas com este formato de *Esquemas de Assinatura Genéricos*:

Definição 11 (*Esquemas de Assinatura Genéricos*)**.** *Um esquema de assinatura $\mathcal{S}$ é denominado* genérico *se gera assinaturas (r, h, σ) onde:*

- *r é um valor escolhido uniformemente em um domínio grande.*
- *h é o hash da mensagem m e de r, $h = H(m, r)$.*
- *σ depende apenas de m, r e h.*

Muitos esquemas importantes são genéricos; em especial, o próprio Fiat/Shamir [FS86] e o Schnorr [Sch89] são genéricos.

4.3.2 A Técnica de Reexecução de Oráculos

Em [PS96], Pointcheval & Stern utilizam o modelo do oráculo aleatório para analisar a segurança de esquemas de assinaturas genéricos e propõem um novo paradigma para provas de segurança, denominado por eles de *Técnica de Reexecução de Oráculos*[8]. Faremos aqui uma exposição um tanto limitada desta técnica, apresentando os conceitos mais relevantes para o seu entendimento e para a compreensão das seções a seguir. Para maiores informações, referimos o leitor ao artigo original de Pointcheval & Stern [PS96], e a uma versão posterior mais extensa [PS00].

Para tornar a discussão mais concreta, utilizaremos o esquema de assinaturas de Schnorr como exemplo no que segue.

Definição 12. (Schnorr). *Sejam p e q primos tais que $p \equiv 1 \pmod{q}$, onde p é muito maior que q, por exemplo $p \approx 2^{1024}$ e $q \approx 2^{160}$. Seja g um elemento de $\mathbb{Z}_p^*$ de ordem q. O esquema de Schnorr é definido pelos três algoritmos abaixo.*

> `Geração de Chaves.` Escolha $x \xleftarrow{R} \mathbb{Z}_q^*$ e calcule $y = g^{-x} \mod p$. x será a chave secreta e y a chave pública.
>
> `Assinatura.` Seja m a mensagem a ser assinada. Escolha $k \xleftarrow{R} \mathbb{Z}_q^*$ e calcule $r = g^k \mod p$. Seja $h = H(m, r) \mod q$. Calcule $\sigma = k + hx \mod q$. A assinatura será então o par (r, σ).
>
> `Verificação.` Seja $h = H(m, r)$. A verificação aceita se a equação a seguir vale:
>
> $$h = H(m, g^{\sigma} y^{h} \mod p).$$

A primeira característica comum a muitos dos esquemas de assinatura genéricos explorada por Pointcheval e Stern é o fato de que obter duas assinaturas σ_1 e σ_2 em valores diferentes de h, mas utilizando o mesmo aleatório r, costuma revelar a chave privada dos esquemas. Por exemplo, no esquema de Schnorr, se existem duas assinaturas (σ, r) e (σ', r') de mensagens diferentes (logo, $h \neq h'$) mas com o mesmo aleatório ($r = r'$ e portanto, $k = k'$), teríamos que:

$$\begin{aligned} \sigma &= k + hx \mod q, \\ \sigma' &= k + h'x \mod q, \end{aligned}$$

implicando que $\sigma - \sigma' \equiv (h - h')x \pmod{q}$ e o valor da chave secreta x é facilmente recuperável:

$$x = (\sigma - \sigma') \cdot (h - h')^{-1} \mod q. \tag{4.1}$$

[8] Do inglês *Oracle Replay Technique*.

Informalmente, a técnica de reexecução de oráculo funciona da seguinte forma: após uma execução bem sucedida de um ataque, onde o adversário $\mathcal{A}$ gerou a assinatura (r, h, σ), o ataque é reexecutado, com a mesma fita aleatória: isso implica que tudo ocorre da mesma forma e eventualmente o adversário geraria a mesma falsificação ao final da execução. O desafiante $\mathcal{D}$, no entanto, *sabe* que a falsificação será de h e que, com alta probabilidade[9], a consulta $h = H(M, r)$ é feita ao oráculo da função de hash. Digamos que esta seja a β-ésima consulta ao oráculo. Se denotarmos por $\mathcal{Q}_i$ a i-ésima consulta de $\mathcal{A}$ ao oráculo e por ρ_i a i-ésima resposta, temos que $\mathcal{Q}_\beta = (M, r)$ e que $\rho_\beta = h$. Para todas as consultas antes da β-ésima, o desafiante se comportará exatamente como no primeiro, e bem-sucedido, ataque; da β-ésima em diante, $\mathcal{D}$ volta a dar resposta aleatórias para as consultas ao oráculos.

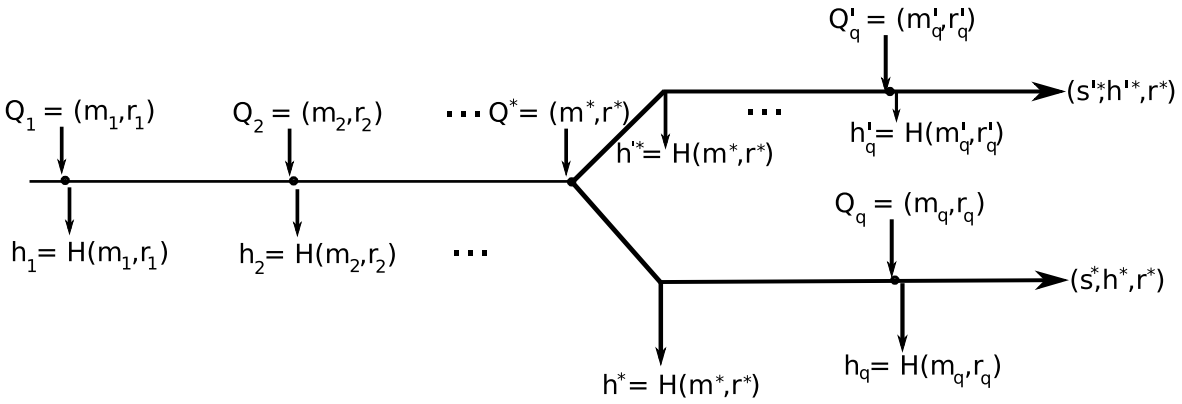


Figura 4.2: Ilustração da Técnica de Reexecução de Oráculos.

Com altíssima probabilidade o valor de $h' = H(M, r)$ na segunda execução do ataque será diferente de h. O valor de r no entanto é o mesmo, pois foi "escolhido" antes da β-ésima consulta à função de hash, enquanto o segundo ataque era exatamente igual ao primeiro. Assuma, por enquanto, que a probabilidade de sucesso de $\mathcal{A}$ se mantém não-desprezível nesta reexecução. Espera-se, portanto que uma segunda falsificação (r', h', σ') seja gerada, onde $h \neq h'$ e $\sigma \neq \sigma'$, mas $r = r'$.

Enunciaremos, sem demonstrar agora, o principal resultado de [PS96], o Lema da Bifurcação[10] para esquemas de assinaturas genéricos:

Teorema 1 (*Lema da Bifurcação.*)**.** *Seja $\mathcal{A}$ uma máquina de Turing probabilística de tempo polinomial cuja entrada consiste de dados públicos. Denotamos por Q_H e Q_S respectivamente o número de consultas que $\mathcal{A}$ pode fazer ao oráculo aleatório e ao oráculo*

[9] Como a demonstração utiliza o modelo do oráculo aleatório, a saída de $H()$ é completamente aleatória e, portanto, a probabilidade da falsificação ser válida sem que a consulta seja feita é $\frac{1}{2^k}$, desprezível.

[10] Do inglês *Forking Lemma*.

de assinaturas. Suponha que, com limite máximo de tempo T, $\mathcal{A}$ produz, com probabilidade $\epsilon \geq 10(Q_S+1)(Q_S+Q_H)/2^k$, uma assinatura válida (m, r, h, σ). Se as triplas (m, r, h, σ) podem ser simuladas sem o conhecimento da chave secreta, com distribuição de probabilidade indistinguível, então uma reexecução do ataque gera duas assinaturas válidas (m, r, h, σ) e (m, r, h', σ') tais que $h \neq h'$ em tempo $T' \leq 23Q_H T/\epsilon$ e com probabilidade $\epsilon' \geq \frac{1}{9}$.

Vamos ilustrar a utilização da técnica de reexecução de oráculos e do Lema da Bifurcação esboçando uma demonstração de segurança para o esquema de Schnorr no modelo do oráculo aleatório.

Teorema 2. *Suponha que em tempo T um adversário $\mathcal{A}$ consegue gerar uma falsificação existencial do esquema de Schnorr através de um ataque adaptativo de mensagem escolhida com probabilidade de sucesso ϵ. Sejam Q_H e Q_S os números de consultas feitas por $\mathcal{A}$, respectivamente ao oráculo aleatório e ao oráculo de assinaturas. Suponha que $\epsilon \geq 10(Q_S+1)(Q_S+Q_H)/q$. Existe então um algoritmo $\mathcal{D}$ que usa $\mathcal{A}$ e consegue resolver o problema do logaritmo discreto em subgrupos de ordem prima em tempo esperado $\leq 120686Q_H T/\epsilon$.*

Demonstração. Recebemos como entrada uma instância do problema do logaritmo discreto, i.e. $g, y \in \mathbb{Z}_q^*$. Devemos mostrar que o desafiante $\mathcal{D}$ é capaz de calcular $x \in \mathbb{Z}_q^*, y = g^x \mod q$. $\mathcal{D}$ começa por escolher g como o elemento de ordem q na geração de parâmetros do sistema e faz y ser a chave pública do usuário a ser atacado.

Suponha agora que exista um algoritmo $\mathcal{A}$ capaz de falsificar assinaturas sob um ataque adaptativo seguindo as restrições indicadas no enunciado do teorema (probabilidade de sucesso ϵ, números máximo de consultas Q_H e Q_S). Como já vimos anteriormente, o esquema de Schnorr obedece os requisitos para a utilização da técnica de reexecução de oráculos; logo, se $\mathcal{D}$ for capaz de simular as assinaturas (r, h, σ) produzidas por um usuário legítimo, sem o conhecimento da chave secreta do usuário, (já que $\mathcal{D}$ desconhece o logaritmo discreto $\log_g y$), $\mathcal{A}$ pode ser utilizado, segundo o Teorema 1, para obter duas assinaturas, (m, r, h, σ) e (m, r, h', σ'), onde $h \neq h'$. Tendo isto, basta-nos utilizar a equação (4.1) e calcular a chave secreta $x = \log_g y$, que é a resposta para a instância do logaritmo discreto.

Para provar que é possível simular as assinaturas, demonstramos o seguinte lema:

Lema 1. *As distribuições a seguir são equivalentes:*

$$\delta = \left\{ (r,h,\sigma) \;\middle|\; \begin{array}{c} K \stackrel{R}{\leftarrow} \mathbb{Z}_q^* \\ h \stackrel{R}{\leftarrow} \mathbb{Z}_q \\ r = g^K \mod p \\ \sigma = K + xh \mod q \end{array} \right\} \; e \; \delta' = \left\{ (r,h,\sigma) \;\middle|\; \begin{array}{c} K \stackrel{R}{\leftarrow} \mathbb{Z}_q \\ h \stackrel{R}{\leftarrow} \mathbb{Z}_q \\ \sigma = K \\ r = g^\sigma y^h \mod p \\ r \neq 1 \mod p \end{array} \right\}$$

Demonstração. Primeiro escolhemos uma tupla $(\epsilon, \gamma, \beta)$ do conjunto de possíveis assinaturas: sejam então $\epsilon \in \mathbb{Z}_p^*$, $\gamma \in \mathbb{Z}_q$ e $\beta \in \mathbb{Z}_q$ tais que $g^\gamma y^\beta \equiv \epsilon \neq 1 \pmod p$. Calculamos então a probabilidade de aparição desta tupla em cada uma das distribuições de probabilidade acima:

$$\begin{aligned} \Pr_{\delta}[(r,h,\sigma) = (\epsilon,\beta,\gamma)] &= \Pr_{K \neq 0, h}\left[\begin{array}{c} g^K = \epsilon; h = \beta; \\ K + xh = \gamma; \end{array} \right] = \frac{1}{q(q-1)} \\ \Pr_{\delta'}[(r,h,\sigma) = (\epsilon,\beta,\gamma)] &= \Pr_{K \neq 0, h}\left[\begin{array}{c} \epsilon = r = g^K y^h; \\ h = \beta; \sigma = K = \gamma; \end{array} \;\middle|\; r \neq 1 \mod p \right] = \frac{1}{q(q-1)} \end{aligned}$$

□

Este lema nos fornece a maneira perfeita para simular assinaturas: ao invés de calcular $r = g^K \mod p$ e depois calcular σ (utilizando x), $\mathcal{D}$ faz o contrário: escolhe $\sigma \stackrel{R}{\leftarrow} \mathbb{Z}_q$ e $h \stackrel{R}{\leftarrow} \mathbb{Z}_q$, calcula $r = g^\sigma y^h \mod p$ e faz $H(m,r) = h$. Na improvável situação em que $r \equiv 1 \pmod p$, $\mathcal{D}$ cancela a simulação e começa novamente. Como h foi escolhido aleatoriamente, a distribuição de $H(\cdot)$ não muda e as assinaturas são simuladas perfeitamente. A segurança existencial do esquema de Schnorr contra ataques adaptativos no modelo do oráculo aleatório está assim demonstrada. ■

4.3.3 Aplicação à CLS

O problema em aplicar cegamente a técnica de reexecução de oráculos ao modelo de criptografia de chave pública sem certificados é a possibilidade de, em CL-PKC, haver substituição de chaves públicas; veja que, implicitamente, na segurança de esquemas como o de Schnorr, assumimos que a chave pública das duas falsificações é a mesma. Esta é uma suposição válida num mundo onde as chaves são explicitamente certificadas, mas em CL-PKC, como já vimos, há a possibilidade de substituição de chaves públicas; neste caso, um adversário poderia potencialmente gerar uma primeira falsificação sob uma chave pública e, na reexecução do ataque, a segunda falsificação seria gerada sob outra chave pública.

Para impedir este tipo de situação temos de redefinir a noção de *esquema genérico* para o cenário de CL-PKC: assim como, em esquemas tradicionais fazemos h depender

também do aleatório r para garantir que ele é o mesmo em ambas as execuções do ataque, em CL-PKC faremos o h depender também da chave pública PK.

Definição 13 (*Esquemas de Assinatura sem Certificados Genéricos*, ou CL-Genéricos)**.** *Um esquema de assinatura sem certificados $\mathcal{S}$ é denominado* genérico *se gera assinaturas (r, h, σ) onde:*

- *r é um valor escolhido uniformemente em um domínio grande.*
- *h é o hash de, pelo menos, a mensagem m, r, a identidade do usuário ID_i e da chave pública PK_{ID_i}.*
- *σ depende apenas de m, r e h.*

Para esquemas CL-genéricos, como na definição acima, a técnica de reexecução de oráculos pode ser utilizada de maneira completamente análoga à que é feita com esquemas genéricos tradicionais. Para provar que isto é verdade, mostramos a seguir uma versão análoga do Lema da Bifurcação para ataques passivos aplicável a esquemas de assinaturas sem certificados genéricos. A demonstração que apresentamos é essencialmente a mesma que Pointcheval & Stern apresentam em [PS00]; tentamos apenas tornar a apresentação um pouco mais pausada e didática, visto que temos mais espaço disponível para seu desenvolvimento. No final da seção, fazemos um argumento análogo ao apresentado em [PS00] para mostrar que o resultado se estende a ataques adaptativos.

Teorema 3 (*Lema da Bifurcação para CL-PKC - Adv. Passivos.*)**.** *Seja $\mathcal{S}$ um esquema de assinaturas CL-Genérico com parâmetro de segurança k. Seja $\mathcal{A}$ uma máquina de Turing probabilística de tempo polinomial cuja entrada consiste de dados públicos. Denotamos por Q o número de consultas que $\mathcal{A}$ pode fazer ao oráculo aleatório. Suponhamos que, com limite máximo de tempo T, $\mathcal{A}$ produza, com probabilidade $\epsilon \geq 7Q/2^k$, uma assinatura válida $(m, ID, PK, r, h, \sigma)$. Se as assinaturas $(m, ID, PK, r, h, \sigma)$ podem ser simuladas sem o conhecimento da chave secreta, com distribuição de probabilidade indistinguível, então uma reexecução do ataque gera duas assinaturas válidas $(m, ID, PK, r, h, \sigma)$ e $(m, ID, PK, r, h', \sigma')$, tais que $h \neq h'$, em tempo $T' \leq 16QT/\epsilon$ e com probabilidade $\epsilon' \geq \frac{1}{9}$.*

Demonstração. Uma execução de um ataque é univocamente definida por um par (ω, f), onde ω denota uma fita aleatória e f denota uma instanciação do oráculo aleatório.

Denotamos por $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_Q$ as consultas realizadas por $\mathcal{A}$ ao oráculo aleatório numa execução específica, e por $\rho = (\rho_1, \rho_2, \ldots, \rho_Q)$ o conjunto das respectivas respostas. Logo, uma escolha aleatória de f equivale a uma escolha aleatória de ρ. Como f é um oráculo aleatório, a probabilidade de $\mathcal{A}$ ter sucesso em gerar uma assinatura $(m, ID, PK, r, h, \sigma)$, onde $h = f(m, ID, PK, r)$[11], sem que a consulta $f(m, ID, PK, r)$ tenha sido feita ao oráculo é muito baixa; pra ser mais preciso, esta probabilidade é menor

que 2^{-k}. Portanto, dada uma execução (ω, f) em que $\mathcal{A}$ tem sucesso, é provável que a consulta tenha sido realizada: denotamos por $Ind(\omega, f)$ o índice desta consulta (fazendo $Ind(\omega, f) = \infty$ caso ela não seja realizada). Sendo assim, $\mathcal{Q}_{Ind(\omega,f)} = (m, ID, PK, r)$.

Definimos então os seguintes conjuntos de execuções:

$$\begin{aligned} S &= \{(\omega, f) | \mathcal{A}^f(\omega) \text{ tem sucesso } \wedge Ind(\omega, f) \neq \infty\}, e \\ S_i &= \{(\omega, f) | \mathcal{A}^f(\omega) \text{ tem sucesso } \wedge Ind(\omega, f) = i\}, \text{ para } i \in \{1, \ldots, Q\}. \end{aligned}$$

Chamamos S de *conjunto de pares* (ω, f) *de sucesso* e chamamos a atenção para o fato de que os S_i particionam S.

As definições anteriores nos dão imediatamente um primeiro limite inferior para a probabilidade de sucesso $\Pr[S] \geq \epsilon - \frac{1}{2^k}$. Lembrando que, por definição, $\epsilon \geq 7Q/2^k$,

$$\begin{aligned} \Pr[S] &\geq \epsilon - \frac{1}{2^k} \\ &\geq \frac{7Q}{2^k} - \frac{1}{2^k} \\ &\geq \frac{7Q - 1}{2^k} \\ &\geq \frac{6}{7} \cdot \epsilon. \end{aligned}$$

Podemos então executar o adversário $\frac{2}{\epsilon}$ vezes com escolhas aleatórias de ω e f. Como $\Pr[S] \geq 6\epsilon/7$, com probabilidade maior que $1 - (1 - 6\epsilon/7)^{2/\epsilon}$, pelo menos um par (ω, f) vai estar em S. É fácil de verificar que:

$$\begin{aligned} 1 - (1 - 6\epsilon/7)^{2/\epsilon} &\geq 1 - e^{-12/7} \\ &\geq \frac{4}{5}. \end{aligned}$$

Seja I o conjunto de índices i mais prováveis, i.e. $I = \{i | \Pr[S_i|S] \geq 1/2Q\}$. O lema a seguir mostra que, em caso de sucesso, a probabilidade de i estar em I é pelo menos $\frac{1}{2}$.

Lema 2. $\Pr[Ind(\omega, f) \in I | S] \geq \frac{1}{2}$.

Demonstração. Como os S_i formam uma partição de S,

$$\Pr[Ind(\omega, f) \in I | S] = \sum_{i \in I} \Pr[S_i|S] = 1 - \sum_{i \notin I} \Pr[S_i|S].$$

Para terminar a demonstração basta lembrarmos dois fatos:

[11] A definição de esquema CL-genérico permite que h seja o hash de *pelo menos* os valores utilizados nesta demonstração. De fato, outros valores podem ser adicionados a este hash e o teorema ainda vale, mas para deixar a apresentação melhor estruturada, escolhemos não levar este fato explicitamente em consideração.

1. *todo elemento que não pertence a I tem probabilidade* $< 1/2Q$. Isto segue direto da definição do conjunto I;

2. *existe pelo menos um elemento em I.* Isto pode ser observado por um argumento simples de contradição: se, para todo i, $\Pr[S_i|S] < 1/2Q$ teríamos $\sum_i \Pr[S_i|S] \leq Q \cdots 1/2Q = \frac{1}{2} < 1$, o que é uma contradição visto que os S_i particionam S.

Logo, $1 - \sum_{i \notin I} \Pr[S_i|S] \geq 1 - Q \cdot 1/2Q \geq \frac{1}{2}$. □

Utilizamos então um resultado da teoria de probabilidades, batizado em [PS96] de "*Lema da Separação*"[12], e revisado a seguir.

Lema 3. (Lema da Separação). *Seja $A \subset X \times Y$ tal que $\Pr[(x,y) \in A] \geq \gamma$. Para qualquer $\alpha < \gamma$, defina*

$$\begin{aligned} B &= \left\{ (x,y) \in X \times Y \;\middle|\; \Pr_{y' \in Y}[(x,y') \in A] \geq \gamma - \alpha \right\}, \text{ e} \\ \overline{B} &= (X \times Y) \setminus B; \end{aligned}$$

então as afirmativas a seguir valem:

(i) $\Pr[B] \geq \alpha$;

(ii) $\forall (x,y) \in B, \Pr_{y' \in Y}[(x,y') \in A] \geq \gamma - \alpha$;

(iii) $\Pr[B|A] \geq \alpha/\gamma$.

Demonstração. Provamos *(i)* por contradição: suponha que $\Pr[B] < \alpha$. Sabemos que $\Pr[A] \geq \gamma$, o que implica que

$$\gamma \leq \Pr[A] = \Pr[A|B] \cdot \Pr[B] + \Pr[A|\overline{B}] \cdot \Pr[\overline{B}].$$

Se $\Pr[B] < \alpha$, temos

$$\gamma \leq \Pr[A|B] \cdot \Pr[B] + \Pr[A|\overline{B}] \cdot \Pr[\overline{B}] < 1 \cdot \alpha + 1 \cdot (\gamma - \alpha) = \gamma,$$

o que implica uma contradição.

A afirmativa *(ii)* é uma conseqüência direta da definição.

A afirmativa *(iii)* é derivada da aplicação das leis de Bayes e da afirmativa *(ii)*:

$$\begin{aligned} \Pr[B|A] &= 1 - \Pr[\overline{B}|A] \\ &= 1 - \Pr[A|\overline{B}] \cdot \Pr[\overline{B}] / \Pr[A] \geq 1 - (\gamma - \alpha) \cdot 1/\gamma = \frac{\alpha}{\gamma}. \end{aligned}$$

□

[12] Do inglês *Splitting Lema*.

Basicamente, o Lema da Separação mostra que, para qualquer conjunto A suficientemente grande ($\Pr[A] \geq \gamma$), podemos encontrar um outro conjunto B relativamente grande ($\Pr[B] \geq \alpha$) tal que, para todo par $(x, y) \in B$, a probabilidade de, ao variarmos a segunda componente do par, digamos para y', o par (x, y') pertencer a A é ainda relativamente alta ($\Pr[(x, y') \in A] \geq \gamma - \alpha$).

Voltemos então à situação atual: temos, com alta probabilidade, uma execução de sucesso $(\omega, f) \in S_\beta$, onde $\beta = Ind(\omega, f)$ e $\beta \in I$. Denotemos, para um i qualquer, por f_{i-} a restrição de f a consultas de índice estritamente menor que i. Definimos f_{i+} de maneira complementar como as consultas de índice maior ou igual a i. Dada uma execução qualquer (ω, f), onde $i = Ind(\omega, f)$, geramos o par $(x, y) \in X \times Y$ simplesmente fazendo $x = \langle \omega, f_{i-} \rangle$, e $y = f_{i+}$. O produto cartesiano $X \times Y$ define então o espaço de possíveis execuções de $\mathcal{A}$, e cada par (x, y) representa uma execução em particular: o interessante é que, se tomarmos o conjunto A do Lema da Separação como o conjunto S_β, ao qual pertence a execução de $\mathcal{A}$ que obteve sucesso, temos certeza que existirá um conjunto B, digamos ω_β, tal que se tomarmos um par $(\langle \omega, f_{\beta-} \rangle, f_{\beta+}) \in B$ e alterarmos a segunda componente (i.e., escolhendo novas respostas para o oráculo aleatório após a consulta β), a probabilidade do novo par $(\langle \omega, f_{\beta-} \rangle, f'_{\beta+})$ pertencer a S_β é alta $(\gamma - \alpha)$. Esta é exatamente a análise que faremos para os i's que pertencem ao conjunto I de índices mais prováveis a seguir:

Para um $i \in I$ em particular, percebemos que o seguinte argumento é válido:

- seja o conjunto A no lema da separação, o conjunto das execuções de sucesso para o i sendo analisado;
- como $\Pr[S_i] \geq \Pr[S]/2Q, \forall i \in I$, concluímos que $\gamma = \Pr[A] = \Pr[S_i] \geq \Pr[S]/2Q$.

Tendo o γ já definido, escolhemos o α do lema da separação como $\gamma/2$. Sabemos então que existe um conjunto B no Lema da Bifurcação, ω_i, tal que para todo $(\langle \omega, f_{\beta-} \rangle, f_{\beta+}) \in \Omega_i$:

$$\begin{aligned} \Pr[(\omega, f') \in S_i | f_{i-} = f'_{i-}] &\geq \gamma - \alpha \\ &\geq \Pr[S]/4Q. \end{aligned}$$

Além disso, obtemos que $\Pr[\Omega_i | S_i] \geq \alpha/\gamma = \frac{1}{2}$.

Vamos fazer uma pequena pausa para revisar o que gostaríamos de conseguir: precisamos de uma execução (ω, f) onde

- o adversário $\mathcal{A}$ tenha sucesso ($(\omega, f) \in S$);
- o índice da consulta $Ind(\omega, f)$ pertence ao conjunto dos índices mais prováveis ($\exists i \in I, (\omega, f) \in S_i$);
- seja $\beta = Ind(\omega, f)$; deve haver uma alta probabilidade de, ao escolher um novo oráculo aleatório $f'_{\beta+}$ para a segunda execução, $(\omega, f_{\beta-} \| f'_{\beta+}) \in S_\beta$; o lema da separação nos diz que isso acontece para $(\langle \omega, f_{\beta-} \rangle, f_{\beta+}) \in \Omega_i$.

Queremos então obter uma execução $(\omega, f) \in S_i \cap \Omega_i$[13]. Como todos os S_i são disjuntos,

$$\begin{aligned}
\Pr[(\exists i \in I), (\omega, f) \in \Omega_i \cap S_i | S] &= \Pr\left[\bigcup_{i \in I} (\Omega_i \cap S_i) \middle| S\right] \\
&= \sum_{i \in I} \Pr[\Omega_i \cap S_i | S] \\
&= \sum_{i \in I} \Pr[\Omega_i | S] \cdot \Pr[S_i | S] \\
&\geq \left(\sum_{i \in I} \Pr[S_i | S]\right) / 2 \\
&\geq \frac{1}{4}.
\end{aligned}$$

Logo, a probabilidade de um par de sucesso (ω, f) estar em $\Omega_i \cap S_i$ é bastante alta. Definimos então $\beta = Ind(\omega, f)$ para um par de sucesso. Com probabilidade $\geq \frac{1}{4}$, $\beta \in I$ e $(\omega, f) \in S_\beta \cap \Omega_\beta$. Lembrando que a repetição do ataque $2/\epsilon$ vezes gera um par de sucesso com probabilidade $\geq \frac{4}{5}$, conseguimos um par $(\omega, f) \in S_i \cap \Omega_i, i \in I$ com probabilidade pelo menos $\frac{1}{5}$.

Se, após obtermos um par de sucesso (ω, f), repetirmos o ataque mantendo ω fixo mas com vários f' escolhidos aleatoriamente, de maneira que $f_{\beta-} = f'_{\beta-}$, sabemos que $\Pr[(\omega, f') \in S_\beta | f'_{\beta-} = f_{\beta-}] \geq \Pr[S]/4Q$. Logo,

$$\begin{aligned}
\Pr_{f'}[(\omega, f') \in S_\beta \text{ e } \rho_\beta \neq \rho'_\beta | f'_{\beta-} = f_{\beta-}] &\geq \Pr_{f'}[(\omega, f') \in S_\beta | f'_{\beta-} = f_{\beta-}] - \Pr_{f'}[\rho'_\beta = \rho_\beta] \\
&\geq \Pr[S]/4Q - \frac{1}{2^k} \\
&\geq \epsilon/14Q.
\end{aligned}$$

Logo, se reexecutarmos o ataque $14Q/\epsilon$ vezes com escolhas aleatórias de f' tais que $f'_{\beta-} = f_{\beta-}$ teremos probabilidade $\frac{3}{5}$ de obter outro sucesso.

Finalmente, após $2/\epsilon + 14Q/\epsilon$ repetições do ataque, com probabilidade $\frac{1}{5} \cdot \frac{3}{5} \geq \frac{1}{9}$, obtemos duas assinaturas válidas obedecendo as condições do teorema. ■

Para estender este resultado para ataques adaptativos de mensagem escolhida usamos exatamente o mesmo argumento presente em [PS00], reapresentado aqui por completude.

Teorema 4 (*Lema da Bifurcação para CL-PKC - Adv. Adaptativos.*)**.** *Seja $\mathcal{S}$ um esquema de assinaturas CL-Genérico com parâmetro de segurança k. Seja $\mathcal{A}$ uma máquina de Turing probabilística de tempo polinomial cuja entrada consiste de dados públicos.*

[13]Abusamos aqui um pouco a notação. Quando escrevemos $(\omega, f) \in \Omega_i$ deve-se ler $(\langle \omega, f_{i-} \rangle, f_{i+}) \in \Omega_i$.

Denotamos respectivamente por Q e S o número de consultas que $\mathcal{A}$ pode fazer ao oráculo aleatório e ao oráculo de assinaturas. Suponha que, com limite máximo de tempo T, $\mathcal{A}$ produz, com probabilidade $\epsilon \geq 10(S+1)(S+Q)/2^k$, uma assinatura válida $(m, ID, PK, r, h, \sigma)$. Se as assinaturas $(m, ID, PK, r, h, \sigma)$ podem ser simuladas sem o conhecimento da chave secreta, com distribuição de probabilidade indistinguível, então uma reexecução do ataque gera duas assinaturas válidas $(m, ID, PK, r, h, \sigma)$ e $(m, ID, PK, r, h', \sigma')$, tais que $h \neq h'$, em tempo $T' \leq 23QT/\epsilon$ e com probabilidade $\epsilon' \geq \frac{1}{9}$.

Demonstração. Como na prova do Teorema 3, sejam $(\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_Q)$ e $(\rho_1, \rho_2, \ldots, \rho_Q)$ respectivamente as consultas de $\mathcal{A}$ e as respostas obtidas em sua interação com o oráculo aleatório. Sejam $\mathcal{X}_1, \mathcal{X}_2, \ldots \mathcal{X}_S$ as consultas realizadas ao oráculo de assinaturas. Sabemos que as assinaturas podem ser simuladas; então para cada consulta $\mathcal{X}_i = (m_i, ID_i, PK_i)$ geramos a assinatura $(m_i, ID_i, PK_i, r_i, h_i, \sigma_i)$: o adversário deve assumir então que $h_i = f(m_i, ID_i, PK_i, r_i)$. A prova do Teorema 3 pode ser repetida inteiramente para esta nova situação, exceto pelo risco de colisões geradas por consultas ao oráculo de assinaturas. Neste momento nota-se a grande importância do "desafio" r_i na assinatura: como a componente r_i é aleatória e, segundo a definição de CL-Genérico, tem probabilidade no máximo $2/2^k$ de assumir um valor qualquer, é fácil limitar a probabiliade destas colisões ocorrerem. Estas colisões surgem de respostas do oráculo de assinatura: digamos que $(m_i, ID_i, PK_i, r_i, h_i, \sigma_i)$ seja gerado, onde $f(m_i, ID_i, PK_i, r_i)$ já está definido e não é igual a h_i. Isto pode acontecer em duas situações distintas:

1. se a consulta $f(m_i, ID_i, PK_i, r_i)$ já tiver sido feita ao oráculo aleatório; isto acontece com probabilidade no máximo $Q \cdot S \cdot 2/2^k \leq \epsilon/5$.

2. se o oráculo de assinaturas já gerou outra resposta para uma consulta $\mathcal{X}_j$, onde $(m_i = m_j, ID_i = ID_j, PK_i = PK_j, r_i = r_j)$ mas $h_i \neq h_j$; isto acontece com probabilidade no máximo $S^2 \cdot 2/2^k \leq \epsilon/10$.

Logo, a probabilidade geral de acontecerem colisões é menor que $3\epsilon/10$. Portanto,

$$\begin{aligned} \Pr[\mathcal{A} \text{ tem sucesso } \wedge \text{ não há colisões}] &\geq \Pr[\text{Sucesso } \mathcal{A}] - \Pr[\textit{colisões}] \\ &\geq \epsilon \cdot (1 - \frac{3}{10}) \\ &\geq 7\epsilon/10, \end{aligned}$$

o que é claramente maior do que $7Q/2^k$. Podemos então aplicar o Teorema 3 que terá probabilidade de sucesso $\epsilon \geq 1/9$ e tempo total $T' \leq 16QT \cdot 10/7\epsilon \leq 23QT/\epsilon$. ■

4.4 Revisão da segurança dos esquemas de CLS

Nesta seção revisaremos as propriedades de segurança mais importantes dos esquemas apresentados em §3.2. Apresentaremos algumas demonstrações de segurança para esquemas que anteriormente não as tinham. Também apresentaremos algumas análises de por que, mesmo com "demonstrações" de segurança apresentadas, alguns esquemas foram posteriormente quebrados. Não discutiremos alguns dos esquemas citados em §3.2 porque eles são inseguros e não tiveram qualquer demonstração de segurança apresentada (Gorantla & Saxena [2005]).

4.4.1 Al-Riyami & Paterson [2003]

Como observado em §3.2.1, o esquema apresentado por Al-Riyami & Paterson não tinha uma demonstração de segurança. Posteriormente, ele foi quebrado por Huang et al. [HSMZ05], que, além de um ataque, propuseram também uma correção que tornava o esquema seguro (chamemos esta segunda versão de `AP-2`). Eles apresentam uma demonstração da segurança de `AP-2`, mas esta demonstração tem dois problemas:

1. ela utiliza a Técnica de Reexecução de Oráculos, mas sem justificar precisamente por que era válido utilizar esta técnica: de fato, a demonstração do jeito que está escrita, aparentemente poderia ser diretamente aplicada ao Al-Riyami/Paterson original.

2. eles cometem o erro discutido em §4.2.1 de assumir que o valor secreto de chaves públicas substituídas pelo adversário podem ser recuperados. Este é um problema razoavelmente mais sério que o anterior pois invalida completamente a demonstração (enquanto a Técnica de Reexecução de Oráculos podia de fato ser utilizada, os autores só não justificaram o porquê).

Em [CD07b], nós analisamos melhor as propriedades do `AP-2`, mostramos uma simplificação para esta correção e demonstramos por que ela era válida: fica claro então, como exposto em §4.3, por que nossa versão do Al-Riyami/Paterson (`AP-3`), assim como o `AP-2`, podem ser demonstrados seguros através da Técnica de Reexecução de Oráculos: ambos são CL-Genéricos, ao contrário da versão original. Além disso, construímos a demonstração sem a suposição incorreta em relação à recuperação de valores secretos. Reproduziremos aqui então a demonstração de segurança do `AP-3`.

O principal teorema que queremos demonstrar é o seguinte:

Teorema 5. *O `AP-3` é EU-CMA seguro no modelo do oráculo aleatório se o GBDHP e o CDHP forem difíceis em* $\mathbb{G}_1$.

Demonstração. Para provar este teorema, começaremos provando lemas auxiliares que estabelecem a segurança do esquema contra adversários Tipo-I e Tipo-II sob ataques de identidade escolhida. Em seguida utilizamos um lema que reduz ataques gerais a ataques de identidade escolhida, e o teorema se torna então corolário destes lemas.

Lema 4. *Seja $\mathcal{A}_I$ um adversário Tipo-I que quebra o* `AP-3` *em tempo T_1 e com probabilidade não-desprezível ϵ_1 num ataque de identidade escolhida. Sejam q_S e q_{H_2} o número máximo de consultas feitas respectivamente ao oráculo de assinaturas e hash-H_2. Suponha que $\epsilon_1 \geq (10(q_S+1)(q_S+q_{H_2}))/2^k$. Então o GCDHP pode ser resolvido em tempo $T_1' \leq (120686 q_{H_1} T_1)/(\epsilon_1(1-\frac{1}{2^k}))$, onde k é o parâmetro de segurança.*

Demonstração. Construímos um algoritmo $\mathcal{D}_I$ que utiliza $\mathcal{A}_I$ para resolver instâncias $(P, aP, bP) \in \mathbb{G}_1^3$ do GCDHP. Relembrando, isso significa encontrar um par $(xP, XabP)$.

Na inicialização do sistema, $\mathcal{D}_I$ seleciona P como o gerador do grupo e faz $P_{pub} \leftarrow aP$. $\mathcal{D}_I$ escolhe aleatoriamente a identidade-alvo ID^* e executa $\mathcal{A}_I$, respondendo consultas de oráculo da forma a seguir:

ID-Hash[$H_1(ID_i)$].

1. se $ID_i = ID^*$:

   faz $Q_i = H_1(ID_i) = bP$, e $y_i = \perp$;

   senão:

   gera y_i aleatório e faz $Q_i = H_1(ID_i) = y_i P$
2. faz $P_i = x_i = \perp$ e salva a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;
3. finalmente, retorna Q_i.

Extração de Chave Parcial(ID_i).

1. Encontra a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;

   se não existe, ou se $y_i = \perp$, então $\mathcal{D}_I$ `ABORTA`.
2. caso contrário, retorna $D_{ID_i} = y_i P_{pub} = y_i(aP)$.

Perceba que $\mathcal{A}_I$ não pode consultar a chave parcial de ID^*.

Extração de Valor Secreto(ID_i).

1. Encontre a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;

   se não existe, $\mathcal{D}_I$ `ABORTA`;

2. se a chave pública foi substituída, então $\mathcal{D}_I$ `ABORTA`;
3. se $x_i = \perp$ (o valor secreto ainda não foi criado), escolha $x_i \stackrel{R}{\leftarrow} \mathbb{Z}_p$;
4. retorna x_i.

Extração de Chave Pública(ID_i).

1. Encontre a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;
   se não existe, se $y_i = \perp$ ou se P_{ID_i} foi substituído, `ABORTA`;
2. se $x_i = \perp$, execute `Extração de Valor Secreto(`ID_i`)` para gerar o valor secreto;
3. retorne $P_{ID_i} = \langle x_i P, x_i P_{pub} \rangle$.

Substituição de Chave Pública(ID_i,P'_i).

1. Encontre a tupla (ID_i, Q_i, P_i, x_i);
   se não existe, $\mathcal{D}_I$ `ABORTA`.
2. caso contrário $\mathcal{D}_I$ faz $x_i = \perp$ e $P_i = P'_i$.

Message-Hash[$H_2(M_j$,R_j,$P_{ID_j})$].

1. Se $H_2(M_j, R_j, P_{ID_j})$ ainda não foi definido:
   escolhe $h_j \stackrel{R}{\leftarrow} \mathbb{Z}_p$;
   faz $H_2(M_j, R_j, P_{ID_j}) = h_j$;
2. retorna $H_2(M_j, R_j, P_{ID_j})$.

Assinatura(ID_i,M_j).

1. Encontre a tupla $(ID_i, Q_i, \langle X_i, Y_i \rangle, y_i)$;
   se não existe, $\mathcal{D}_I$ `ABORTA`.
2. escolhe $S_i \stackrel{R}{\leftarrow} \mathbb{G}_1$ e $h_j \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$;
3. calcula $R_i = e(S_i, P)e(H_1(ID_i), -Y_i)^{h_j}$
4. se $H_2(M_j, R_t, Y_i)$ já está definido, `ABORTA`;
   caso contrário, faz $H_2(M_j, R_t, Y_i) = h_j$;

5. retorna a assinatura $\sigma = (S_i, h_j)$.

Estes procedimentos simulam um oráculo de assinaturas *forte*. Se $\mathcal{D}_I$ não abortar, $\mathcal{A}_I$ irá gerar uma falsificação $\gamma = (ID^*, m, S, h)$ com probabilidade ϵ_1. A probabilidade de $\mathcal{D}_I$ abortar é desprezível. Este esquema é CL-Genérico; logo, podemos utilizar a técnica de reexecução de oráculos (§4.3) para obter um par de falsificações (γ_1, γ_2) válidas para o mesmo valor de U. Logo, temos que:

$$\begin{aligned} U = e(S_1, P)e(H_1(ID^*), -Y^*)^{h_1} &= e(S_2, P)e(H_1(ID^*), -Y^*)^{h_2} \\ e(S_1, P)e(bP, -x^*aP)^{h_1} &= e(S_2, P)e(bP, -x^*aP)^{h_2} \\ e(S_1 - S_2, P) &= e(bP, -x^*aP)^{h_2 - h_1} \\ e(S_1 - S_2, P) &= e(-(h_2 - h_1)x^*abP, P) \end{aligned}$$

Seja $W = (h_2 - h_1)(S_1 - S_2) = x^*abP$. Lembrando que $X_i = x^*P$, temos que (X_i, W) é uma resposta válida para a nossa instância do GCDHP. ■

Lema 5. *Seja $\mathcal{A}_{II}$ um adversário Tipo-II que quebra o `AP-3` em tempo T_2 e com probabilidade não desprezível ϵ_2. Sejam q_S e q_{H_2} o número máximo de consultas feitas respectivamente ao oráculo de assinaturas e de Hash-H_1. Suponha que $\epsilon_2 \geq (10(q_S+1)(q_S+q_{H_2}))/2^k$. Logo, o CDHP pode ser resolvido em tempo $T_2' \leq (120686 q_{H_2} T_2)/(\epsilon_2(1 - \frac{1}{2^k}))$, onde k é o parâmetro de segurança.*

Demonstração. Construímos o algoritmo desafiante $\mathcal{D}_{II}$ que recebe $(P, aP, bP) \in \mathbb{G}_1^3$ como entrada e, utilizando $\mathcal{A}_{II}$, calcula abP. Na inicialização, $\mathcal{D}_{II}$ escolhe P como o gerador do grupo, escolhe $s \xleftarrow{R} \mathbb{Z}_p^*$ e faz $P_{pub} \leftarrow sP$. $\mathcal{D}_{II}$ escolhe a identidade-alvo ID^* e executa $\mathcal{A}_{II}$, respondendo consultas de oráculos com os procedimentos a seguir:

ID-Hash$[H_1(ID_i)]$**.**

1. se $ID_i = ID^*$:
   faz $Q_i = H_1(ID_i) = aP$, e $y_i = \perp$;
   senão:
   gera y_i aleatório e faz $Q_i = H_1(ID_i) = y_iP$;
2. faz $P_i = x_i = \perp$ e salva a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;
3. finalmente, retorna Q_i.

Extração de Chave Parcial(ID_i)**.**

1. Encontra a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;

se não existe, ou se $y_i = \perp$, então $\mathcal{D}_{II}$ `ABORTA`.

2. caso contrário, retorna $D_{ID_i} = sQ_i$.

Perceba que $\mathcal{A}_{II}$ é capaz de calcular estes valores sozinho.

Extração de Valor Secreto(ID_i).

1. Encontre a tupla $(ID_i, Q_i, P_i, y_i, x_i)$;
   se não existe, $\mathcal{D}_{II}$ `ABORTA`;
2. se a chave pública foi substituída, então $\mathcal{D}_{II}$ `ABORTA`;
3. se $x_i = \perp$ (o valor secreto ainda não foi criado), escolha $x_i \xleftarrow{R} \mathbb{Z}_p$;
4. retorna x_i.

Extração de Chave Pública(ID_i).

1. Encontre a tupla $(ID_i, Q_i, P_{ID_i}, y_i, x_i)$;
   se não existe, se $y_i = \perp$ ou se P_{ID_i} foi substituído, `ABORTA`;
2. se $ID_i = ID^*$, retorna $P_{ID_i} = \langle bP, s(bP)\rangle$;
3. se $x_i = \perp$, execute `Extração de Valor Secreto(`ID_i`)` para gerar o valor secreto;
4. retorne $P_{ID_i} = \langle x_i P, x_i P_{pub}\rangle$.

Substituição de Chave Pública(ID_i,P_i').

1. Encontre a tupla (ID_i, Q_i, P_i, x_i);
   se não existe, ou se $ID_i = ID^*$, $\mathcal{D}_{II}$ `ABORTA`.
2. $\mathcal{D}_{II}$ faz $x_i = \perp$ e $P_i = P_i'$.

Message-Hash[$H_2(M_j$,R_j,$P_{ID_j})$].

1. Se $H_2(M_j, R_j, P_{ID_j})$ ainda não foi definido:
   escolhe $h_j \xleftarrow{R} \mathbb{Z}_p$;
   faz $H_2(M_j, R_j, P_{ID_j}) = h_j$;
2. retorna $H_2(M_j, R_j, P_{ID_j})$.

Assinatura(ID_i,M_j).

1. Encontre a tupla $(ID_i, Q_i, \langle X_i, Y_i\rangle, y_i)$;
   se não existe, $\mathcal{D}_{II}$ `ABORTA`.
2. escolhe $S_i \stackrel{R}{\leftarrow} \mathbb{G}_1$ e $h_j \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$;
3. calcula $R_i = e(S_i, P)e(H_1(ID_i), -Y_i)^{h_j}$
4. se $H_2(M_j, R_t, Y_i)$ já está definido, `ABORTA`;
   caso contrário, faz $H_2(M_j, R_t, Y_i) = h_j$;
5. retorna a assinatura $\sigma = (S_i, h_j)$.

Novamente, estes procedimentos simulam um oráculo de assinaturas *forte*. Se $\mathcal{D}_{II}$ não abortar, $\mathcal{A}_{II}$ irá gerar uma falsificação $\gamma = (ID^*, m, S, h)$ com probabilidade ϵ_2. A probabilidade de $\mathcal{D}_{II}$ abortar é desprezível. Novamente utilizamos a Técnica de Reexecução de Oráculos para obter duas assinaturas, γ_1 e γ_2, em tempo $T_2' \leq \frac{120686 q_{H_1} T_2}{\epsilon_2(1-\frac{1}{2^k})}$. Temos então que:

$$\begin{aligned} U = e(S_1, P)e(H_1(ID^*), -Y^*)^{h_1} &= e(S_2, P)e(H_1(ID^*), -Y^*)^{h_2} \\ e(S_1, P)e(aP, -s^*bP)^{h_1} &= e(S_2, P)e(aP, -sbP)^{h_2} \\ e(S_1 - S_2, P) &= e(aP, -sbP)^{h_2 - h_1} \\ e(S_1 - S_2, P) &= e(-(h_2 - h_1)sabP, P); \end{aligned}$$

Seja $W = (h_2 - h_1)(S_1 - S_2)s^{-1} = abP$. Logo, W é a resposta para a nossa instância do CDHP. ■

Por último, utilizamos uma adaptação do lema de [CC03] para reduzir ataques gerais a ataques de identidade escolhida:

Lema 6. *Seja $\mathcal{A}$ um adversário que faz até q_H consultas ao hash de identidades e (T, ϵ)-quebra um esquema de CLS. Seja ID^* uma identidade-alvo escolhida aleatoriamente. Então, existe um adversário $\mathcal{A}'$ que (T', ϵ')-quebra o esquema para a identidade ID^*, para:*

$$T' \leq T, \quad \epsilon' \geq \epsilon(1 - \frac{1}{2^k})/q_H.$$

Demonstração. Não provaremos formalmente este lema, mas a intuição por trás do seu funcionamento é simples. Devido à aleatoriedade do hash de identidades, a probabilidade de uma falsificação correta ser gerada sem que a consulta da identidade seja feita ao hash é desprezível $(1 - \frac{1}{2^k})$: se temos então que q_H diferentes identidades foram consultadas, o desafiante $\mathcal{D}$ terá uma chance de $\frac{1}{q_H}$ de acertar a identidade da falsificação final. ■

O teorema é um corolário direto dos três lemas acima. ■

Provamos assim que `AP-3` é EU-CMA seguro: a mesma demonstração, com adaptações mínimas, poderia ser utilizada para o `AP-2`, mas como a proposta original (`AP`) não é CL-Genérico, a prova não pode ser adaptada a ela.

Todas as versões derivadas do Al-Riyami/Paterson original, e alguns outros esquemas de CLS que compartilham o seu procedimento de geração de chaves estão sujeitos a ataques *direcionados* de KGC maliciosa.

Ataque *direcionado* de KGC Maliciosa. Para executar este ataque, a KGC procede como a seguir:

1. durante a inicialização do sistema:
   - escolhe a identidade-alvo ID^*;
   - escolhe $\alpha \xleftarrow{R} \mathbb{Z}_p^*$;
   - faz com que $P = \alpha H_1(ID^*)$;
   - prossegue normalmente com o resto da inicialização.
2. Quando a chave pública $P_{ID^*} = (X_{ID^*}, Y_{ID^*})$ for disponibilizada, a KGC pode calcular a chave secreta correspondente:

$$S_{ID^*} = \alpha^{-1} \cdot Y_{ID^*}.$$

Este procedimento funciona porque a chave secreta completa é

$$S_{ID^*} = x \cdot D_{ID^*} = x \cdot s \cdot H_1(ID^*).$$

Como $Y_{ID^*} = x \cdot s \cdot P = x \cdot s \cdot \alpha \cdot H_1(ID^*)$, temos então que

$$\alpha^{-1} \cdot Y_{ID^*} = x \cdot s \cdot H_1(ID^*) = S_{ID^*}.$$

O esquema é vulnerável somente à versão dirigida do ataque de KGC maliciosa, o que, como explicado anteriormente (§4.2.3), não é um problema tão preocupante: se utilizarmos encapsulamento de chaves públicas para atingir o Nível de Segurança 3 de Girault, este ataque torna-se impossível.

4.4.2 Li, Chen & Sun [2005]

Como observado anteriormente, este esquema é praticamente igual ao `AP` original: sendo assim, é vulnerável ao mesmo tipo de ataque de KGC maliciosa (apenas o dirigido) e pode ser demonstrado seguro se for transformado num esquema CL-Genérico (colocando a chave pública junto com o hash do desafio aleatório e da mensagem): chamemos esta versão do esquema de `LCS-2`. Faremos aqui um esboço da demonstração de segurança deste esquema, uma vez que ela é bastante semelhante à apresentada na seção anterior para o `AP-3`

Teorema 6. *O* `LCS-2` *é EU-CMA seguro no modelo do oráculo aleatório se o GBDHP e o CDHP forem difíceis em* $\mathbb{G}_1$.

Esboço da Demonstração. O `LCS-2` é CL-Genérico: poderemos então utilizar a Técnica de Reexecução de Oráculos. A primeira observação importante é que, se obtivermos o par de assinaturas $(\gamma_1 = (r, h, m, \sigma), \gamma_2 = (r, h', m, \sigma'))$, temos que:

$$S_A = (\sigma - \sigma') \cdot (h' - h)^{-1}. \tag{4.2}$$

Logo, o par de assinaturas obtidos através da reexecução é suficiente para revelar a chave secreta do usuário-alvo.

Para ataques do Tipo I, reduzimos o GCDHP à falsificação. $\mathcal{D}$ recebe (P, aP, bP), faz $P_{pub} = aP$, e $H_1(ID^*) = bP$. A simulação de assinaturas aqui é um pouco mais complicada do que para o `AP-3`, mas ainda é possível:

1. escolha $h, x \overset{R}{\leftarrow} \mathbb{Z}_p^*$;
2. calcule $U = -hQ_{ID^*} + xP$;
3. faça $H_2(m, U, PK_{ID^*}) = h$, abortando caso já esteja definido;
4. calcule $\sigma = x.Y_{ID^*}$.

Esta assinatura é aceita uma vez que:

$$e(P, \sigma) = e(P, xY_{ID^*}) = e(Y_{ID^*}, xP) = e(Y_{ID^*}, hQ_{ID^*} - hQ_{ID^*} + xP) = e(Y_{ID^*}, hQ_{ID^*} + U).$$

O restante da adaptação é trivial e obteremos no final as duas falsificações. Podemos então utilizar a eq. (4.2) para calcular $S_{ID^*} = x_{ID^*}abP$ e a resposta do GCDHP será (X_{ID^*}, S_{ID^*}).

Para ataques de Tipo 2, reduzimos o CDHP à falsificação. $\mathcal{D}$ recebe (P, aP, bP), escolhe $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$, faz $P_{pub} = sP$, $H_1(ID^*) = aP$, e $P_{ID^*} = (bP, s \cdot bP)$. O procedimento anterior de simulação de assinaturas ainda funciona, e podemos obter $S_{ID^*} = s \cdot abP$. Como $\mathcal{D}$ conhece o valor de s, é fácil calcular a resposta para o nosso CDHP, $s^{-1}S_{ID^*}$. ■

A demonstração acima tem as mesmas propriedades da apresentada para o `AP-3`: reduções a partir dos mesmos problemas, mesma eficiência, oráculo forte de assinaturas, o que é bastante natural visto que o esquema em si é bastante parecido com o `AP`.

4.4.3 Zhang, Wong, Xu & Feng [2006]

Não há muito o que comentar em relação a este esquema, exceto que:

1. ele é seguro inclusive na presença de oráculos fortes de assinatura;

2. sua segurança está baseada diretamente no CDHP, não necessitando de suposições mais fortes como a do GCDHP;

3. ele *aparentemente* não é vulnerável a ataques de KGC maliciosa.[14]

4.4.4 Goya & Terada [2006]

Como já comentamos em §3.2.6, mostramos em [CD07b] que a proposta de Goya & Terada é insegura. Discutiremos aqui detalhadamente os problemas encontrados nesta demonstração e que levam ao ataque apresentado. Para facilitar nossa discussão, faremos um esboço da prova de segurança apresentada em [Goy06] a seguir.

Lema 7. *Seja $\mathcal{A}_I$ um adversário Tipo-I que quebra o Goya/Terada em tempo T_1 e com probabilidade não-desprezível ϵ_1 sob um ataque de identidade escolhida. Sejam q_S e q_{H_1} respectivamente o número máximo de consultas feitas ao oráculo de assinaturas e de hash de identidade. Suponha que $\epsilon_1 \geq ((10(q_S+1)(q_S+q_{H_1}))/2^k$. Então, o q-SDHP pode ser resolvido em tempo $T_1' \leq (120686 q_{H_1} T_1)/(\epsilon_1(1-\frac{1}{2^k}))$, onde k é o parâmetro de segurança.*

Contrói-se então um algoritmo desafiante $\mathcal{D}_I$ que usa $\mathcal{A}_I$ para resolver o q-SDHP. A entrada para $\mathcal{D}_I$ é $(P, Q, \alpha Q, \alpha^2 Q, \ldots, \alpha^q Q)$ e ele deve retornar um par $(c, \frac{1}{c+\alpha}P)$.

Na inicialização obtém-se $q-1$ pares da forma $(w_i, \frac{1}{\alpha+w_i}P)$, e calcula-se $Q_{pub} = \alpha Q$. $\mathcal{D}_I$ executa $\mathcal{A}_I$ com uma identidade-alvo ID^* escolhida aleatoriamente.

Consultas a oráculos são simuladas de forma intuitiva, utilizando os pares $(w_i, \frac{1}{\alpha+w_i}P)$ computados para responder a consultas `ID-Hash` e `RevelaChaveParcial` ($H_1(ID_i) = w_i, D_{ID_i} = \frac{1}{\alpha+w_i}P$), a não ser que ID^* esteja na consulta: neste caso, uma resposta aleatória w^*, diferente de qualquer w_i dos pares pré-computados, é escolhida e $(H_1(ID^*) = w^*, D_{ID^*} = \perp)$.

Se $\mathcal{D}_I$ não aborta durante a simulação, $\mathcal{A}_I$ gera a falsificação $\gamma_1 = (S_1, h_1)$. A Técnica de Reexecução de Oráculos pode então ser usada para obter uma segunda falsificação

[14] Não demonstramos este fato, mas, para a KGC, conhecer o logaritmo discreto de $H_1(ID^*)$ não parece trazer qualquer poder adicional.

$\gamma_2 = (S_2, h_2)$, tal que:

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

e

$$Y = (h_1 - h_2)^{-1}(S_1 - S_2) = t^*(1/(\alpha + w^*))P,$$

que é *quase* o resultado que buscamos, exceto pelo fator t^*: esta é o valor secreto correspondente à chave pública N^* sob a qual a falsificação é válida, que potencialmente foi substituída durante o ataque. Em [Goy06] supõe-se que este valor pode ser recuperado de $\mathcal{A}_I$; então é simples calcular $W = t^{*-1}Y$, e prosseguir com a técnica primeiro apresentada em [BLMQ05] (e também utilizada em [Goy06]) para resolver o q-SDH.

O problema é que, como discutimos inicialmente em §4.2.1, esta suposição não é válida: ela equivale a supor que a única maneira pela qual $\mathcal{A}_I$ poderia calcular um N^* é primeiro escolhendo um t^* e depois calculando $N^* = g^{t^*}$, o que obviamente não é verdade.

De fato, no ataque apresentado em §3.2.6 temos $N^* = g^{x_{ID_i}(s + H_1(ID_i))}$, mas o adversário não é capaz de calcular $x_{ID_i}(s + H_1(ID_i))$.

4.4.5 Yap, Heng & Goi [2006]

Agora que já discutimos com bom nível de detalhes as questões envolvidas em demonstrações de segurança de esquemas de CLS, podemos entender melhor por que o YHG, apesar de dispor de uma demonstração de segurança e a exemplo do Goya & Terada, é inseguro. O primeiro grande equívoco cometido pelos autores foi utilizar a Técnica de Reexecução de Oráculos em um esquema que não é CL-Genérico: isso basta para tornar a demonstração oferecida por eles inválida. No entanto, se o problema fosse apenas este, seria fácil de "consertar" o esquema: bastaria colocarmos a chave pública no hash da mensagem.

O outro problema existente na demonstração não é tão facilmente corrigido: implicitamente, os autores assumem que o desafiante $\mathcal{D}$ pode recuperar o valor secreto associado à chave pública utilizada na falsificação (§4.2.1 problema discutido em detalhes também na sub-seção anterior no contexto do Goya & Terada): no ataque apresentado em §3.2.4, o adversário é incapaz de calcular o valor secreto correspondente à chave pública. Isto invalida qualquer chance do esquema ser facilmente adaptado para uma versão segura.

4.4.6 Liu, Au & Susilo [2006]

Como já mencionado em §3.2.7, a grande peculiaridade deste esquema é ser seguro no Modelo Padrão. Além disso, ele é seguro mesmo quando o oráculo de assinaturas é forte e não incorre em qualquer dos erros "comuns" em provas de segurança discutidos nesta seção.

4.4.7 Choi et al. [2007], Du & Wen [2007]

Ambos os esquemas extremamente eficientes propostos recentemente têm problemas em suas demonstrações de segurança: ambos fazem a suposição de que valores secretos podem ser recuperados do adversário após a simulação. Como já discutimos extensamente neste texto, esta suposição é errada e potencialmente leva a vulnerabilidades no esquema. No entanto, não conseguimos ainda encontrar qualquer ataque a um destes esquemas[15].

A situação destes esquemas é então incerta: por um lado são dois esquemas extremamente eficientes (os mais eficientes disponíveis na literatura) mas cujo nível de segurança é desconhecido. Em relação a ataques de KGC maliciosa, no entanto, ambos os esquemas parecem invulneráveis.

4.5 Conclusão

Neste capítulo fizemos uma discussão extensa sobre a segurança de esquemas de assinaturas digitais sem certificados, apresentando um pouco de perspectiva histórica da área de Segurança Demonstrável e discutindo com boa profundidade os modelos e formalizações mais importantes de CLS. Apresentamos alguns resultados originais, como a análise da Técnica de Reexecução de Oráculos quando aplicada à esquemas de CLS (§4.3), as análises de demonstrações de segurança (em pelo menos um caso levando diretamente a ataques ao esquema, e em outro caso levando a uma versão mais eficiente e segura que o original) e construções de novas demonstrações para esquemas que anteriormente não as tinham (§4.4).

[15] Note que os dois esquemas são muito semelhantes e, provavelmente, um ataque a um deles indicaria um caminho promissor para se descobrir um ataque ao outro.

Capítulo 5

Uma proposta de CLS

Neste capítulo apresentaremos a proposta de um esquema de assinaturas sem certificados com agregação. Este esquema requer o cálculo de três emparelhamentos para a verificação de assinaturas, sendo ainda bastante eficiente se comparado com os outros esquemas de CLS disponíveis na literatura. Ele é especialmente eficiente para a agregação de Tipo 2, onde um usuário assina múltiplas mensagens, pois o número de emparelhamentos para a verificação não depende da quantidade de mensagens agregadas, mantendo-se sempre em três. No caso mais geral, onde assinaturas de vários usuários podem ser agregadas, o esquema requer o cálculo adicional de dois emparelhamentos para cada novo assinador. Em §5.2 apresentamos o esquema e discutimos brevemente alguns aspectos relevantes à sua compreensão. Em seguida (§5.3 e §5.4) demonstramos a segurança da nossa proposta, respectivamente para assinaturas individuais e para assinaturas agregadas, reduzindo-a ao Diffie-Hellman Computacional.

5.1 Agregação de Assinaturas

A noção de agregação de assinaturas foi proposta por Boneh et al. em [BGLS03]. Neste artigo é descrito um esquema de assinaturas que permite a agregação de n assinaturas em uma única assinatura curta. Esta "assinatura agregada", juntamente com as n mensagens originais, são suficientes para convencer um verificador que os n assinadores de fato assinaram as n mensagens. Existem muitas variações do conceito de "agregação", como por exemplo:

- **assinaturas agregadas seqüenciais,** onde o n-ésimo assinador deve receber o agregado referente aos $n-1$ outros assinadores e "adicionar" a sua assinatura a ele;
- **assinaturas agregadas irrestritas,** onde é permitida a existência de mensagens repetidas no agregado (o que não era o caso no esquema original de [BGLS03]);

- **assinaturas agregadas parciais,** onde a assinatura agregada não tem tamanho constante, mas cresce linearmente com a quantidade de assinaturas individuais que compõem o agregado (ainda que se mantendo menor do que n assinaturas distintas).

O esquema que propomos a seguir é de agregação irrestrita.

5.2 Descrição do Esquema

`Inicializar.` Seja k o parâmetro de segurança;
sejam $\mathbb{G}$ e $\mathbb{G}_T$ grupos tais que:

- A suposição do *CDH* vale em $\mathbb{G}$;
- $p \leftarrow |\mathbb{G}| = |\mathbb{G}_T|$
- $\exists\, e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ emparelhamento bilinear admissível.

Seja $P \in \mathbb{G}$ um gerador arbitrário de $\mathbb{G}$; escolha as seguintes funções de hash:

- $H_1 : \{0,1\}^* \times \mathbb{G} \rightarrow \mathbb{Z}_p^*$;
- $H_2,\ H_3 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}$.

Escolha $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$; seja $P_{pub} = sP$;
retorne $mpk = \langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub} \rangle$ e $msk = s$.

`ExtrairChaveParcial`(ID_i).
$r_i \overset{R}{\leftarrow} \mathbb{Z}_p^*$; $R_i = r_i P$;
$d_{ID_i} = (r_i + sH_1(ID_i, R_i)) \bmod p$;
retorne $\langle d_{ID_i}, R_i \rangle$.

`GerarChavesUsuário`(ID_i).
$x_{ID_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$;
$P_{ID_i} = x_{ID_i} P$;
retorne $\langle P_{ID_i}, x_{ID_i} \rangle$.

`Assinar.`
Retorne $\sigma = d_{ID_i} H_2(M_i, ID_i, P_{ID_i}, R_i) + x_{ID_i} H_3(M_i, ID_i, P_{ID_i}, R_i)$.

`Verificar.`
Seja $h_1 = H_1(ID_i, R_i)$;
seja $H_2 = H_2(M_i, ID_i, P_{ID_i}, R_i)$;
seja $H_3 = H_3(M_i, ID_i, P_{ID_i}, R_i)$;
retorne `ACEITA` se e somente se

$$e(P, \sigma) \overset{?}{=} e(H_2, R_i + h_1 P_{pub}) e(H_3, P_{ID_i}).$$

5.2.1 Agregação de assinaturas

Os algoritmos que permitem a agregação de assinaturas e a verificação de um agregado são bastante simples, como pode-se observar a seguir:

`Agregar.` Dada a lista de assinaturas que devem ser agregadas $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, calcular:

$$\gamma = \sum_{\forall i} \sigma_i.$$

`VerificarAgregado.` Sejam γ um agregado, $\mathbb{U} = \{u_1, u_2, \ldots, u_n\}$ a lista de assinadores, e $\mathbb{M} = \{M_1, M_2, \ldots, M_n\}$ a lista de mensagens (note que podem haver repetições em ambas as listas).
Supomos aqui que u_i assinou m_i e, conseqüentemente $|\mathbb{U}| = |\mathbb{M}|$.
Para cada usuário $u_i \in \mathbb{U}$ defina a lista de mensagens assinadas por u_i, $M_{u_i} = \{M_j \in \mathbb{M}, u_j = u_i\}$. Seja $\mathbb{U}^*$ o conjunto de usuário *distintos* em $\mathbb{U}$ e sejam

$$\begin{aligned} h_{u_i} &= H_1(ID_{u_i}, R_{u_i}), \\ \gamma_1 &= \prod_{\forall u_i \in \mathbb{U}^*} e(\sum_{\forall M_i \in \mathbb{M}_{u_i}} H_2(M_{u_i}, ID_{u_i}, P_{ID_{u_i}}, R_{u_i}), R_{u_i} + h_{u_i} P_{pub}), \\ \gamma_2 &= \prod_{\forall u_i \in \mathbb{U}^*} e(\sum_{\forall M_i \in \mathbb{M}_{u_i}} H_3(M_{u_i}, ID_{u_i}, P_{ID_{u_i}}, R_{u_i}), P_{ID_{u_i}})). \end{aligned}$$

O agregado é aceito se e somente se

$$e(P, \gamma) = \gamma_1 \gamma_2. \tag{5.1}$$

Perceba que o procedimento de verificação acima se aplica a todos os tipos de agregação e implica p.ex. que, para agregação do Tipo 2 é necessário apenas o cálculo de três emparelhamentos, independentemente do número de mensagens agregadas.

5.2.2 Chaves públicas

É interessante notar que, ao contrário dos outros esquemas de assinaturas sem certificados, as chaves públicas desta nossa proposta têm dois componentes independentes: um gerado pelo usuário (P_{ID_i}), e um gerado pela KGC (R_i). Sempre que nos referimos à chave pública falamos implicitamente destes dois componentes considerados em conjunto: sendo assim quando falarmos que um adversário pode substituir chaves públicas, queremos dizer que ele pode substituir qualquer dos dois componentes da chave (ou ambos). Isto é especialmente importante porque no trabalho que inspirou este esquema, o IBS de [Her06], a possibilidade de substituição dos R_i não é considerada: o autor não deixa claro se supõe

que os R_i são autenticados (o que não faria muito sentido em se tratando de um IBS, cuja principal vantagem é a ausência de qualquer tipo de certificação explícita) ou se simplesmente não toma a substituição dos R_i como um ataque plausível. Provamos aqui, no entanto, que o esquema é seguro mesmo que o adversário possa substituir os R_i.

5.3 Segurança do esquema de assinaturas

Nesta seção analisamos a segurança do esquema de assinaturas "simples", e deixamos a análise da segurança da agregação de assinaturas para §5.4.

5.3.1 Adversários Tipo I

Queremos provar que nosso esquema é seguro contra adversários Tipo I, como na definição 8. Usaremos uma seqüência de jogos para provar que, se existe um adversário $\mathcal{A}_I$ capaz de quebrar o esquema com probabilidade não-desprezível, nós podemos construir um desafiante $\mathcal{D}_I$ que resolve o problema de Diffie-Hellman Computacional *(CDHP)* também com probabilidade não-desprezível. Como assumimos que o CDHP é difícil em $\mathbb{G}$, isto mostra que $\mathcal{A}_I$ não pode existir. Lembramos que todas as nossas demonstrações serão feitas no Modelo do Oráculo Aleatório.

Teorema 7. *Se existe um adversário de Tipo I $\mathcal{A}_I$ capaz de quebrar a segurança EU-CMA do nosso esquema com probabilidade não-desprezível $\lambda(k)$, então existe um algoritmo $\mathcal{W}$ que resolve o CDHP no grupo $\mathbb{G}$ com probabilidade não desprezível*

$$\Pr[\mathcal{W}] \approx \left(\frac{\lambda(k)^2}{q_u^2 q_m^2 q_{h_1}} \right),$$

onde $\Pr[\mathcal{W}]$ *é a probabilidade do CDHP ser resolvido.*

Demonstração. Suponha que um tal $\mathcal{A}_I$ exista. Durante o ataque, $\mathcal{A}_I$ terá acesso aos seguintes oráculos:

- $\mathbf{H_1}(ID, R), \mathbf{H_2}(M_I D, P_{ID}, R), \mathbf{H_3}(M, ID, P_{ID}, R)$,
- **RevelaChavePública**(ID),
- **RevelaChaveParcial**(ID),
- **RevelaValorSecreto**(ID, R', P'_{ID}),
- **SubstituiChavePública**(ID),
- **Assina**(M, ID).

Estes oráculos devem ser simulados por $\mathcal{D}_I$. O jogo terá então a seguinte estrutura:

1. Seja k o parâmetro de segurança; $\mathcal{D}_I$ gera (mpk, msk) apropriados;

2. $\mathcal{A}_I$ é executado recebendo mpk como entrada;

3. $\mathcal{A}_I$ faz no máximo q_{H_1}, q_{H_2} e q_{H_3} consultas respectivamente aos oráculos $H_1(.)$, $H_2(.)$ e $H_3(.)$, e q_s consultas ao oráculo de assinatura.

4. Após um número polinomial de passos $T = poli(k)$, $\mathcal{A}_I$ retorna

$$\varsigma = \langle S, (\sigma^*, M^*, ID^*, P^*_{ID}, R^*)\rangle.$$

5. A Probabilidade de sucesso de $\mathcal{A}_I$ é $\Pr[\mathcal{A}_I] = \Pr[S = 1] = \lambda(k)$

O adversário vence o jogo se $\lambda(k)$ é não desprezível em relação a k. Nós descreveremos a seguir uma série de jogos relacionados entre si, sendo que o último deles descreve um $\mathcal{D}_I$ capaz de resolver o CDHP com probabilidade relacionada a $\lambda(k)$.

Jogo 0. Este primeiro jogo é uma simulação direta da descrição feita acima. $\mathcal{D}^0_I$ recebe uma instância do CDHP: a descrição do grupo $\mathbb{G}$ e a tupla $\langle P, aP, bP\rangle \in \mathbb{G}^3$.
$\mathcal{D}^0_I$ escolhe então $(\mathbb{G}_T, e)$ tais que:

- $|\mathbb{G}| = |\mathbb{G}_T|$;
- $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ é um emparelhamento admissível.

Seja $p = |\mathbb{G}| = |\mathbb{G}_T|$. $\mathcal{D}^0_I$ escolhe $s \xleftarrow{R} \mathbb{Z}^*_p$ e faz $P_{pub} = sP$.
$\mathcal{D}^0_I$ executa $\mathcal{A}_I$ com $\langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub}\rangle$ como entrada e simulando os oráculos como a seguir:

- $\mathbf{H_1}(ID_i, R_i)$.
  Se já não foi definido, escolhe $h_{1_i} \xleftarrow{R} \mathbb{Z}^*_p$ e faz $H_1(ID_i, R_i) = h_{1_i}$;
  retorne $H_1(ID_i, R_i)$.
- $\mathbf{H_2}(M_i, ID_i, P_{ID_i}, R_i)$.
  Se já não foi definido, escolhe $h_{2_i} \xleftarrow{R} \mathbb{Z}^*_p$ e faz $H_2(M_i, ID_i, P_{ID_i}, R_i) = h_{2_i}P$;
  retorne $H_2(M_i, ID_i, P_{ID_i}, R_i)$.
- $\mathbf{H_3}(M_i, ID_i, P_{ID_i}, R_i)$.
  Se já não foi definido, escolhe $h_{3_i} \xleftarrow{R} \mathbb{Z}^*_p$ e faz $H_3(M_i, ID_i, P_{ID_i}, R_i) = h_{3_i}P$;
  retorne $H_3(M_i, ID_i, P_{ID_i}, R_i)$.
- **RevelaValorSecreto**(ID_i).
  Se já não foi definido, escolhe $x_{ID_i} \xleftarrow{R} \mathbb{Z}^*_p$ e faz $P_{ID_i} = x_{ID_i}P$;
  retorne x_{ID_i}.
- **RevelaChaveParcial**(ID_i).
  Se já não foi definida, escolhe $r_i \xleftarrow{R} \mathbb{Z}^*_p$ e faz $R_i = r_iP$;
  calcule $d_{ID_i} = r_i + sH_1(ID_i, R_i) \pmod p$; retorne (d_{ID_i}, R_i).

- **RevelaChavePública**(ID_i).
  Se P_{ID_i} ainda não foi definida, invoca **RevelaValorSecreto**($\mathbf{ID_i}$);
  Se R_i ainda não foi definida, invoca **RevelaChaveParcial**($\mathbf{ID_i}$);
  retorne (P_{ID_i}, R_i).
- **SubstituiChavePública**(ID_i, P'_{ID_i}, R'_i).
  Faz $P_{ID_i} = P'_{ID_i}$ e $R_i = R'_i$.
- **Assina**(M_i, ID_i).
  Seja $h_{1_i} = H_1(ID_i, R_i)$;
  seja $h_{2_i}P = H_2(M_i, ID_i, P_{ID_i}, R_i)$;
  seja $h_{3_i}P = H_3(M_i, ID_i, P_{ID_i}, R_i)$;
  retorne $\sigma_i = h_{2_i}R_i + h_{2_i}h_{1_i}P_{pub} + h_{3_i}P_{ID_i}$.

Perceba que os algoritmos acima permitem a implementação de um oráculo de assinaturas *forte*, pois as assinaturas podem ser geradas corretamente mesmo que a chave pública do usuário tenha sido subtituída. Como mostrado abaixo, as assinaturas são corretamente simuladas.

$$\begin{aligned} e(P, \sigma_i) &= e(P, h_{2_i}R + h_{2_i}h_{1_i}P_{pub} + h_{3_i}P_{ID_i}) \\ &= e(P, h_{2_i}(R + h_{1_i}P_{pub}))e(P, h_{3_i}P_{ID_i}) \\ &= e(h_{2_i}P, R + h_{1_i}P_{pub}))e(h_{3_i}P, P_{ID_i}) \\ &= e(H_2(M_i, ID_i, P_{ID_i}, R_i), R + H_1(ID_i, R_i)P_{pub}))e(H_3(M_i, ID_i, P_{ID_i}, R_i), P_{ID_i}). \end{aligned}$$

Após a simulação do ataque, $\mathcal{A}_I$ deve retornar a tupla $\langle S, (\sigma^*, M^*, ID^*, P_{ID_i}, R^*)\rangle$. Neste jogo inicial, $\mathcal{D}_I^0$ tem sucesso exatamente quando $\mathcal{A}_I$ retorna $S = 1$, portanto

$$\Pr[\mathcal{D}_I^0] = \Pr[\mathcal{A}_I] = \Pr[S = 1] = \lambda(k).$$

Jogo 1. Agora $\mathcal{D}_I^1$ usa também a segunda entrada para o CDHP no jogo. Ao invés de escolher uma chave mestra secreta s e então calcular $P_{pub} = sP$, $\mathcal{D}_I^1$ faz $P_{pub} = aP$. Isso implica que $\mathcal{D}_I^1$ não conhece a chave mestra e, portanto, não é capaz de calcular chaves parciais como antes. Logo, $\mathcal{D}_I^1$ tomará precauções especiais no cálculo de $H_1(.)$ para garantir que é capaz de computar chaves parciais corretamente. Os dois oráculos que mudam, em relação ao Jogo 0, são:

- $\mathbf{H_1}(ID_i, R_i)$.
  Se a chave parcial de ID_i ainda não foi calculada,

  invoca **RevelaChaveParcial**(ID_i)

  Se $H_1(ID_i, R_i)$ ainda não está definido:

escolhe $h_{1_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;
faz $H_1(ID_i, R_i) = h_{1_i}P$.

Retorne $H_1(ID_i, R_i)$.

- **RevelaChaveParcial**(ID_i).
  Se a chave ainda não está definida

  escolhe $d_{ID_i}, h_{1_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;
  calcula $R_i = d_{ID_i}P - h_{1_i}P_{pub}$;
  define $H_1(ID_i, R_i) = h_{1_i}$.

  Retorne (d_{ID_i}, R_i)

Esta mudança não é perceptível para $\mathcal{A}_I$; as respostas de ambos os oráculos continuam uniformemente distribuídas em $\mathbb{Z}_p^*$. Portanto $\mathcal{A}_I$ deve se comportar da mesma maneira que no Jogo 0 e teremos:

$$\Pr[\mathcal{D}_I^1] = \Pr[\mathcal{D}_I^0] = \lambda(k).$$

Jogo 2. Este jogo é idêntico ao Jogo 1 exceto que $\mathcal{D}_I^2$ escolhe um par (ID^t, R^t) alvo e só tem sucesso caso $(ID^* = ID^t)$ e $(R^t = R^*)$. Um detalhe importante é que $\mathcal{D}_I^2$ não pode saber uma chave parcial válida para (ID^t, R^t); caso uma tenha sido gerada, $\mathcal{D}_I^2$ deve falhar. Seja q_u o número máximo de pares (ID_i, R_i) consultado durante toda a execução do jogo. Um limite superior bastante conservador para q_u é:

$$q_u \leq q_{H_1} + q_{H_2} + q_{H_3} + q_S.$$

Antes do jogo começar, $\mathcal{D}_I^2$ escolhe $t \stackrel{R}{\leftarrow} [1, q_u]$. Alteramos então o oráculo $H_1(.)$ da seguinte forma:

- $\mathbf{H_1}(ID_i, R_i)$
  Se a chave parcial de ID_i ainda não foi calculada,

  invoca **RevelaChaveParcial**(ID_i)

  Se esta é a t-ésima consulta distinta

  faz $ID^t = ID_i$ e $R^t = R_i$;
  se a chave parcial conhecida para ID^t inclui R^t, `FALHA`

  Se $H_1(ID_i, R_i)$ ainda não está definido:

  escolhe $h_{1_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$;
  faz $H_1(ID_i, R_i) = h_{1_i}P$.

  Retorne $H_1(ID_i, R_i)$.

Esta mudança só afeta $\mathcal{A}_I$ se $\mathcal{D}_I^2$ tiver que abortar (porque a chave parcial conhecida de ID^t inclui R^t). Mas isso acontece apenas com baixíssima probabilidade. Seja F_1 o evento em que $\mathcal{D}_I^2$ aborta neste ponto; temos $\Pr[F_1] = \frac{1}{p}$. Seja ainda F_2 o evento no qual $\mathcal{A}_I$ não faz a consulta $H_1(ID^*, R^*)$. Como $H_1(.)$ é um oráculo aleatório, a chance da falsificação ser válida sem que o a consulta tenha sido feita é de apenas $\frac{1}{p}$. Redefinimos então a saída de $\mathcal{D}_I^2$ para

$$\text{saída}_{\mathcal{D}_I^2} = \begin{cases} \langle 0, \perp \rangle, & \text{se } S = 1 \wedge (ID^* \neq ID^t \vee R^* \neq R^t) \\ \text{saída}_{\mathcal{A}_I}, & \text{c.c.} \end{cases}$$

Sendo assim, a probabilidade de sucesso de $\mathcal{D}_I^2$ é significativamente menor que a de $\mathcal{D}_I^1$; o par (ID^t, R^t) tem que ser adivinhado corretamente. Temos então:

$$\Pr[\mathcal{D}_I^2] = \Pr[\mathcal{D}_I^1 \wedge \overline{F_1} \wedge \overline{F_2} \wedge (ID^* = ID^t \wedge R^* = R^t)] = \lambda(k)(1 - 1/p)^2 \frac{1}{q_u}.$$

Logo, se $\lambda(k)$ é não-desprezível, $\Pr[\mathcal{D}_I^2]$ também o será.

Jogo 3. No Jogo 3, $\mathcal{D}_I^3$ escolhe também uma tupla $(M^t, ID^t, P_{ID^t}, R^t)$ alvo. Seja q_m o número máximo de tuplas distintas $H_2(M_i, ID^t, P_{ID^t}, R^t)$ consultadas ao longo da execução do jogo. Perceba que neste momento os valores de ID^t e R^t já foram escolhidos, como no Jogo 2. No Jogo 3 iteramos adicionalmente sobre todos os valores válidos de M_i e P_{ID^t} consultados ao longo da execução. Sendo assim, um limite superior bastante conservador para q_m é $q_m \leq q_{H_2} + q_s$. Antes do jogo começar, $\mathcal{D}_I^3$ escolhe um valor $u \xleftarrow{R} [1, q_m]$ e o oráculo $H_2(.)$ é alterado para:

- $\mathbf{H_2}(M_i, ID_i, P_{ID_i}, R_i)$. Se esta é a u-ésima consulta distinta

  faz $M^u = M_i, ID^u = ID_i, P_{ID^u} = P_{ID_i}, R^u = R_i$

  Se já não foi definido, escolhe $h_{2_i} \xleftarrow{R} \mathbb{Z}_p^*$ e faz $H_2(M_i, ID_i, P_{ID_i}, R_i) = h_{2_i} P$;
  retorne $H_2(M_i, ID_i, P_{ID_i}, R_i)$.

Novamente, essa mudança não tem efeitos perceptíveis por $\mathcal{A}_I$, permanecendo o comportamento deste igual ao do Jogo 2. Redefinimos a saída de $\mathcal{D}_I^3$ para:

$$\text{saída}_{\mathcal{D}_I^3} = \begin{cases} \langle 0, \perp \rangle, & \text{se } S = 1 \wedge (ID^* \neq ID^t \vee R^* \neq R^t) \\ \langle 0, \perp \rangle, & \text{se } S = 1 \wedge (M^* \neq M^u \vee ID^* \neq ID^u \vee P_{ID^*} \neq P_{ID^u} \vee R^* \neq R^u) \\ \text{saída}_{\mathcal{A}_I}, & \text{c.c.} \end{cases}$$

A probabilidade de sucesso de $\mathcal{D}_I^3$ será menor ainda que a de $\mathcal{D}_I^2$ pois a tupla $(M^u, ID^u, P_{ID^u}, R^u)$ também tem que ser adivinhada. Definimos então

S_2^* como o evento em que $(ID^* = ID^t e R^* = R^t)$

S_3^* como o evento em que $(M^* = M^u, ID^* = ID^u, P_{ID^*} = P_{ID^u} e R^* = R^u)$

Temos então:

$$\begin{aligned}
\Pr[S_2^*] &= 1/q_u \left(1 - \frac{1}{p}\right)^2; \\
\Pr[S_3^*] &= 1/q_m \left(1 - \frac{1}{p}\right); \\
\Pr[\mathcal{D}_I^3] &= \Pr[\mathcal{D}_I^1 \wedge S_2^* \wedge S_3^*] = \Pr[\mathcal{D}_I^2 \wedge S_3^*] = \lambda(k) \left(\frac{1}{q_u}\right) \left(\frac{1}{q_m}\right) \left(1 - \frac{1}{p}\right)^3.
\end{aligned}$$

Logo, se $\lambda(k)$ é não-desprezível, $\Pr[\mathcal{D}_I^3]$ também o é.

Jogo 4. Finalmente, $\mathcal{D}_I^4$ usa a terceira entrada do CDHP, definindo $H_2(M^u, ID^u, P_{ID^u}, R^u) = bP$. Esta mudança não altera a distribuição do oráculo $H_2(.)$, sendo portanto imperceptível para $\mathcal{A}_I$. O único problema é que $\mathcal{D}_I^4$ não é mais capaz de forjar assinaturas nesta tupla pois não conhece mais o logaritmo discreto de $H_2(M^u, ID^u, P_{ID^u}, R^u)$. Mas isto não representa um problema, pois $\mathcal{D}_I^4$ só teria sucesso em casos em que a falsificação gerada por $\mathcal{A}_I$ é de $H_2(M^u, ID^u, P_{ID^u}, R^u)$, casos em que por definição esta tupla não pode ter sido consultada ao oráculo de assinaturas. Sendo assim, esta mudança não altera a probabilidade de sucesso de $\mathcal{D}_I^4$, permanecendo igual à do Jogo 3:

$$\begin{aligned}
\Pr[\mathcal{D}_I^4] &= \Pr[\mathcal{D}_I^3] \\
&= \Pr[\mathcal{D}_I^1 \wedge S_2^* \wedge S_3^*] = \Pr[\mathcal{D}_I^2 \wedge S_3^*] \\
&= \lambda(k) \left(\frac{1}{q_u}\right) \left(\frac{1}{q_m}\right) \left(1 - \frac{1}{p}\right)^3.
\end{aligned}$$

Analisemos então a saída de $\mathcal{D}_I^4$ em caso de sucesso:

$$\begin{aligned}
e(P, \sigma^*) &= e(H_2(.), R^* + H_1(.)P_{pub}) e(H_3(.), P_{ID^*}) \\
&= e(bp, R^* + h_{1_i} aP) e(h_{3_i} P, P_{ID^*}).
\end{aligned}$$

Podemos então definir $W = \sigma^* - h_{3_i} P_{ID^*}$ e notar que:

$$e(P, W) = e(bP, R^* + h_{1_i} aP).$$

Ou seja, como conhecemos o valor de h_{1_i}, W *quase* nos dá a resposta para o CDHP; mas infelizmente não conhecemos r, tal que $rP = R^*$ [1]. Para resolvermos este problema a

[1] Perceba que, se r tal que $rP = R^*$ fosse conhecido teríamos $e(P, W) = e(bP, rP + h_{1_i} aP)$, e a resposta do CDHP seria $W' = h_{1_i}^{-1}(W - rbP)$.

técnica de *Oracle Replay*, como discutida em §4.3, será utilizada. Enunciaremos informalmente no que consiste esta *reexecução de oráculos* e a utilizaremos para finalizar esta prova. Na seção seguinte provaremos formalmente que esta técnica pode ser usada no contexto da prova atual.

Informalmente, a técnica de reexecução de oráculos consiste em tomar a execução de um jogo que teve sucesso e reexecutá-la utilizando as mesmas escolhas aleatórias (para o adversário) e dando as mesmas respostas para todas as consultas a oráculos *até um certo momento*: quando o adversário fizer uma consulta previamente escolhida, digamos Q^*, a simulação deixa de ser idêntica à anterior; daí em diante as escolhas aleatórias passam a ser feitas de maneira independente. A esperança é que esta segunda execução, parcialmente copiada da primeira, também tenha sucesso e guarde algumas (bem escolhidas) semelhanças com a primeira, mas, por outro lado não seja *exatamente* igual. Especificamente, no caso da presente demonstração, gostaríamos de obter, através de uma reexecução uma segunda falsificação $\varsigma' = \langle \sigma', M', ID', P'_{ID}, R' \rangle$ tal que $M' = M, ID' = ID, P'_{ID} = P_{ID}, R' = R$, mas $H_2(M, ID, P_{ID}, R) \neq H'_2(M', ID', P'_{ID}, R')$, onde a falsificação resultante da primeira execução é $\varsigma = \langle \sigma, M, ID, P_{ID}, R \rangle$. Ou seja, a consulta em que as respostas do oráculo começam a divergir na segunda execução é exatamente a da tupla que foi alvo da falsificação. Sendo assim fazemos $W' = \sigma' - h'_{3_i} P'_{ID}$ e obtemos:

$$\begin{aligned} e(P, W' - W) &= \frac{e(bP, R^*)e(bP, H'_1(.)aP)}{e(bP, R^*)e(bP, H_1(.)aP)} \\ &= e(bP, (H'_1 - H_1)aP). \end{aligned}$$

Logo, a resposta do CDHP é $Y = (W' - W)/(H'_1 - H_1) = abP$. Resta-nos analisar qual a probabilidade de sucesso do Jogo 4 após a utilização da técnica de execução de oráculo. Fazemos a seguinte alegação:

Alegação 1. *Seja $\epsilon = \lambda(k) - \frac{1}{p}$, onde $\lambda(k)$ é a probabilidade de sucesso de $\mathcal{A}_I$ no Jogo 4 acima. Podemos usar a Técnica de Reexecução de Oráculo para construir um $\mathcal{W}$ tal que, com probabilidade*

$$\Pr[\mathcal{W}] = \left(\frac{1}{q_u q_m} \left(1 - \frac{1}{p}\right)^3 \right)^2 \left(\frac{\epsilon^2}{8 q_{h_1}} \right) \left(1 - \frac{1}{q_{h_1}}\right) \approx \left(\frac{\epsilon^2}{8 q_{h_1} q_u^2 q_m^2} \right).$$

$\mathcal{W}$ consegue gerar duas falsificações $\varsigma_i = (\sigma_i, M_i, ID_i, P_{ID_i}, R_i)$ para $i \in \{1, 2\}$ tais que:

- $ID_1 = ID_2 = ID^*$;
- $R_1 = R_2 = R^*$;
- $H'_1(ID_1, R_1) \neq H''_1(ID_2, R_2)$

Provaremos esta alegação em §5.3.2, mas a utilizamos para mostrar que a probabilidade de sucesso $\Pr[\mathcal{W}]$ é não-desprezível caso $\lambda(k)$ também o seja, encerrando assim a prova do teorema. □

5.3.2 A Técnica de Reexecução de Oráculo

Para terminar a prova do Teorema 7 precisamos provar a Alegação 1. Usaremos aqui a mesma técnica de reexecução de oráculos primeiro discutida em §4.3. Infelizmente o nosso esquema não é um esquema CL-Genérico: de fato, assinaturas nem são aleatorizadas. Por outro lado, como as chaves parciais são basicamente assinaturas Schnorr, existe um certo nível de aleatorização "implícita" que pode ser usada de forma bastante semelhante à da técnica apresentada em §4.3. De fato, a demonstração é bastante semelhante à apresentada em §4.3, sendo apresentada aqui de forma mais sucinta.

Tome uma execução arbitrária do Jogo 4 acima. Seja $\lambda(k)$ a probabilidade de sucesso de $\mathcal{A}_I$. A probabilidade de $\mathcal{A}_I$ ter sucesso mas a consulta $H_1(ID^*, R^*)$ não ter sido feita é $\frac{1}{p}$, devido à aleatoriedade de $H_1(.)$. Seja β o índice da consulta $H_1(ID^*, R^*)$ ($\beta = \infty$ se a consulta não foi feita).

Corolário 1. $\Pr[S = 1 \wedge \beta \neq \infty] \geq \lambda(k) - \frac{1}{p}$

Definiremos um algoritmo $\mathcal{W}$ que se comporta exatamente como $\mathcal{D}_I^4$ até $\mathcal{A}_I$ gerar uma falsificação e depois muda seu comportamento. Seja ω a fita aleatória utilizada na simulação por $\mathcal{A}$. Sejam $\mathcal{Q}_{1,1}, \mathcal{Q}_{1,2}, \ldots, \mathcal{Q}_{1,q_{h_1}}$ as diferentes consultas feitas por $\mathcal{A}_I$ ao oráculo $H_1(.)$ ao longo da simulação. Seja $\rho = (\rho_1, \rho_2, \ldots, \rho_{q_{h_1}})$ a lista das respostas retornadas por $\mathcal{D}_I^4$.

Fato 1. Sabemos que $\mathcal{Q}_{1,\beta} = (ID^*, R^*)$.

Seja ψ o conjunto de execuções de $\mathcal{A}_I$ tais que $S = 1$e $\beta \neq \infty$. Seja ψ_i o subconjunto de ψ tal que $\beta = i$.

Fato 2. $\psi = \bigcup_{\forall i} \psi_i$, e $\Pr[(\omega, H_1(.)) \in \psi] = \epsilon - \frac{1}{p}$, onde a probabilidade é calculada sobre todas as possíveis execuções (ω, H_1).
Seja I o conjunto de índices dos ψ_i mais prováveis, i.e

$$I = \left\{ i | \Pr[(\omega, H_1) \in \psi_i | (\omega, H_1) \in \psi] \geq \frac{1}{2q_{h_1}} \right\} \tag{5.2}$$

Seja então $\psi_I = \{(\omega, H_1) \in \psi_i | i \in I\}$. Logo, $\forall i \in I$,

$$\Pr[(\omega, H_1) \in \psi_i] = \Pr[(\omega, H_1) \in \psi].\Pr[(\omega, H_1) \in \psi_i | (\omega, H_1) \in \psi] \geq \left(\epsilon - \frac{1}{p}\right)\left(\frac{1}{2q_{h_1}}\right)$$

Lema 8. $\Pr[(\omega, H_1) \in \psi_I | (\omega, H_1) \in \psi] \geq \frac{1}{2}$.

Demonstração. Como os conjuntos são disjuntos,

$$\begin{aligned}\Pr[(\omega, H_1) \in \psi_I | (\omega, H_1) \in \psi] &= \sum_{i \in I} \Pr[(\omega, H_1) \in \psi_i | (\omega, H_1) \in \psi] \\ &= 1 - \sum_{i \notin I} \Pr[(\omega, H_1) \in \psi_i | (\omega, H_1) \in \psi]\end{aligned}$$

Lembrando a definição de I (eq. (5.2)),

$$\forall i \notin I, \Pr[(\omega, H_1) \in \psi_i | (\omega, H_1) \in \psi] < \frac{1}{2q_{h_1}}$$

e como existem no máximo q_{h_1} índices i, temos que

$$\begin{aligned}\Pr[(\omega, H_1) \in \psi_I | (\omega, H_1) \in \psi] &= 1 - \sum_{i \notin I} \Pr[(\omega, H_1) \in \psi_i | (\omega, H_1) \in \psi] \\ &\geq 1 - q_{h_1} \left(\frac{1}{2q_{h_1}} \right) \\ &\geq \frac{1}{2}\end{aligned}$$

□

O lema acima nos diz que, dada uma execução de $\mathcal{A}_I$ tal que $S = 1$ (i.e., $\mathcal{A}_I$ consegue gerar uma falsificação válida), a probabilidade desta execução estar no conjunto "mais provável" I é de pelo menos $\frac{1}{2}$. Usaremos a seguir, novamente, o Lema da Separação; voltamos a enunciá-lo aqui, para facilidade de referência. Para sua demonstração, referimos o leitor a §4.3, Lema 3.

Lema 9. (Lema da Separação). *Sejam X e Y dois conjuntos finitos onde duas distribuições de probabilidade estão sendo consideradas. Seja $A \subset X \times Y$ um conjunto tal que $\Pr[A] \geq \gamma$, onde a distribuição de probabilidade em $X \times Y$ é a probabilidade conjunta induzida pelas distribuições de X e Y. Para qualquer $\alpha < \gamma$, definimos*

$$B = \{(x, y) \in X \times Y | Pr_{y' \in Y}[(x, y') \in A] \geq \gamma - \alpha\}$$

e $\overline{B} = X \times Y - B$; então as afirmações seguintes são verdadeiras:

1. $\Pr[B] \geq \alpha$;
2. *para qualquer $(x, y) \in B, Pr_{y' \in Y}[(x, y') \in A] \geq \gamma - \alpha$;*
3. $\Pr[B|A] \geq \alpha/\gamma$.

Utilizamos este lema para "dividir" a execução em duas: as consultas feitas antes e depois de $\mathcal{Q}_{1,\beta}$. Fazendo isto, podemos utilizar o Lema da Separação para provar que existe uma quantidade suficiente de execuções de $\mathcal{A}_I$ que têm sucesso e que começam com a mesma seqüência de consultas a H_1 (até a consulta imediatamente anterior a $Q_{1,\beta}$). Especificamente, usamos o Lema da Separação com os seguinte valores:

$$\begin{aligned} X &= (\omega, H_{1\beta^-}) \\ Y &= H_{1\beta^+} \\ \gamma &= \tilde{\epsilon}/2q_{h_1} \\ \alpha &= \tilde{\epsilon}/4q_{h_1} = \gamma/2 \end{aligned}$$

onde:

- H_{1i^-} denota as consultas de hash $\{H_{1,1}, H_{1,2}, \ldots, H_{1,i}\}$;
- H_{1i^+} denota as consultas de hash $\{H_{1,i+1}, H_{1,i+2}, \ldots, H_{1,q_{h_1}}\}$;
- $\tilde{\epsilon} = \Pr[S = 1 \wedge \beta \neq \infty] = \epsilon\left(1 - \frac{1}{p}\right)$.

Logo, se fizermos $A = \psi_\beta$, existe um $\Omega_\beta \subset \psi_\beta$ (o B no Lema da Separação) tal que:

$$\Pr[(\omega, H_1) \in \Omega_\beta | (\omega, H_1) \in \psi_\beta] = \frac{\alpha}{\gamma} = \frac{1}{2},$$

e, $\forall (\omega, H_1) \in \Omega_\beta$,

$$\Pr[(\omega, H_{1\beta^-} || H_{1\beta^+}) \in \psi_\beta] = \gamma - \alpha = \frac{\tilde{\epsilon}}{4q_{h_1}}.$$

Portanto, seja $\mathcal{W}$ o algoritmo que se comporta exatamente como $\mathcal{D}_I^4$ até obter um primeiro sucesso, e depois repete a simulação com um $(\omega, H_{1\beta^-})$ fixo e variando aleatoriamente $\tilde{H_{1\beta^+}}$. Sabemos que

$$\Pr\left[\left((\omega, H_{1\beta^-} || \tilde{H_{1\beta^+}}) \in \psi_\beta\right) \wedge (\tilde{\rho_\beta} \neq \rho_\beta)\right] = \frac{\tilde{\epsilon}}{4q_{h_1}}\left(1 - \frac{1}{q_{h_1}}\right).$$

Seja $\langle S, (V_1, M_1, ID_1, P_{ID_1}, R_1), (V_2, M_2, ID_2, P_{ID_2}, R_2)\rangle$ a saída de $\mathcal{W}$, onde $\varsigma_i = (V_i, M_i, ID_i, P_{ID_i}, R_i)$ é a falsificação obtida na i-ésima execução do ataque, e $S = 1$ se e somente se $(ID_1 = ID_2 \wedge R_1 = R_2)$, e ambas as falsificações são válidas. Para $\mathcal{W}$ ter sucesso e conseguir resolver o CDHP, quatro coisas precisam acontecer:

1. A primeira execução do ataque tem que pertencer ao conjunto de execuções mais prováveis, I. Denotemos este evento por W_1.

$$\Pr[W_1] = \Pr[(\omega, H_1) \in \psi_I] = \frac{1}{2}\left(\epsilon - \frac{1}{p}\right) = \frac{\tilde{\epsilon}}{2}.$$

2. A execução tem que estar no conjunto B do Lema da Separação. Denotemos este evento por W_2.

$$\Pr[W_2] = \Pr[(\omega, H_1) \in \Omega_\beta | (\omega, H_1) \in \psi_\beta] = \frac{1}{2}.$$

3. A reexecução do oráculo tem que ser bem sucedida. Denotemos este evento por W_3.

$$\Pr[W_3] = \Pr[((\omega, H_{1\beta^-} || \tilde{H_{1\beta^+}}) \in \psi_\beta) \wedge (\tilde{\rho_\beta} \neq \rho_\beta)] = \left(\frac{\tilde{\epsilon}}{4q_{h_1}}\right)\left(1 - \frac{1}{q_{h_1}}\right).$$

4. Finalmente, as condições de sucesso de $\mathcal{D}_I^4$ têm que acontecer em ambas as execuções do ataque. Denotemos este evento por W_4:

$$\Pr[W_4 | W_1 \wedge W_2 \wedge W_3] = \left(\frac{1}{q_u q_m}\left(1 - \frac{1}{p}\right)^3\right)^2.$$

Isto nos dá a probabilidade de sucesso de $\mathcal{W}$, concluindo a prova da Alegação 1

$$\Pr[\mathcal{W}] = \Pr[W_1]\Pr[W_2]\Pr[W_3]\Pr[W_4] = \left(\frac{\tilde{\epsilon}^2}{8q_{h_1}}\right)\left(1 - \frac{1}{q_{h_1}}\right)\left(\left(\frac{1}{q_u q_m}\right)\left(1 - \frac{1}{p}\right)^3\right)^2.$$

□

5.3.3 Adversários Tipo II

A prova de segurança contra adversários Tipo-II tem uma estrutura semelhante à anterior.

Teorema 8. *Se existe um adversário de Tipo II $\mathcal{A}_{II}$ capaz de quebrar a segurança EU-CMA do nosso esquema com probabilidade não-desprezível $\lambda(k)$ então existe um algoritmo $\mathcal{D}_{II}^5$ que resolve o CDHP no grupo $\mathbb{G}$ com probabilidade não-desprezível*

$$\Pr[\mathcal{D}_{II}^5] = \left(\frac{\lambda(k)}{q_{ID} q_m}\right)\left(1 - \frac{1}{p}\right)^2.$$

Demonstração. Novamente, o adversário $\mathcal{A}_{II}$ terá acesso aos seguintes oráculos:

- $\mathbf{H_1}(ID, R), \mathbf{H_2}(M_I D, P_{ID}, R), \mathbf{H_3}(M, ID, P_{ID}, R)$,
- **RevelaChavePública**(ID),
- **RevelaChaveParcial**(ID),
- **RevelaValorSecreto**(ID, R', P'_{ID}),
- **SubstituiChavePública**(ID),
- **Assina**(M, ID).

Novamente contruiremos nossa demonstração como uma seqüência de jogos, onde o primeiro jogo a seguir é uma tradução direta da noção de segurança contra adversários Tipo-II, e o último jogo da série descreve um desafiante $\mathcal{D}_{II}^5$ capaz de usar $\mathcal{A}_{II}$ para resolver o CDHP.

Jogo 0. $\mathcal{D}_{II}^0$ recebe uma instância do CDHP: a descrição do grupo $\mathbb{G}$ e a tupla $\langle P, aP, bP\rangle \in \mathbb{G}^3$.
$\mathcal{D}_{II}^0$ escolhe então $(\mathbb{G}_T, e)$ tais que:

- $|\mathbb{G}| = |\mathbb{G}_T|$;
- $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ é um emparelhamento admissível.

Seja $p = |\mathbb{G}| = |\mathbb{G}_T|$. $\mathcal{D}_{II}^0$ escolhe $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $P_{pub} = sP$.
$\mathcal{D}_{II}^0$ executa $\mathcal{A}_{II}$ com $\langle \mathbb{G}, \mathbb{G}_T, e, P, P_{pub}\rangle$ como entrada e simulando os oráculos como a seguir:

- $\mathbf{H_1}(ID_i, R_i)$.
  Se já não foi definido, escolhe $h_{1_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $H_1(ID_i, R_i) = h_{1_i}$;
  retorne $H_1(ID_i, R_i)$.
- $\mathbf{H_2}(M_i, ID_i, P_{ID_i}, R_i)$.
  Se já não foi definido, escolhe $h_{2_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $H_2(M_i, ID_i, P_{ID_i}, R_i) = h_{2_i}P$;
  retorne $H_2(M_i, ID_i, P_{ID_i}, R_i)$.
- $\mathbf{H_3}(M_i, ID_i, P_{ID_i}, R_i)$.
  Se já não foi definido, escolhe $h_{3_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $H_3(M_i, ID_i, P_{ID_i}, R_i) = h_{3_i}P$;
  retorne $H_3(M_i, ID_i, P_{ID_i}, R_i)$.
- **RevelaValorSecreto**(ID_i).
  Se já não foi definido, escolhe $x_{ID_i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $P_{ID_i} = x_{ID_i}P$;
  retorne x_{ID_i}.
- **RevelaChaveParcial**(ID_i).
  Se já não foi definida, escolhe $r_i \overset{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $R_i = r_iP$;
  calcule $d_{ID_i} = r_i + sH_1(ID_i, R_i) \pmod p$; retorne (d_{ID_i}, R_i).
- **RevelaChavePública**(ID_i).
  Se P_{ID_i} ainda não foi definida, invoca **RevelaValorSecreto**$(\mathbf{ID_i})$;
  Se R_i ainda não foi definida, invoca **RevelaChaveParcial**$(\mathbf{ID_i})$;
  retorne (P_{ID_i}, R_i).
- **SubstituiChavePública**(ID_i, P'_{ID_i}, R'_i).
  Faz $P_{ID_i} = P'_{ID_i}$ e $R_i = R'_i$.
- **Assina**(M_i, ID_i).
  Seja $h_{1_i} = H_1(ID_i, R_i)$;
  seja $h_{2_i}P = H_2(M_i, ID_i, P_{ID_i}, R_i)$;

seja $h_{3_i}P = H_3(M_i, ID_i, P_{ID_i}, R_i)$;
retorne $\sigma_i = h_{2_i}R_i + h_{2_i}h_{1_i}P_{pub} + h_{3_i}P_{ID_i}$.

A simulação deste Jogo 0 é exatamente igual à do Jogo 0 da demonstração de segurança contra adversários Tipo-I. Assim, mantemos a propriedade de que o oráculo de assinaturas é um oráculo *forte*. Como mostrado abaixo, as assinaturas são corretamente simuladas.

$$\begin{aligned} e(P, \sigma_i) &= e(P, h_{2_i}R + h_{2_i}h_{1_i}P_{pub} + h_{3_i}P_{ID_i}) \\ &= e(P, h_{2_i}(R + h_{1_i}P_{pub}))e(P, h_{3_i}P_{ID_i}) \\ &= e(h_{2_i}P, R + h_{1_i}P_{pub}))e(h_{3_i}P, P_{ID_i}) \\ &= e(H_2(M_i, ID_i, P_{ID_i}, R_i), R + H_1(ID_i, R_i)P_{pub}))e(H_3(M_i, ID_i, P_{ID_i}, R_i), P_{ID_i}). \end{aligned}$$

Após a simulação do ataque, $\mathcal{A}_{II}$ deve retornar a tupla $\langle S, (\sigma^*, M^*, ID^*, P_{ID_i}, R^*)\rangle$. Neste jogo inicial, $\mathcal{D}^0_{II}$ tem sucesso exatamente quando $\mathcal{A}_{II}$ retorna $S = 1$, portanto

$$\Pr[\mathcal{D}^0_{II}] = \Pr[\mathcal{A}_{II}] = \Pr[S = 1] = \lambda(k).$$

Jogo 1. O Jogo 1 é semelhante ao Jogo 0, exceto pelo fato de que $\mathcal{D}^1_{II}$ escolherá uma *identidade-alvo*, e só terá sucesso caso a falsificação gerada por $\mathcal{A}_{II}$ seja sob essa identidade. Seja então q_{ID} o número total de identidades consultadas durante toda a simulação; antes do início do jogo, $\mathcal{D}^1_{II}$ escolhe $t \overset{R}{\leftarrow} \{1, \ldots, q_{ID}\}$. Podemos criar então artificialmente o oráculo **CriaUsuário**(ID_i) que é chamado por todos os outros oráculos e controla quantos usuários já foram criados até o momento:

- **CriaUsuário**(ID_i).
  Se ID_i ainda não apareceu no jogo:

  $id\_count = id\_count + 1$.

  Se esta é a t-ésima identidade distinta consultada:

  faz $ID^t = ID_i$.

Esta mudança não traz qualquer alteração no andamento do jogo, sendo assim imperceptível ao adversário. Redefinimos então a saída de $\mathcal{D}^1_{II}$ para:

$$\text{saída}_{\mathcal{D}^1_{II}} = \begin{cases} \langle 0, \perp\rangle, & \text{se } (S = 1) \wedge (ID^* \neq ID_t) \\ \text{saída}_{\mathcal{A}_{II}}, & \text{c.c.} \end{cases}$$

Como estamos lidando com oráculos aleatórios, a probabilidade de $\mathcal{A}_{II}$ gerar uma falsificação válida sem consultar a identidade ID^* aos oráculos é baixíssima, $(1 - \frac{1}{p})$. Por outro lado, a probabilidade de $\mathcal{D}^1_{II}$ adivinhar corretamente a identidade-alvo é $\frac{1}{q_{ID}}$. Temos então:

$$\Pr[\mathcal{D}^1_{II}] = \lambda(k)\left(\frac{1}{q_{ID}}\right)\left(1 - \frac{1}{p}\right).$$

Ou seja, se $\lambda(k)$ for não-desprezível, a probabilidade de sucesso de $\mathcal{D}_{II}^1$ também o será.

Jogo 2. Alteramos então o jogo para que dois tipos de consultas, **RevelaValorSecreto** e **SubstituiChavePública**, não possam ser executadas em ID^t. Ficamos então com os oráculos:

- **RevelaValorSecreto**(ID_i).
  Invoca **CriaUsuário**(ID_i)
  Se $ID_i = ID^t$, `FALHA`.
  Se já não foi definido, escolhe $x_{ID_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $P_{ID_i} = x_{ID_i} P$;
  retorne x_{ID_i}.
- **SubstituiChavePública**(ID_i, P'_{ID_i}, R'_i).
  Invoca **CriaUsuário**(ID_i)
  Se $ID_i = ID^t$, `FALHA`.
  Faz $P_{ID_i} = P'_{ID_i}$ e $R_i = R'_i$.

O interessante desta alteração é que, além de ser imperceptível para $\mathcal{A}_{II}$, ela não altera a probabilidade de sucesso de $\mathcal{D}_{II}^1$ pois, por definição, sempre que $\mathcal{D}_{II}^1$ tinha sucesso a falsificação gerada por $\mathcal{A}_{II}$ era sob a identidade ID^t; sendo assim, ele não poderia ter consultado os oráculos acima nesta identidade (por definição do tipo de ataque!). Logo, temos que:

$$\Pr[\mathcal{D}_{II}^2] = \Pr[\mathcal{D}_{II}^1] = \lambda(k) \left(\frac{1}{q_{ID}}\right) \left(1 - \frac{1}{p}\right).$$

Jogo 3. No Jogo 3, $\mathcal{D}_{II}^3$ passa a utilizar a segunda entrada para o CDHP como a chave pública da identidade-alvo ID^t. Alteramos então o oráculo para:

- **RevelaChavePública**(ID_i).
  Invoca **CriaUsuário**(ID_i);
  se $ID_i = ID^t$, faz $P_{ID_i} = aP$ e $x_{ID_i} = \perp$;
  se P_{ID_i} ainda não foi definida, invoca **RevelaValorSecreto**($\mathbf{ID_i}$);
  se R_i ainda não foi definida, invoca **RevelaChaveParcial**($\mathbf{ID_i}$);
  retorne (P_{ID_i}, R_i).

Isto não altera a distribuição das respostas do oráculo, sendo portanto imperceptível para $\mathcal{A}_{II}$. Além disso, a probabilidade de sucesso de $\Pr[\mathcal{D}_{II}^3]$ permanece igual à de $\Pr[\mathcal{D}_{II}^2]$:

$$\Pr[\mathcal{D}_{II}^3] = \Pr[\mathcal{D}_{II}^2] = \lambda(k) \left(\frac{1}{q_{ID}}\right) \left(1 - \frac{1}{p}\right).$$

Jogo 4. No Jogo 4, $\mathcal{D}_{II}^4$ escolhe uma tupla $(M^u, ID^u, P_{ID^u}, R^u)$ alvo. Seja q_m um limite superior para o número de $H_3(M_i, ID_i, P_{ID_i}, R_i)$ consultadas ao longo da execução do jogo. Um limite superior conservador para q_m é $q_m \leq q_{h_3} + q_s$. Antes do início do jogo, $\mathcal{D}_{II}^4$ escolhe então $u \stackrel{R}{\leftarrow} \{1, \ldots, q_m\}$. O oráculo $H_3(.)$ passa então a ser:

- $\mathbf{H_3}(M_i, ID_i, P_{ID_i}, R_i)$.
  Invoca **CriaUsuário**(ID_i); se esta é a u-ésima consulta distinta:

  faz $M^u = M_i, ID^u = ID_i, P_{ID^u} = P_{ID_i}, R^u = R_i$

  Se já não foi definido, escolhe $h_{3_i} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ e faz $H_3(M_i, ID_i, P_{ID_i}, R_i) = h_{3_i}P$; retorne $H_3(M_i, ID_i, P_{ID_i}, R_i)$.

Novamente esta mudança é imperceptível para o adversário e seu comportamento não deve mudar. Redefinimos então a saída de $\mathcal{D}_{II}^4$ para:

$$\text{saída}_{\mathcal{D}_{II}^4} = \begin{cases} \langle 0, \perp \rangle, & \text{se } S = 1 \wedge ID^* \neq ID^t \\ \langle 0, \perp \rangle, & \text{se } S = 1 \wedge (M^* \neq M^u \vee ID^* \neq ID^u \vee P_{ID^*} \neq P_{ID^u} \vee R^* \neq R^u) \\ \text{saída}_{\mathcal{A}_{II}}, & \text{c.c.} \end{cases}$$

A probabilidade de sucesso de $\mathcal{D}_{II}^4$ será ainda menor que a de $\mathcal{D}_{II}^3$ pois a tupla $(M^u, ID^u, P_{ID^u}, R^u)$ tem que ser adivinhada corretamente. Porém como pode ser observado abaixo, ela ainda é não-desprezível se $\lambda(k)$ o for. Seja S_2^* o evento em que $\mathcal{D}_{II}^4$ acerta a identidade-alvo, e S_3^* o evento em que $\mathcal{D}_{II}$ acerta a tupla-alvo. Temos então:

$$\begin{aligned} &Pr[S_2^*] = 1/q_{ID}(1 - \tfrac{1}{p}); \\ &Pr[S_3^*] = 1/q_m(1 - \tfrac{1}{p}); \\ &Pr[\mathcal{D}_{II}^4] = Pr[\mathcal{D}_{II}^3 \wedge S_3^*] = Pr[\mathcal{D}_{II}^0 \wedge S_2^* \wedge S_3^*] = \tfrac{\lambda(k)}{q_{ID}q_m}(1 - \tfrac{1}{p})^2. \end{aligned}$$

Jogo 5. Finalmente, no Jogo 5, a última entrada do CDHP é utilizada e fazemos $H_3(M^u, ID^u, P_{ID^u}, R^u) \leftarrow bP$. Novamente, esta mudança é imperceptível para $\mathcal{A}_{II}$ e as probabilidades de sucesso se mantêm:

$$Pr[\mathcal{D}_{II}^5] = Pr[\mathcal{D}_{II}^4] = \frac{\lambda(k)}{q_{ID}q_m}(1 - \frac{1}{p})^2.$$

Porém a falsificação gerada no Jogo 5 nos permite resolver o CDHP. Sabemos que

$$e(P, \sigma^*) = e(H_2(M^*, ID^*, P_{ID^*}, R_i), R^* + H_1(ID^*, R^*)P_{pub})e(H_3(M^*, ID^*, P_{ID^*}, R^*), P_{ID^*}),$$

onde

- $P_{ID^*} = aP$;
- $H_3(M^*, ID^*, P_{ID^*}, R^*) = bP$

É fácil portanto calcular $W = \sigma^* - h_{2^*}R^* - h_{2^*}H_1(ID^*, R^*)P_{pub}$ que nos dá:

$$\begin{aligned} e(P, W) &= e(H_3(M^*, ID^*, P_{ID^*}, R^*), P_{ID^*}) \\ &= e(bP, aP). \end{aligned}$$

Ou seja, W é a resposta para a instância do CDHP. □

5.4 Segurança da agregação de assinaturas

Para provar a segurança da agregação de assinaturas no nosso esquema, utilizaremos a técnica de [BNN07]. Neste artigo os autores provam a segurança da agregação reduzindo a falsificação de uma assinatura para a falsificação de um agregado, i.e. construir um algoritmo $\mathcal{B}$ capaz de usar qualquer adversário $\mathcal{A}$, que falsifica assinaturas agregadas, para falsificar uma única assinatura. Como esta idéia pode ser um pouco confusa, tentaremos dar uma intuição mais forte de como essa demonstração será estruturada antes de apresentarmos a prova em si.

O primeiro fato a se notar é que não diferenciaremos explicitamente o caso de adversários Tipo I e de adversários Tipo II: faremos um tratamento uniforme. A demonstração consiste então na descrição de um algoritmo $\mathcal{B}$ que será um falsificador de assinaturas do nosso esquema. Sendo assim, ele poderá cumprir o papel dos $\mathcal{A}_I$ ou $\mathcal{A}_{II}$ das demonstrações de segurança das seções anteriores: ele recebe acesso aos mesmos oráculos (chamados coletivamente de $\mathcal{O}_{\mathcal{B}}$) e tem as mesmas restrições de comportamento. A grande diferença deste $\mathcal{B}$ é que ele também fará o papel de desafiador para um $\mathcal{A}_{\text{agg}}$, um adversário contra a versão agregada do esquema; $\mathcal{A}_{\text{agg}}$ pode ser de Tipo I ou Tipo II e segue as definições de §4.2.2. Sendo assim, $\mathcal{B}$ terá que simular os oráculos definidos em §4.2.2 (e chamados coletivamente aqui de $\mathcal{O}_{\mathcal{A}_{\text{agg}}}$) para $\mathcal{A}_{\text{agg}}$.

O tipo de ataque de $\mathcal{B}$ é determinado pelo tipo de $\mathcal{A}_{\text{agg}}$: descreveremos um $\mathcal{B}$ genérico que se comportará como um adversário Tipo I (Tipo II) quando for um adversário agregado de Tipo I (resp. Tipo II). Uma execução genérica de $\mathcal{B}$ se parece com:

1. $\mathcal{B}$ é inicializado e recebe como entrada os parâmetros de sistema ($mpk_{\mathcal{B}}$) e (possivelmente) uma entrada auxiliar *aux*;
2. $\mathcal{B}$ gera os parâmetros do sistema ($mpk_{\mathcal{A}_{\text{agg}}}$), e (possivelmente) informação auxiliar $aux_{\mathcal{A}_{\text{agg}}}$, para $\mathcal{A}_{\text{agg}}$;
3. $\mathcal{B}$ executa $\mathcal{A}_{\text{agg}}$ com (mpk, aux) como entrada;
4. $\mathcal{A}_{\text{agg}}$ executa por um tempo polinomial;
   - $\mathcal{B}$ simula um ambiente de ataque apropriado para $\mathcal{A}_{\text{agg}}$, especificamente simulando acesso aos oráculos $\mathcal{O}_{\mathcal{A}_{\text{agg}}}$;
5. com probabilidade $\lambda(k)$, $\mathcal{A}_{\text{agg}}$ gera uma falsificação $\langle \gamma^*, \mathbb{U}^*, \mathbb{M}^* \rangle$ tal que:
   - `VerificarAgregado`$(mpk_{\mathcal{A}_{\text{agg}}}, \gamma^*, \mathbb{U}^*, \mathbb{M}^*) =$ `ACEITA`;
   - existe pelo menos um par $(u_i, m_i), u_i \in \mathbb{U}^*, m_i \in \mathbb{M}^*$ que não foi consultado por $\mathcal{A}_{\text{agg}}$ ao seu oráculo de assinatura;
   - as restrições específicas do tipo de ataque (I ou II) não foram quebradas.

6. $\mathcal{B}$ usa $\langle \gamma^*, \mathbb{U}^*, \mathbb{M}^* \rangle$ para gerar uma falsificação ς^* de um usuário u^* na mensagem m^* tal que:
   - $\texttt{Verifica}(mpk_{\mathcal{B}}, \gamma^*, u^*, m^*) = \texttt{ACEITA}$;
   - $\mathcal{B}$ não consultou o par (u^*, m^*) ao seu oráculo de assinatura;
   - as regras específicas do tipo de ataque (I ou II) não foram quebradas;
   - a probabilidade de sucesso de $\mathcal{B}$ é não-desprezível se $\lambda(k)$ o for.

Se conseguirmos construir um tal $\mathcal{B}$, a demonstração estará completa.

Teorema 9. *Se existir um adversário agregado de Tipo I (Tipo II) capaz de quebrar a versão agregada do nosso esquema, existe um adversário simples de Tipo I (resp. Tipo II) capaz de quebrar a versão padrão do esquema.*

Demonstração. A definição de $\mathcal{B}$ é, na verdade, bastante simples: as informações públicas que ele passará para $\mathcal{A}_{\text{agg}}$ são exatamente as mesmas que recebeu como entrada (i.e., $mpk_{\mathcal{B}} = mpk_{\mathcal{A}_{\text{agg}}}$), assim como qualquer entrada auxiliar[2]. Todos os oráculos a que $\mathcal{A}_{\text{agg}}$ tem acesso são simulados simplesmente repassando a consulta para o oráculo equivalente ao qual $\mathcal{B}$ tem acesso (i.e., repassando consultas a oráculos de $\mathcal{O}_{\mathcal{A}_{\text{agg}}}$ para o oráculo equivalente de $\mathcal{O}_{\mathcal{B}}$).

Após a execução do ataque, $\mathcal{A}_{\text{agg}}$ gera $\langle \gamma^*, \mathbb{U}^*, \mathbb{M}^* \rangle$ tal que (segundo a eq. (5.1)):

$$e(P, \gamma^*) = \prod_{u_i} \left[e\left(\sum_{M_i \in \mathbb{M}_{u_i}} H_2(.), R_{u_i} + h_{u_i} P_{\text{pub}} \right) e\left(\sum_{M_i \in \mathbb{M}_{u_i}} H_3(.), P_{ID_{u_i}} \right) \right].$$

Seja (u^*, m^*) um par arbitrário que não foi consultado por $\mathcal{A}_{\text{agg}}$ ao oráculo de assinatura; para que o ataque tenha sucesso tal par tem que existir. Dividiremos agora os pares (u_i, m_i) em três classes:

1. Se $(u_i \neq u^*)$ então $i \in C_1$;
2. se $(u_i = u^*) \wedge (m_i \neq m^*)$ então $i \in C_2$;
3. se $(u_i = u^*) \wedge (m_i = m^*)$ então $i \in C_3$.

Basicamente, o que $\mathcal{B}$ precisa fazer então é "remover" cada σ_i correspondente aos $i \in (C_1 \cup C_2)$ do agregado γ^*. Para fazer isso ele pode consultar cada par $(u_i, m_i), i \in C_1 \cup C_2$ ao seu oráculo de assinatura (i.e., em $\mathcal{O}_{\mathcal{B}}$) e duas características do nosso esquema vão garantir que este procedimento funcione:

[2] A utilização desta noção de "entrada auxiliar" visa meramente uniformizar o tratamento, e deve-se entender que $aux = \perp$ em ataques Tipo I e que $aux = msk$ em ataques do Tipo II.

1. Como o oráculo é um oráculo *forte*, mesmo que chaves públicas tenham sido substituídas ao longo do ataque, as respostas dadas pelo oráculo serão corretas.

2. Como as assinaturas do nosso esquema são determinísticas, $(\gamma^* - \sigma_i)$ é uma assinatura agregada válida nas listas $(\mathbb{U} - u_i, \mathbb{M} - m_i)$.

Seja então $\gamma' = \gamma^* - \sum_{i \in (C_1 \cup C_2)} \sigma_i$, temos que γ' é uma assinatura agregada com (possivelmente) várias cópias da mesma assinatura, afinal $i, j \in C_3 \rightarrow (u_i = u_j = u^*) \wedge (m_i = m_j = m^*)$ e retiramos todas as assinaturas referentes a $i \notin C_3$ de γ'. Logo, $\gamma' = k\sigma^*$, onde $k = |C_3|$. Portanto, $\mathcal{B}$ pode simplesmente calcular $\sigma^* = \gamma' k^{-1}$, uma assinatura válida de u^* em m^*.

Perceba que a probabilidade de sucesso de $\mathcal{B}$ é exatamente $\lambda(k)$: sempre que $\mathcal{A}_{\text{agg}}$ gera uma falsificação agregada γ^*, $\mathcal{B}$ pode executar o procedimento acima e obter σ^*. Além disso, qualquer overhead no tempo de execução de $\mathcal{B}$ (em relação a $\mathcal{A}_{\text{agg}}$) certamente é polinomial. Estes fatos concluem a prova do teorema, ou seja, agregar assinaturas no nosso esquema é tão seguro quanto utilizar assinaturas simples. □

5.5 Conclusão

Apresentamos neste capítulo uma proposta de esquema de assinaturas digitais sem certificados (§5.2), discutindo em profundidade a sua segurança (§5.3 e §5.4). Este esquema possui características bastante interessantes, como a possibilidade de agregar assinaturas (e fazê-lo de forma especialmente eficiente quando se tem um usuário assinando várias mensagens), a eficiência e o fato de ser demonstravelmente seguro.

Capítulo 6

Considerações Finais

Este documento apresentou uma visão atualizada da área de Assinaturas de Chave Pública sem Certificados *(CL-PKS)*[1], apresentado uma série de trabalhos realizados pelo autor durante o seu mestrado. A Criptografia de Chave Pública sem Certificados *(CL-PKC)*[2] é um novo paradigma de certificação implícita de chaves públicas proposto por Al-Riyami e Paterson [ARP03] para tentar unir as principais vantagens da criptografia de chave pública tradicional (ou explicitamente certificada) e da criptografia baseada em identidade:

- o nível de confiança não muito elevado demandado pelas autoridades centrais tradicionais;
- a maior simplicidade trazida pela ausência de certificados explícitos em criptografia baseada em identidade.

Consegue-se assim esquemas bastante seguros, mas cuja utilização em larga escala requer um custo administrativo razoavelmente menor. Nesta tese fizemos uma ampla revisão do estado atual dos esquema de CL-PKS, e trouxemos uma série de pequenas contribuições à área:

- estudo da aplicabilidade do Lema da Bifurcação a esquemas de CL-PKC (§ 4.3.3);
- algumas pequenas otimizações a esquemas seguros (§ 4.4.1);
- demonstração de segurança para um esquema cuja segurança ainda era um problema em aberto (§ 4.4.2);
- explicação de falhas em demonstrações de segurança de alguns esquemas (§ 4.4.5, § 4.4.7, § 4.4.4);
- um ataque desconhecido a um esquema anteriormente suposto seguro (§ 4.4.4);

Encerrada esta revisão da literatura disponível, apresentamos o principal resultado desta pesquisa: um esquema de CL-PKS que, além de ser o mais eficiente disponível na literatura que é demonstravelmente seguro, possibilita a agregação eficiente de assinaturas

[1] Do ingês *Certificateless Public-Key Signatures.*
[2] Do ingês *Certificateless Public-Key Cryptography.*

(propriedade não disponível em nenhum dos outros esquemas propostos anteriormente). A apresentação deste novo esquema é acompanhada da sua demonstração de segurança para assinaturas individuais, da proposta de um modelo de segurança apropriado para a agregação de assinaturas em CL-PKC, e da demonstração da segurança da agregação de suas assinaturas neste modelo.

Publicações

Abaixo, relacionamos as publicações com participação do autor e relacionadas ao trabalho descrito nesta tese:

- Mini-curso no SBSeg 2007, ministrado pelo autor em co-autoria com Ricardo Dahab e Augusto Devegili. O material escrito por nós foi publicado como capítulo do livro "Minicursos SBSeg 2007" [CDD07].
- Em co-autoria com Ricardo Dahab, um artigo contendo a maior parte dos resultados apresentados no capítulo 4 sobre segurança de CL-PKS publicado no ProvSec 2007 [CD07b].
- Novamente em co-autoria com Ricardo Dahab, um artigo publicado no eprint descrevendo o esquema de CL-PKS descrito no capítulo 5 [CD07a].
- Em co-autoria com Diego Aranha, Julio Lopez e Ricardo Dahab, um artigo descrevendo um esquema de Cifrassinatura para CL-PKS publicado no SBSeg 2008.

Sugestões de tópicos de pesquisa

A área de Criptografia de Chave Pública sem Certificados é ainda muito nova e, portanto, muito fértil. Muita pesquisa ainda é necessária até o amadurecimento da área. Alguns primeiros caminhos interessantes são claros:

- **Estudo de modelos de segurança para cifrassinaturas em CL-PKC.** Como citamos anteriormente, propusemos um novo esquema de cifrassinaturas em CL-PKC, mas sem demonstrar a sua segurança. De maneira semelhante, existem alguns outros esquemas disponíveis na literatura, igualmente sem demonstrações de segurança. Chegar a um modelo adequado e conseguir a demonstração da segurança de um desses esquemas (ou de um novo), seria um resultado bastante positivo.
- **Estudo da segurança de esquemas mais eficientes do que o proposto aqui.** Como discutimos no capítulo 4, existem propostas de dois esquemas mais eficientes que o apresentado nesta tese, mas a sua demonstração de segurança contém falhas. Chegar a uma conclusão definitiva sobre a sua segurança (seja pela descoberta de um ataque, seja pela proposta de uma nova demonstração), é de extrema importância.
- **Estudo de esquemas seguros no modelo padrão.** Até onde sabemos, existe apenas um esquema de CL-PKS seguro no modelo padrão na literatura. Como a

segurança no modelo padrão é mais forte do que a segurança no modelo do oráculo aleatório, é importante que continuemos a tentar tornar os esquemas de CL-PKS com esta propriedade cada vez mais eficientes.

- **Discussão mais aprofundada do uso de CL-PKC na prática.** CL-PKC é um paradigma razoavelmente novo e, até onde sabemos, ainda não utilizado com sucesso em situações reais. Um trabalho neste sentido seria de suma importância, identificando potenciais problemas no seu uso, estabelecendo padrões para os seus processos e protocolos, assim como o estudo de PKIs fez para a criptografia tradicional de chave pública.
- **Estudo dos modelos de segurança.** O modelo de segurança para CL-PKC tem uma peculiaridade bastante trabalhosa: todas as demonstrações devem levar em consideração dois modelos completamente diferentes de adversários, o que geralmente leva à construção de basicamente duas demonstrações independentes, duplicando o trabalho necessário para conseguir provar a segurança de qualquer esquema. Aparentemente, isto é necessário devido às suposições básicas de CL-PKC. Mas talvez o que falte seja simplesmente uma nova maneira de olhar para a situação, que simplifique o resultado final.

Referências Bibliográficas

[ACL+06] Man Ho Au, Jing Chen, Joseph K. Liu, Yi Mu, Duncan S. Wong, and Guomin Yang. Malicious kgc attacks in certificateless cryptography. Cryptology ePrint Archive, Report 2006/255, 2006. `http://eprint.iacr.org/`.

[ARP03] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless public key cryptography. In Chi-Sung Laih, editor, *ASIACRYPT*, volume 2894 of *Lecture Notes in Computer Science*, pages 452–473. Springer, 2003.

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Cachin and Camenisch [CC04], pages 56–73.

[BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In Cachin and Camenisch [CC04], pages 171–188.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Kilian [Kil01], pages 213–229.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2003.

[BLMQ05] Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In Bimal K. Roy, editor, *ASIACRYPT*, volume 3788 of *Lecture Notes in Computer Science*, pages 515–532. Springer, 2005.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, pages 514–532, 2001.

[BNN07] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Unrestricted aggregate signatures. In Lars Arge, Christian Cachin, Tomasz Jurdzinski, and

Andrzej Tarlecki, editors, *ICALP*, volume 4596 of *Lecture Notes in Computer Science*, pages 411–422. Springer, 2007.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *EUROCRYPT*, pages 92–111, 1994.

[BR04] Mihir Bellare and Phillip Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint Archive, Report 2004/331, 2004. `http://eprint.iacr.org/`.

[BSN06] Lynn Margaret Batten and Reihaneh Safavi-Naini, editors. *Information Security and Privacy, 11th Australasian Conference, ACISP 2006, Melbourne, Australia, July 3-5, 2006, Proceedings*, volume 4058 of *Lecture Notes in Computer Science*. Springer, 2006.

[CC03] Jae Choon Cha and Jung Hee Cheon. An identity-based signature from gap diffie-hellman groups. In Yvo Desmedt, editor, *Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 18–30. Springer, 2003.

[CC04] Christian Cachin and Jan Camenisch, editors. *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*. Springer, 2004.

[CD07a] Rafael Castro and Ricardo Dahab. Efficient certificateless signatures suitable for aggregation. Cryptology ePrint Archive, Report 2007/454, 2007. `http://eprint.iacr.org/`.

[CD07b] Rafael Castro and Ricardo Dahab. Two notes on the security of certificateless signatures. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *The 1st International Conference on Provable Security (ProvSec) 2007*, volume 4784 of *Lecture Notes in Computer Science*. Springer, 2007.

[CDD07] Rafael Castro, Ricardo Dahab, and Augusto Jun Devegili. *Sétimo Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais: Minicursos do SBSeg 2007*, chapter Introdução à Segurança Demonstrável, pages 103–152. UFRJ/NCE, Rio de Janeiro, 2007.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *STOC*, pages 209–218, 1998.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 40–57. Springer, 2004.

[CPHL07] Kyu Young Choi, Jong Hwan Park, Jung Yeon Hwang, and Dong Hoon Lee. Efficient certificateless signature schemes. In Jonathan Katz and Moti Yung, editors, *ACNS*, volume 4521 of *Lecture Notes in Computer Science*, pages 443–458. Springer, 2007.

[CPK06] Xuefei Cao, Kenneth G. Paterson, and Weidong Kou. An attack on a certificateless signature scheme. Cryptology ePrint Archive, Report 2006/367, 2006. `http://eprint.iacr.org/`.

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

[DW07] Hongzhen Du and Qiaoyan Wen. Efficient and provably-secure certificateless short signature scheme from bilinear pairings. Cryptology ePrint Archive, Report 2007/250, 2007. `http://eprint.iacr.org/`.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[Gir91] Marc Girault. Self-certified public keys. In Donald W. Davies, editor, *EUROCRYPT*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497. Springer, 1991.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC*, pages 365–377. ACM, 1982.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[Goy06] Denise H. Goya. Proposta de esquemas de criptografia e de assinatura sob modelo de criptografia de cha pública sem certificado. Master's thesis, USP, 2006.

[GS05] M. Choudary Gorantla and Ashutosh Saxena. An efficient certificateless signature scheme. In Yue Hao, Jiming Liu, Yuping Wang, Yiu ming Cheung, Hujun Yin, Licheng Jiao, Jianfeng Ma, and Yong-Chang Jiao, editors, *CIS (2)*, volume 3802 of *Lecture Notes in Computer Science*, pages 110–116. Springer, 2005.

[Her06] Javier Herranz. Deterministic identity-based signatures for partial aggregation. *The Computer Journal*, 49(3):322–330, 2006.

[HK07] Shai Halevi and Hugo Krawczyk. Security under key-dependent inputs. Cryptology ePrint Archive, Report 2007/315, 2007. `http://eprint.iacr.org/`.

[HP01] Russ Housley and Tim Polk. *Planning for PKI: Best Practices Guide for Deploying Public Key Infrastructure*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[HSMZ05] Xinyi Huang, Willy Susilo, Yi Mu, and Futai Zhang. On the security of certificateless signature schemes from asiacrypt 2003. In Yvo Desmedt, Huaxiong Wang, Yi Mu, and Yongqing Li, editors, *CANS*, volume 3810 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 2005.

[HWZD06] Bessie C. Hu, Duncan S. Wong, Zhenfeng Zhang, and Xiaotie Deng. Key replacement attack against a generic construction of certificateless signature. In Batten and Safavi-Naini [BSN06], pages 235–246.

[Jou00] Antoine Joux. A one round protocol for tripartite diffie-hellman. In Wieb Bosma, editor, *ANTS*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer, 2000.

[Kil01] Joe Kilian, editor. *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*. Springer, 2001.

[LAS06] Joseph K. Liu, Man Ho Au, and Willy Susilo. Self-generated-certificate public key cryptography and certificateless signature / encryption scheme in the standard model. Cryptology ePrint Archive, Report 2006/373, 2006. `http://eprint.iacr.org/`.

[LCS05] X. Li, K. Chen, and L. Sun. Certificateless signature and proxy signature schemes from bilinear pairings. *Lithuanian Mathematical Journal*, 45(1):76–83, 2005.

[Mil85] Victor S. Miller. Use of elliptic curves in cryptography. In Hugh C. Williams, editor, *CRYPTO*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer, 1985.

[MRS88] Silvio Micali, Charles Rackoff, and Bob Sloan. The notion of security for probabilistic cryptosystems. *SIAM J. Comput.*, 17(2):412–426, 1988.

[MVO91] Alfred Menezes, Scott Vanstone, and Tatsuaki Okamoto. 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*, pages 80–89, New York, NY, USA, 1991. ACM.

[Par06] Je Hong Park. An attack on the certificateless signature scheme from euc workshops 2006. Cryptology ePrint Archive, Report 2006/442, 2006. `http://eprint.iacr.org/`.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *EUROCRYPT*, pages 387–398, 1996.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[PS06] Kenneth G. Paterson and Jacob C. N. Schuldt. Efficient identity-based signatures secure in the standard model. In Batten and Safavi-Naini [BSN06], pages 207–222.

[Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, 1989.

[Sco05] Michael Scott. Computing the tate pairing. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 293–304. Springer, 2005.

[Sha49] C.E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28:656–715, 1949.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.

[Sho01] Victor Shoup. OAEP reconsidered. In Kilian [Kil01], pages 239–259.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. `http://eprint.iacr.org/`.

[SK03] Ryuichi SAKAI and Masao KASAHARA. Id based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive, Report 2003/054, 2003. `http://eprint.iacr.org/`.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *FOCS*, pages 80–91. IEEE, 1982.

[YHG06] Wun-She Yap, Swee-Huay Heng, and Bok-Min Goi. An efficient certificateless signature scheme. In Xiaobo Zhou, Oleg Sokolsky, Lu Yan, Eun-Sun Jung, Zili Shao, Yi Mu, Dong Chun Lee, Daeyoung Kim, Young-Sik Jeong, and Cheng-Zhong Xu, editors, *EUC Workshops*, volume 4097 of *Lecture Notes in Computer Science*, pages 322–331. Springer, 2006.

[ZF06] Zhenfeng Zhang and Dengguo Feng. Key replacement attack on a certificateless signature scheme. Cryptology ePrint Archive, Report 2006/453, 2006. `http://eprint.iacr.org/`.

[ZWXF06] Zhenfeng Zhang, Duncan S. Wong, Jing Xu, and Dengguo Feng. Certificateless public-key signature: Security model and efficient construction. In Jianying Zhou, Moti Yung, and Feng Bao, editors, *ACNS*, volume 3989 of *Lecture Notes in Computer Science*, pages 293–308, 2006.