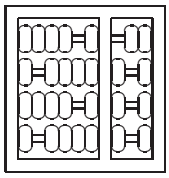

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Criptografia de Chave Pública Sem Certificados

Diego F. Aranha *Julio López*

Technical Report - IC-07-24 - Relatório Técnico

August - 2007 - Agosto

Criptografia de Chave Pública Sem Certificados

Diego F. Aranha* Julio López*

Resumo

Criptografia de Chave Pública sem Certificados (*Certificateless Public Key Cryptography* – CL-PKC) é um paradigma de construção de sistemas criptográficos de chave pública projetado para solucionar as limitações da Criptografia Baseada em Identidades (*Identity-Based Cryptography* – ID-PKC). Neste trabalho, as propriedades, vantagens e implicações desde novo modelo são apresentadas e discutidas, com ênfase em sua instanciação a partir de emparelhamentos bilineares sobre curvas elípticas. Considerações a respeito da segurança do paradigma são formuladas com base no grau de confiança dado à autoridade central, no modelo de adversário adotado e no conjunto de problemas intratáveis mais comuns dos quais depende a segurança dos protocolos. Adicionalmente, esquemas de cifração e assinatura elaborados sob estas premissas são apresentados com detalhes de implementação.

1 Introdução

O advento da criptografia de chave pública [1] revolucionou a forma de se construir sistemas criptográficos e possibilitou a integração de forma definitiva entre teoria criptográfica e implementação em aplicações reais. Particularmente, trouxe a possibilidade de se estabelecer serviços criptográficos como sigilo e assinatura não-repudiável em ambientes em que não existe qualquer relação de confiança entre os envolvidos ou canal seguro para distribuição de chaves. O antigo problema da distribuição de chaves converteu-se na dificuldade de obtenção de uma chave pública ausência de uma entidade. Como solução para este novo problema, um repositório público foi inicialmente proposto como ponto de distribuição de chaves [1], mas é fácil perceber que não há como um repositório deste tipo fornecer autenticidade e que, sem garantias de autenticidade, é possível para um atacante substituir uma chave pública armazenada e posteriormente participar em protocolos personificando entidades legítimas. A autenticação mútua das chaves é crucial para que tais intervenções por parte de agentes não-autorizados possam ser detectadas e evitadas.

Convencionalmente, a verificação da autenticidade de chaves públicas é reduzida à validação de *certificados de titularidade* [2]. O papel de um certificado é mapear uma chave pública a uma entidade, utilizando-se uma assinatura por uma terceira parte confiada, a *autoridade certificadora*, como atestado de integridade. As autoridades certificadoras costumam se organizar em uma estrutura hierárquica para descentralização dos serviços

*Instituto de Computação, Universidade Estadual de Campinas, 13084-971, Campinas - SP. Pesquisa financiada pelo CNPq, processo número 1318202005-2.

comumente disponibilizados como: requisição, emissão, validação e revogação de certificados. A união entre padrões de certificado, autoridades certificadoras e procedimentos para gerência de certificados formam uma *Infra-estrutura de Chaves Públicas* (*Public Key Infrastructure* – PKI) [3]. O funcionamento básico de uma infra-estrutura de chaves públicas está representado na Figura 1.

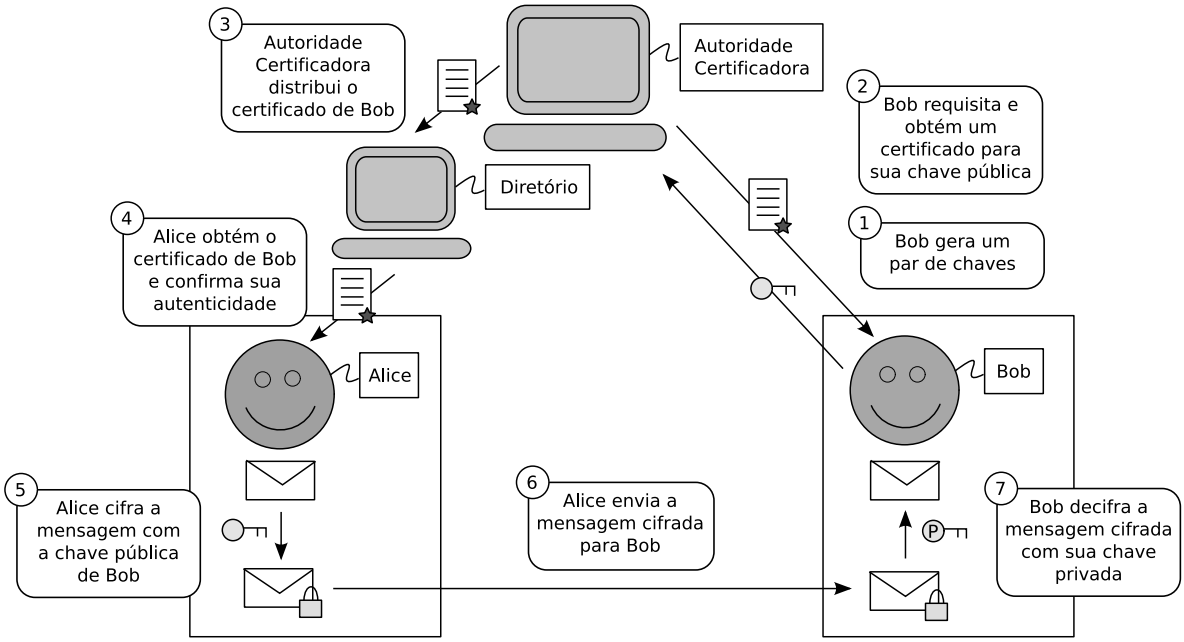

Figura 1: Funcionamento básico de uma PKI tradicional.

Uma infra-estrutura de chaves públicas representa relações que existem entre entidades do mundo real, como filiação a organizações e delegação de serviços de uma organização para outra. Os tipos de relações presentes são diversos, o que traz sérios problemas para o projeto de infra-estruturas que sejam tanto genéricas o suficiente para representar qualquer tipo de relação, quanto simples o suficiente para terem ampla aceitação, possibilidade de padronização e eficiência. Operações de validação de certificados e revogação de chaves públicas tendem a representar situações extremas [3]. A validação exige a determinação de uma cadeia de assinaturas que garante a autenticidade de uma chave pública, bem como a verificação computacionalmente custosa de cada uma destas assinaturas. Técnicas como certificação cruzada criam caminhos múltiplos nas árvores de validação de certificados e dificultam ainda mais a determinação dos conjuntos de assinaturas para verificação, conferindo características exponenciais para um problema que deveria ter complexidade linear. A revogação, por sua vez, traz problemas de distribuição da informação de revogação e demanda agilidade para que o tempo entre a solicitação e a revogação efetiva de um certificado seja mínimo. Isto é claramente importante para que seja minimizado o tempo em que uma chave revogada continue em uso. Pode-se observar que os maiores problemas que afetam o projeto e o emprego de uma infra-estrutura de chaves públicas são relacionados à

escalabilidade.

Para atenuar alguns dos problemas conhecidos de infra-estruturas tradicionais de chaves públicas – o armazenamento, distribuição e verificação de certificados – alternativas que implementam certificação implícita vêm sendo propostas. Destacam-se a Criptografia Baseada em Identidades (*Identity-Based Public Key Cryptography* – ID-PKC) [4] e suas derivadas [5, 6], entre elas a Criptografia de Chave Pública Sem Certificados (*Certificateless Public Key Cryptography* – CL-PKC) [7].

Este documento é organizado como se segue. A seção 2 define em linhas gerais o modelo ID-PKC. A seção 3 define o modelo CL-PKC, formaliza os conceitos apresentados e fornece fundamentação matemática relacionada, incluindo o modelo de adversário considerado. A seção 4 apresenta sistemas criptográficos que enquadram-se no modelo CL-PKC e, por fim, detalhes da implementação realizada são apresentados na seção 5 e as conclusões pertinentes são apresentadas na seção 6.

2 ID-PKC

Sistemas baseados em identidades foram concebidos por Shamir [4] em 1984, mas suas primeiras realizações funcionais para cifração só foram apresentadas em 2001, por Cocks [8] utilizando resíduos quadráticos; e por Boneh e Franklin[9] a partir de emparelhamentos bilineares sobre curvas elípticas. A motivação original para sistemas baseados em identidade era aproveitar a autenticidade de informação publicamente conhecida para simplificar a autenticação de chaves públicas. O objetivo era desenvolver primitivas criptográficas que pudessem utilizar identificadores arbitrários como chave pública, quando a verificação da autenticidade da ligação entre o identificador e a entidade correspondente é trivial.

Uma aplicação imediata para primitivas baseadas em identidade são sistemas de correio eletrônico, como apresentado na Figura 2. Quando uma entidade A deseja enviar uma mensagem para uma entidade B que possui o endereço *bob@mail.com*, basta que a entidade A cifre sua mensagem utilizando a cadeia de caracteres *bob@mail.com* como chave. Não há necessidade da entidade A obter ou verificar a autenticidade da chave pública de B. Ao receber a mensagem cifrada, a entidade B recorre a uma terceira parte confiada chamada *Gerador de Chaves Privadas* (*Private Key Generator* – PKG). A entidade B autentica-se com o PKG e obtém sua chave privada, podendo então decifrar a mensagem recebida. Note que, ao contrário da infra-estrutura atual, a entidade A pode enviar mensagens cifradas para a entidade B mesmo que B ainda não possua um certificado de chave pública. A certificação é implícita: a entidade B só poderá decifrar a mensagem caso possua uma chave privada correta emitida pelo PKG (com o efeito de certificá-la).

Generalizando as técnicas apresentadas para o sistema de correio eletrônico, define-se um modelo ID-PKC a partir de seis algoritmos [9]:

Inicializar. Recebe um parâmetro de segurança k e retorna os parâmetros de sistema params e a chave mestre s. Os parâmetros do sistema incluem uma descrições do espaço de mensagens $\mathcal{M}$, do espaço de criptogramas $\mathcal{C}$ e do espaço de assinaturas $\mathcal{S}$. Seguindo a prática comum, os parâmetros de sistema são conhecidos publicamente, enquanto o segredo s é mantido em sigilo pelo PKG.

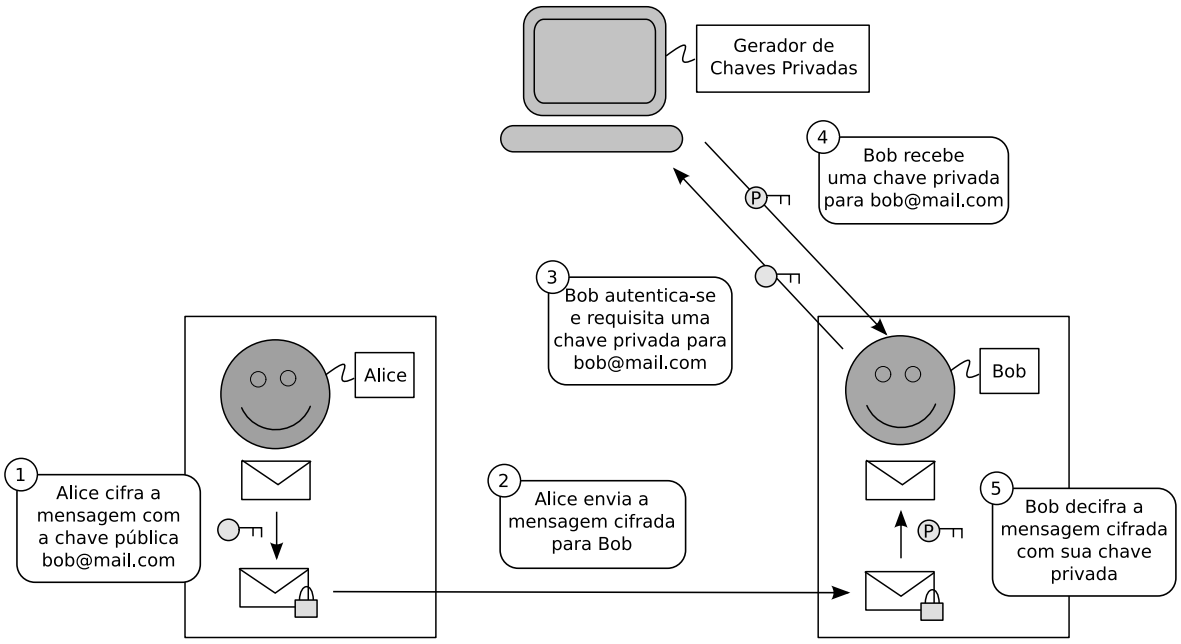

Figura 2: Sistema de Criptografia Baseada em Identidades.

Extrair. Recebe como entrada params, s e um identificador arbitrário $\mathsf{ID} \in \{0,1\}^*$, e retorna uma chave privada d. Como descrito anteriormente, o identificador ID é utilizado como chave pública e d é a chave privada correspondente. Resumidamente, o algoritmo extrai a chave privada de uma chave pública.

Cifrar. recebe como entrada params, ID, $M \in \mathcal{M}$ e retorna um criptograma $C \in \mathcal{C}$.

Decifrar. recebe como entrada params, $C \in \mathcal{C}$, uma chave privada d e retorna $M \in \mathcal{M}$.

Assinar. recebe como entrada params, uma mensagem $M \in \mathcal{M}$, a chave privada d e produz uma assinatura $S \in \mathcal{S}$ para M.

Verificar. recebe como entrada params, a assinatura $S \in \mathcal{S}$ da mensagem $M \in \mathcal{M}$ para uma identidade ID e retorna $\mathsf{verdadeiro}$ ou falso, dependendo se a assinatura é aceitável ou não.

Os algoritmos devem satisfazer condições básicas de consistência:

$$\forall M \in \mathcal{M} : \mathsf{Decifrar}(\mathsf{params}, C = \mathsf{Cifrar}(\mathsf{params}, \mathsf{ID}, M), d) = M$$

$$\forall M \in \mathcal{M} : \mathsf{Verificar}(\mathsf{params}, S = \mathsf{Assinar}(\mathsf{params}, d, M), \mathsf{ID}, M) = \mathsf{verdadeiro}$$

A possibilidade de se produzir chaves públicas a partir de aspectos de identidade elimina a necessidade de certificados e resolve alguns dos problemas associados, mas duas

conseqüências negativas são inseridas: a dependência na geração de chaves privadas introduz custódia inerente em sistemas baseados em identidade (o PKG conhece a chave privada das entidades participantes); e o transporte de chaves privadas entre PKG e clientes exige um canal seguro de comunicação, o que dificulta a distribuição de chaves. É importante observar também que ainda há a necessidade de verificação da autenticidade dos parâmetros de sistema, para verificar que os parâmetros do PKG legítimo estejam sendo utilizados. Vantagens que compensam em parte estas limitações aparecem quando consideram-se a simplicidade trazida para o controle de expiração de chaves (a data de validade da chave pode ser concatenada à chave pública) e para a delegação de privilégios (o papel que cada par de chaves exerce pode também ser codificado diretamente no identificador fornecido ao PKG). Entretanto, a revogação de chaves é difícil, visto que revogar a identidade do detentor do par de chaves pode trazer complicações.

Um volume considerável de pesquisa tem sido realizado no que tange à derivação de primitivas e protocolos que façam uso dessas características. Parte deste esforço está focado em produzir sistemas criptográficos semelhantes aos baseados em identidade, mas com a eliminação da necessidade de custódia das chaves privadas e minimização dos perigos relacionados – comprometimento da chave mestre, ausência de irretratabilidade e efeitos-colaterais da atuação de um PKG malicioso.

3 CL-PKC

As limitações impostas pelo modelo ID-PKC restringem sua aplicabilidade a grupos fechados, com relações de confiança bem-definidas. Assim, a Criptografia de Chave Pública Sem Certificados, proposta por Paterson e Riyami [7], surge como um novo paradigma para criptografia de chave pública que não requer o uso de certificados e não possui a custódia de chave inerente a ID-PKC. Dadas estas características, pode-se afirmar que o modelo situa-se entre a PKI convencional e ID-PKC.

O paradigma CL-PKC mantém a terceira parte confiada presente em ID-PKC, mas altera radicalmente seu papel. A terceira parte agora é denominada *Centro de Geração de Chaves* (*Key Generation Center* – KGC). Ao invés de gerar a chave privada a partir da identidade ID_A de uma entidade A requisitante, o KGC fornece apenas uma chave privada parcial D_A. Esta modificação possibilita que a entidade A combine a chave privada parcial D_A com um segredo d_A de seu conhecimento exclusivo, produzindo a chave privada completa S_A. Desta forma, a chave privada S_A não é conhecida pelo KGC. Intuitivamente, para que as chaves privadas e públicas tenham funcionalidades complementares, é necessário que a chave pública P_A de A seja produzida pela combinação entre o segredo d_A e os parâmetros de sistema distribuídos pelo KGC. O procedimento de geração de chaves privadas parciais, a princípio, continua exigindo confidencialidade e autenticação, já que o KGC deve ter garantias de que a chave privada parcial foi gerada e transmitida para a entidade correta. Observa-se que o sistema deixa de ser baseado em identidades, já que a chave pública não pode ser produzida utilizando-se apenas informação a respeito da identidade de A. A propriedade de que a chave pública pode ser gerada antes da chave privada é mantida, possibilitando efetivamente que a chave pública armazene informação para certificação implícita.

A chave pública da entidade A pode então ser publicada em um repositório, não existindo mais a necessidade de se vincular a chave pública à entidade A por meio de um certificado – a chave pública isolada permite a verificação de titularidade. As alterações em relação ao paradigma ID-PKC podem ser verificadas na Figura 3.

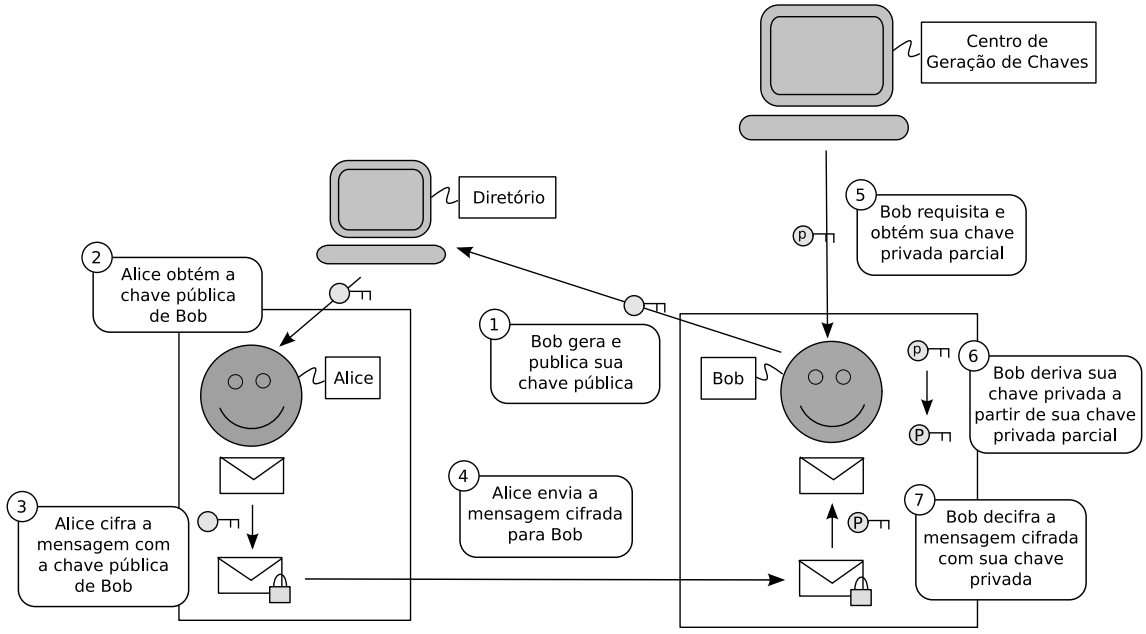

Figura 3: Sistema de Criptografia de Chave Pública Sem Certificados.

3.1 Propriedades

São várias as propriedades do paradigma CL-PKC.

3.1.1 Revogação

Assim como em ID-PKC, a identidade de A é mantida como um identificador ID_A arbitrário, podendo carregar datas de validade ou informação para delegação de privilégios. Instâncias desta técnica podem construir as chaves com curto tempo de vida ou mensagens cifradas para serem lidas no futuro. A revogação de chaves expiradas é automática e eficiente [9].

O problema de revogação imediata, quando ocorre comprometimento da chave privada, por exemplo, não é tão simples. A informação de revogação ainda precisa ser distribuída para os usuários do sistema e pode ser realizada por meios tradicionais (listas de revogação ou protocolos de verificação em tempo real). O requisito de agilidade na distribuição de informação de revogação ainda é mantido. A melhor solução encontrada para este problema utiliza revogação de identificadores produzidos a partir do identificador original e mantém os mesmos custos de comunicação presentes na PKI tradicional [10]. A construção de um

sistema baseado em identidades ou sem certificados que suporte revogação imediata mais eficiente que a PKI tradicional ainda é um problema em aberto e de difícil tratamento [11].

Estas observações aplicam-se também à revogação dos parâmetros da autoridade central, ao se forçar a reinicialização de todos os pares de chaves e a revogação efetiva das chaves antigas [12].

3.1.2 Ausência de certificados

O modelo elimina a necessidade de implementar certificados de titularidade. A chave privada parcial cumpre o papel de certificado, mas altera os mecanismos de distribuição e de informações de certificação. Efeitos colaterais são a simplificação da gerência de chaves e a diminuição dos requisitos de banda, já que é desnecessário contatar a autoridade central para obter certificados adicionais que permitam determinar a validade de uma chave pública. A privacidade das chaves também é elevada, já que os identificadores utilizados carregam apenas a exata informação para funcionamento do sistema, enquanto que, dependendo da aplicação, certificados comuns costumam carregar informação adicional (conta em banco, cadastro de pessoa física, entre outros).

3.1.3 Flexibilidade

A possibilidade de se gerar uma chave pública antes da chave privada correspondente permite que uma entidade B cifre uma mensagem para A utilizando um identificador específico. O identificador deve conter a identidade de A, mas pode conter também informação ligada a uma condição que A precisa demonstrar para o KGC antes que sua chave privada parcial seja emitida. Adicionalmente, o modelo CL-PKC coexiste com ID-PKC: uma entidade A pode solicitar a geração de uma chave privada para Criptografia Baseada em Identidades e imediatamente convertê-la em uma chave adequada para criptografia sem certificados (a mesma infra-estrutura pode ser utilizada para dar suporte aos dois cenários). Vários trabalhos [13, 14] já propõem e analisam a segurança de construções genéricas de CL-PKC utilizando sistemas baseados em identidades, para as primitivas de cifração e assinatura.

3.1.4 Confiança e irretratabilidade

Em infra-estruturas de chave pública, o nível de confiança na terceira parte confiada pode assumir diferentes patamares. Uma formulação simples para confiança na autoridade central é proposta por Girault [5]:

- **Nível de confiança 1:** a autoridade confiada conhece ou pode calcular as chaves privadas das entidades registradas. Assim, pode personificar qualquer das entidades a qualquer hora sem ser detectada;
- **Nível de confiança 2:** a autoridade confiada não conhece e não pode calcular as chaves privadas das entidades, mas ainda pode personificar uma entidade sem ser detectada, pela geração de informação falsa de autenticação; ou

- **Nível de confiança 3:** a autoridade confiada não conhece e não pode calcular as chaves privadas das entidades. A autoridade pode personificar uma entidade, mas a fraude sempre pode ser detectada.

O modelo de PKI alcança nível de confiança 3: a geração desonesta de pares de chaves para uma entidade específica pode ser detectada pela existência de múltiplos certificados válidos para uma mesma chave. ID-PKC e demais sistemas onde há custódia de chaves privadas restringem-se ao nível de confiança 1. O modelo CL-PKC, em sua formulação original, alcança nível de confiança 2: a existência de múltiplos pares de chaves para uma mesma identidade é evidência da ação desonesta por parte da entidade cliente ou do KGC, e não é possível decidir entre ambos.

A propriedade de que a chave pública pode ser gerada antes da chave privada pode ser utilizada para elevar o nível de confiança da autoridade central no modelo CL-PKC, com a inserção da chave pública de A na informação ID_A utilizada para a geração da chave parcial. Esta simples modificação confina a chave privada parcial gerada à uma chave pública específica e permite a transmissão de chaves privadas parciais em claro [12].

A existência de duas chaves públicas funcionais para uma entidade implica a existência de duas chaves privadas parciais para a mesma entidade, o que indica ação desonesta do KGC. Agora, o nível de confiança é superior a 2, mas não é suficiente para atingir o nível 3. Para examinar esta possibilidade, tem-se o seguinte cenário: dada uma entidade A dotada de uma chave pública P_A, a autoridade central gera uma nova chave privada parcial ligando uma chave pública P'_A à identidade de A. Se uma entidade B cifrar uma mensagem para A utilizando a chave pública P'_A, o KGC pode interceptar e decifrar a mensagem para depois recifrá-la sob a chave P_A. A evidência de que existem duas chaves públicas funcionais para uma mesma entidade depende da existência de uma mesma mensagem M cifrada sob chaves públicas distintas. As entidades A e B só podem detectar o ataque se puderem comparar posteriormente as chaves públicas P_A e P'_A utilizadas para cifração, ou seja, é necessário que as entidades ao menos suspeitem que um ataque aconteceu. Infelizmente, não é possível implicar o KGC em atuação maliciosa, pois B poderia ter cifrado M sob chaves distintas P_A e P'_A propositadamente. O ataque do KGC pode ser detectado, entretanto, se B for considerado honesto e garantir para a entidade A que cifrou M com apenas uma chave pública – o KGC é a única entidade que poderia converter a mensagem M cifrada com a chave pública $P'A$ em um criptograma de M cifrado com a chave pública P_A, já que detém a chave privada correspondente à chave pública P'_A. A solução definitiva deste problema vem de um efeito colateral importante do confinamento de chaves públicas: a possibilidade de se transmitir chaves privadas parciais em claro – a chave privada parcial agora está restrita à escolha prévia de uma chave pública, que depende de um segredo conhecido apenas por A. Se as chaves privadas parciais são públicas, a chave privada parcial falsa é evidência da ação desonesta do PKG e o sistema alcança nível de confiança 3, compatível com a PKI tradicional. Portanto, o nível de confiança da formulação original é ligeiramente inferior ao nível 3, e o fator de diferença depende da disponibilidade de chaves privadas parciais ou da honestidade dos participantes [12].

O confinamento de chaves privadas parciais permite que o sistema atinja nível 3 para primitivas que exigem prova de posse de uma chave privada, como autenticação e assinatura.

A existência de duas assinaturas verificáveis sob chaves públicas distintas ou de fragmentos de comunicação que exibem provas de possessão para chaves privadas distintas sempre permitem a detecção de um PKG desonesto. O nível de confiança 3 agrega irretratabilidade à primitiva de assinatura, já que não há a possibilidade de qualquer entidade personificar outra sem ser detectada. Desta forma, esquemas de assinatura baseados no modelo CL-PKC suportam irretratabilidade.

3.2 Trabalhos relacionados

O conceito de Criptografia de Chave Pública Sem Certificados foi inspirado nas chaves *auto-certificadas* propostas por Girault [5]. Uma chave pública é auto-certificada quando é construída em conjunto pelo usuário e pela autoridade central e carrega informação de certificação embutida. O esquema funciona da seguinte forma: ao criar um par de chaves (e_A, d_A), o usuário A entrega a chave pública e_A para a autoridade central. A autoridade central constrói um *testemunho* w a partir da combinação entre a chave pública e_A e a identidade real do usuário. Este testemunho tem valor compatível ao de uma assinatura de autoridade certificadora, legitimando a ligação entre chave e identidade. Após este procedimento, qualquer usuário pode cifrar mensagens para o usuário A utilizando o testemunho w, a identidade de A e a chave pública da autoridade central. O testemunho funciona como um certificado simplificado, que diminui requisitos de comunicação, armazenamento e processamento. Entretanto, uma falha grave foi encontrada no modelo original [15] e permite à autoridade central construir parâmetros públicos vulneráveis à fatoração ou cálculo do logaritmo discreto. Uma autoridade central maliciosa pode assim recuperar mensagens cifradas para os seus usuários. Esta vulnerabilidade abaixa o nível de confiança do esquema para o nível 2 e corrigi-la implica em aumentar significativamente o tamanho dos parâmetros envolvidos, comprometendo as vantagens obtidas com a simplificação dos certificados.

O trabalho de Gentry [16] compartilha uma estrutura similar e utiliza emparelhamentos bilineares para simplificar o procedimento de revogação na PKI tradicional. Neste modelo, a chave privada de uma entidade A consiste em dois componentes: um permanente, gerado por A; e outro emitido temporariamente pela autoridade certificadora. Duas chaves públicas correspondentes completam o conjunto de chaves de A: uma escolhida por A e outra calculada a partir da chave pública da autoridade certificadora, da chave pública escolhida por A e de informação temporal. Por construção, uma mensagem cifrada por qualquer entidade só pode ser decifrada por A se ela detiver as duas chaves privadas. A segunda chave privada fornece certificação implícita: um usuário B ao cifrar uma mensagem para A tem a garantia de que a entidade A só poderá decifrá-la caso tenha realizado o procedimento de certificação no intervalo de tempo corrente. Também não há necessidade de B verificar a chave pública de A previamente: a validade vem da confiança depositada na autoridade certificadora para a distribuição do componente secundário da chave privada de A.

Outros sistemas de certificação implícita foram propostos tentando apresentar propriedades avançadas sem depender de emparelhamentos bilineares. Um sistema CL-PKC eficiente que não utiliza emparelhamentos baseia-se na intratabilidade de problemas convencionais [17]. Entretanto, não fornece a possibilidade de confinar chaves privadas parciais a chaves

públicas geradas previamente. Mais radicalmente, um sistema híbrido entre ID-PKC e a PKI tradicional foi proposto, para argumentar que o paradigma ID-PKC é desnecessário [11]. Neste novo sistema, a autoridade central utiliza sua chave-mestre e a identidade do usuário para alimentar o gerador pseudo-aleatório que produzirá o par de chaves. Os demais procedimentos, incluindo a revogação e verificação, são idênticos à PKI tradicional. Este esquema poderia ser transformado em um híbrido entre CL-PKC e a PKI tradicional mas, como as chaves geradas são convencionais, também não haveria suporte para a geração de chaves públicas antes das chaves privadas parciais correspondentes. Esta desvantagem limita o nível de confiança máximo alcançado pelo sistema, sendo um fator decisivo para sua avaliação.

A proposta original de criptografia sem certificados utiliza emparelhamentos bilineares e permite o confinamento de chaves privadas parciais. A técnica de confinamento agrega irretratabilidade às assinaturas e alcança nível de confiança compatível com a PKI tradicional, sem a necessidade de certificados. Estas são características essenciais para a utilização de um paradigma de criptografia em um ambiente de rede aberta.

3.3 Fundamentos matemáticos

Como visto anteriormente, as formulações de CL-PKC que exibem as propriedades mais atraentes são instanciadas a partir de emparelhamentos bilineares sobre curvas elípticas.

3.3.1 Curvas Elípticas

Seja p um número primo, m um número positivo e $\mathbb{F}_q$ um corpo finito com $q = p^m$ elementos; p é chamado de *característica* do corpo e m de *grau de extensão*. Denota-se por $\mathbb{F}_q^*$ o corpo $\mathbb{F}_q - \{0\}$.

Uma *curva elíptica* E sobre o corpo $\mathbb{F}_q$ é o conjunto de soluções $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ que satisfazem a equação de Weierstrass na forma:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \tag{1}$$

onde $a_i \in \mathbb{F}_q$ para $i = 1, 2, 3, 4, 6$, e um *ponto no infinito* denotado por ∞. Se $\mathbb{K}$ é uma extensão $\mathbb{K} \equiv \mathbb{F}_{q^k}$ do corpo $\mathbb{F}_q$, o conjunto de pontos $\mathbb{K}$-*racionais* de E, denotado por $E(\mathbb{K})$, é o conjunto de pontos $(x, y) \in E$ tais que $x, y \in \mathbb{K}$. O número de pontos da curva $E(\mathbb{K})$, denotado por $\#E(\mathbb{K})$, é chamado de *ordem* da curva sobre o corpo $\mathbb{K}$. A *condição de Hasse* afirma que $\#E(\mathbb{F}_q) = q + 1 - t$, onde $|t| \leq 2\sqrt{q}$ é chamado de *traço de Frobenius*. Curvas em que a característica p divide t são chamadas de *curvas supersingulares*.

Sem perda de generalidade, se a característica do corpo não é 2 ou 3, a equação da curva elíptica pode ser simplificada na forma:

$$y^2 = x^3 + ax + b, \tag{2}$$

onde a, $b \in \mathbb{F}_q$ e o discriminante $\delta = -16(4a^3 + 27b^2) \neq 0$. O *twist quadrático* $E^t(\mathbb{F}_q)$ de uma curva $E(\mathbb{F}_q)$ é dada por $y^2 = x^3 + v^2ax + v^3b$ para todo não-resíduo quadrático $v \in \mathbb{F}_q$.

O conjunto $\{(x, y) \in \mathbb{K} \times \mathbb{K} : E(\mathbb{K})\} \cup \{\infty\}$ sob a operação de grupo $+$, forma um grupo aditivo denotado por $(E(\mathbb{K}), +)$. A operação de adição de pontos no grupo é definida da seguinte forma:

- Seja $P \in E(\mathbb{K})$. Então, $P + \infty = P$ e $\infty + P = P$. O elemento ∞ serve como identidade aditiva para o grupo. Se $P = \infty$, então $-P = \infty$. A notação $E(\mathbb{K})^*$ denota o grupo $E(\mathbb{K})$ excluindo o elemento de identidade ∞;
- Seja $P = (x, y) \in E(\mathbb{K})^*$. Então, $-P = (-x, y) = (x, -y)$ e $P + (-P) = \infty$. O inverso de P é $-P$;
- Seja $P = (x, y) \in E(\mathbb{K})^*$ e $Q = (x', y') \in E(\mathbb{K})^*$. Se $x \neq x'$, então $P + Q = -R$, onde $-R$ é a reflexão do ponto R no eixo x e R é o ponto de intersecção entre a curva elíptica e a linha que passa por P e Q;
- Seja $P = (x, y) \in E(\mathbb{K})^*$. Então, $P + P = -R$, onde $-R$ é a reflexão do ponto R no eixo x. O ponto R é a intersecção entre a curva elíptica e a tangente à curva E que passa por P.

A operação do grupo é comutativa e associativa. Logo, $(E(\mathbb{K}), +)$ é um grupo abeliano com elemento de identidade ∞. A notação mP denota a multiplicação de $P \in E(\mathbb{K})$ por $m \in \mathbb{Z}_q$. O valor de mP é dado pela relação de recorrência:

$$mP = \begin{cases} \infty, & \text{se } m = 0; \\ (-m)(-P) & \text{se } m \leq -1; \\ (m-1)P + P & \text{se } m \geq 1. \end{cases} \tag{3}$$

Seja $n = \#(\mathbb{F}_{q^k})$. A *ordem* de um ponto $P \in E$ é o menor inteiro $r > 0$ tal que $rP = \infty$ e sempre divide a ordem da curva. O conjunto de pontos de torção r de E, denotado por $E(\mathbb{K})[r]$, é o conjunto $\{P \in E(\mathbb{K}) | rP = \infty\}$. Destas definições, segue que $\langle P \rangle$, o grupo de pontos gerado por P, é um subgrupo de $E(\mathbb{K})[r]$, que por sua vez é um subgrupo de $E(\mathbb{K})[n]$. Dizemos que o subgrupo $\langle P \rangle$ tem *grau de mergulho* k se $r | q^k - 1$ e $r \nmid q^s - 1$ para todo $0 < s < k$.

3.3.2 Emparelhamentos Bilineares

Seja $\mathbb{G}_1$ um grupo cíclico aditivo de ordem prima q e $\mathbb{G}_2$ um grupo cíclico multiplicativo tal que $|\mathbb{G}_1| = |\mathbb{G}_2|$. Seja P o gerador de $\mathbb{G}_1$. Um mapeamento $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ é dito um *emparelhamento bilinear admissível* [9] se satisfaz as seguintes propriedades:

1. *Bilinearidade:* dados Q, W, $Z \in \mathbb{G}_1$, temos

$$e(Q, W + Z) = e(Q, W) \cdot e(Q, Z) \text{ e } e(Q + W, Z) = e(Q, Z) \cdot e(W, Z).$$

Conseqüentemente, para quaisquer a, $b \in \mathbb{Z}_q$, temos:

$$e(aQ, bW) = e(Q, W)^{ab} = e(abQ, W) = e(Q, abW) = e(bQ, aW) = e(bQ, W)^a.$$

2. *Não-degeneração:* $e(P, P) \neq 1_{\mathbb{G}_2}$, onde $1_{\mathbb{G}_2}$ é o elemento de identidade do grupo $\mathbb{G}_2$.

3. *Eficiência:* O mapeamento e pode ser calculado eficientemente, ou seja, tem complexidade polinomial.

Tipicamente, $\mathbb{G}_1$ é um subgrupo do grupo de pontos de uma curva elíptica sobre um corpo finito $E(\mathbb{F}_q)$ e $\mathbb{G}_2$ é um subgrupo do grupo multiplicativo de um corpo finito relacionado a $\mathbb{F}_q$ (uma de suas extensões, por exemplo). O mapeamento e é obtido pela modificação do emparelhamento de Weil ou de Tate sobre uma curva elíptica supersingular em $\mathbb{F}_q$ [18].

A definição de emparelhamentos bilineares pode ser generalizada para a construção de *emparelhamentos assimétricos*, ou seja: o mapeamento e é da forma $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, com $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = q$ e $\mathbb{G}_1 \neq \mathbb{G}_2$. Há ainda dois geradores $P \in \mathbb{G}_1^*$ e $Q \in \mathbb{G}_2^*$, tais que $P = \psi(Q)$, onde ψ é um homomorfismo de $\mathbb{G}_2$ em $\mathbb{G}_1$ eficientemente computável.

3.3.3 Problemas Relacionados

Um conjunto de problemas clássicos dos quais se tem evidência de intratabilidade são utilizados explicitamente ou implicitamente por sistemas criptográficos de chave pública. São estes:

Problema do Logaritmo Discreto (Discrete Logarithm Problem – DLP): Seja $\mathbb{G}$ um grupo cíclico finito e g um gerador de $\mathbb{G}$. Dados $\langle g, g^a \rangle$, com escolha uniformemente aleatória de $a \in \mathbb{Z}_{|\mathbb{G}|}$, encontrar $a \in \mathbb{G}$.

O algoritmo mais eficiente para cálculo de logaritmos discretos [19] é uma variação do algoritmo de cálculo de índices [20] e apresenta complexidade sub-exponencial. Para um grupo de pontos em curva elíptica, o Problema do Logaritmo Discreto consiste em obter m a partir do resultado da operação de multiplicação mP. Existem evidências de que a técnica de cálculo de índices não pode ser estendida para grupos de pontos em curvas elípticas [21].

Problema Diffie-Hellman Computacional (Computational Diffie Hellman Problem – CDHP): Seja $\mathbb{G}$ um grupo cíclico finito e g um gerador de $\mathbb{G}$. Dados $\langle g, g^a, g^b \rangle$ com escolha uniformemente aleatória de $a, b \in \mathbb{Z}_{|\mathbb{G}|}$, encontrar $g^{ab} \in \mathbb{G}$.

Problema de Decisão Diffie-Hellman (Decisional Diffie Hellman Problem – DDHP): Seja $\mathbb{G}$ um grupo cíclico finito e g um gerador de $\mathbb{G}$. Dados $\langle g, g^a, g^b, g^c \rangle$ com escolha uniformemente aleatória de $a, b, c \in \mathbb{Z}_{|\mathbb{G}|}$, determinar se $g^{ab} = g^c \in \mathbb{G}$.

A derivação de problemas análogos aos problemas ditos convencionais no contexto de emparelhamentos bilineares é direta. Sistemas criptográficos construídos a partir de emparelhamentos bilineares também utilizam a intratabilidade potencial destes problemas para fornecer segurança e suporte para prova de propriedades específicas. A instanciação de um sistema particular depende da existência de um gerador de parâmetros compatíveis com o

sistema criptográfico considerado.

Gerador de Parâmetros Diffie-Hellman Bilinear: Um algoritmo probabilístico $\mathcal{IG}$ é um *gerador de parâmetros Diffie-Hellman bilinear*, se:

1. $\mathcal{IG}$ recebe um parâmetro de segurança k como entrada, para $k \geq 1$;
2. $\mathcal{IG}$ tem complexidade polinomial em k; e
3. $\mathcal{IG}$ produz um primo q, a descrição dos grupos $\mathbb{G}_1$, $\mathbb{G}_2$ de ordem prima q e um emparelhamento $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.

Problema Diffie-Hellman Bilinear (Bilinear Diffie-Hellman Problem – BDHP): Seja $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$ a saída de um algoritmo $\mathcal{IG}(k)$ e seja P um gerador de $\mathbb{G}_1$. Dados $\langle P, aP, bP, cP \rangle$, com escolhas uniformemente aleatórias de $a, b, c \in \mathbb{Z}_q$, calcular $e(P, P)^{abc} \in \mathbb{G}_2$.

Problema Diffie-Hellman Bilinear Generalizado (Generalized Bilinear Diffie-Hellman Problem – GBDHP): Seja $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$ a saída de um algoritmo $\mathcal{IG}(k)$ e seja P um gerador de $\mathbb{G}_1$. Dados $\langle P, aP, bP, cP \rangle$, com escolhas uniformemente aleatórias de $a, b, c \in \mathbb{Z}_q$, encontrar um par $\langle Q \in \mathbb{G}_1^*, e(P, Q)^{abc} \in \mathbb{G}_2 \rangle$.

Problema de Decisão Diffie-Hellman Bilinear (Decisional Bilinear Diffie-Hellman Problem – DBDHP): Seja $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$ a saída de um algoritmo $\mathcal{IG}(k)$ e seja P um gerador de $\mathbb{G}_1$. Dados $\langle P, aP, bP, cP \rangle$ e $Q = e(P, P)^{abc} \in \mathbb{G}_2^*$, com escolhas uniformemente aleatórias de $a, b, c \in \mathbb{Z}_q$, determinar se $Q = e(P, P)^{abc}$.

Uma conseqüência da bilinearidade é que para instâncias de emparelhamento $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$, o DDHP no grupo $\mathbb{G}_1$ pode ser solucionado em tempo polinomial: determinar se $cP = abP$ para a tupla $\langle P, aP, bP, cP \rangle$, com $a, b, c \in \mathbb{Z}_q^*$ pode ser eficientemente computado verificando se $e(aP, bP) = e(P, cP)$.

A partir dos problemas relacionados, pode-se perceber que a intratabilidade do Problema do Logaritmo Discreto em $\mathbb{G}_1$ depende da intratabilidade do mesmo problema em $\mathbb{G}_2$, visto que o cálculo de logaritmos discretos em $\mathbb{G}_2$ fornece um método para o cálculo de logaritmos discretos em $\mathbb{G}_1$. Assim, o parâmetro de segurança k utilizado deve ser de magnitude suficiente para que o cálculo de logaritmos discretos em $\mathbb{G}_2$ seja difícil.

3.4 Formalização

Formalmente, o modelo CL-PKC compreende sete algoritmos [7]:

Inicializar. Recebe um parâmetro de segurança k e retorna os parâmetros de sistema params e a chave mestre s. Os parâmetros de sistema incluem uma descrições do espaço de mensagens $\mathcal{M}$, do espaço de criptogramas $\mathcal{C}$ e do espaço de assinaturas $\mathcal{S}$. Seguindo a prática comum, os parâmetros de sistema são conhecidos publicamente, enquanto s é mantida em sigilo pelo KGC.

Extrair. Recebe como entrada params, s e um identificador arbitrário $\mathsf{ID}_A \in \{0,1\}^*$ para a entidade A e retorna uma chave privada parcial D_A. Tipicamente, este algoritmo é executado pelo KGC e a chave é transportada para A por meio de um canal opcionalmente autenticado e confidencial (depende do confinamento da chave privada parcial para o identificador, como discutido anteriormente).

Gerar-chaves. Recebe como entrada params e constrói a chave privada S_A e a chave pública P_A para a entidade A. A construção da chave privada completa S_A está condicionada à geração de um valor secreto d_A e sua combinação com a chave privada parcial D_A. A chave pública geralmente pode ser gerada antes da chave privada.

Cifrar. Recebe como entrada params, $M \in \mathcal{M}$, a chave pública P_A e o identificador ID_A da entidade A. Retorna um criptograma $C \in \mathcal{C}$ ou um símbolo nulo $\perp$, indicando falha durante a cifração.

Decifrar. Recebe como entrada params, $C \in \mathcal{C}$, uma chave privada S_A e retorna a mensagem em claro $M \in \mathcal{M}$ ou o símbolo nulo $\perp$ indicando falha durante a decifração (ou seja, a certificação implícita não pôde ser verificada).

Assinar. Recebe como entrada params, uma mensagem $M \in \mathcal{M}$ e uma chave privada S_A e produz uma assinatura $S \in \mathcal{S}$ para M.

Verificar. Recebe como entrada params, a chave pública P_A, o identificador ID_A da entidade A e a assinatura $S \in \mathcal{S}$ para uma mensagem $M \in \mathcal{M}$. Retorna $\mathsf{verdadeiro}$ ou falso, dependendo de quando a assinatura é aceitável, ou $\perp$, caso haja falha durante a verificação.

Normalmente, os algoritmos para geração das chaves privada e pública são executados pela entidade A, após gerar seu segredo d_A. Fica claro perceber que não existe uma ordenação temporal na geração das chaves pública e privada. Supõe-se ainda que d_A é selecionado aleatoriamente em um conjunto de tamanho adequado, e que A é a única entidade que conhece S_A e d_A. Os algoritmos devem conservar propriedades de consistência:

$$\forall M \in \mathcal{M} : \mathsf{Decifrar}(\mathsf{params}, C = \mathsf{Cifrar}(\mathsf{params}, P_A, \mathsf{ID}_A, M), S_A) = M$$

$$\forall M \in \mathcal{M} : \mathsf{Verificar}(\mathsf{params}, S = \mathsf{Assinar}(\mathsf{params}, S_A, M), P_A, \mathsf{ID}_A, M) = \mathsf{verdadeiro}$$

3.5 Modelo de adversário

O adversário para CL-PKC tem poder idêntico ao de um adversário para PKI no que diz respeito à substituição de chaves públicas em repositórios de chaves. Mesmo com a ausência de informação de autenticação para as chaves públicas, pode-se ver que um ataque desta natureza não tem resultados úteis: sem a chave privada correta, cuja produção depende da cooperação do KGC, um adversário não é capaz de decifrar mensagens cifradas com a chave pública falsa nem produzir assinaturas verificáveis com a chave pública falsa.

De forma similar à PKI tradicional, deve-se assumir que o KGC não participa de ataques de substituição de chaves já que, com o poder de gerar um par de chaves para qualquer

entidade e distribuir a chave pública correspondente, o KGC poderia personificar qualquer entidade com sucesso. Assim, é premissa de segurança do sistema que o KGC não distribua chaves públicas falsas. A capacidade de participar em outras atividades maliciosas, como escutas passiva ou ativa, é conservada.

Desta forma, são considerados dois adversários distintos no modelo CL-PKC:

- Um adversário que não tem acesso à chave mestre do KGC, mas pode extrair chaves privadas parciais ou gerar chaves privadas completas, tem acesso às chaves públicas das entidades e pode substituí-las em um repositório público de chaves; e
- Um adversário que detém a chave mestre do KGC e possui acesso às chaves públicas das entidades, mas não pode substituir nenhuma chave pública. A posse da chave mestre permite que o adversário compute chaves privadas parciais e completas para si.

O poder do adversário inclui a habilidade de substituir indefinidamente chaves públicas presentes em um repositório público de chaves. Como resultado, o adversário pode impedir que uma determinada entidade comunique-se com as demais por meio da distribuição de chaves públicas falsas para as entidades comunicantes. Este ataque chama-se *negação de decifração*, por efetivar uma espécie de negação de serviço para a confidencialidade da vítima, e é inerente aos sistemas de cifração baseados no paradigma CL-PKC. A solução deste problema depende da combinação de sistemas criptográficos convencionais de criptografia sem certificados [22].

Outros modelos de adversário foram propostos, em buscas de características mais próximas a ataques reais [23]. O modelo de adversário aqui apresentado é utilizado neste trabalho, porque a maioria dos sistemas criptográficos CL-PKC propostos teve sua segurança foi provada considerando adversários com estas características [23].

4 Sistemas criptográficos

Nesta seção, esquemas criptográficos construídos a partir do conceito de CL-PKC são descritos. Em todos os esquemas, a primitiva é instanciada utilizando emparelhamentos bilineares sobre curvas elípticas. É importante observar que os protocolos a seguir não são os mais eficientes já propostos e têm o procedimento de geração de chaves vulnerável à ação de um KGC malicioso [24]. A apresentação destes protocolos, portanto, tem a finalidade única de ilustrar os conceitos apresentados.

4.1 Cifração

São apresentados dois sistemas criptográficos para cifração. O primeiro sistema é um instanciamento simplificado para destacar a construção. O segundo é uma modificação do primeiro pela aplicação de uma transformação [25], que confere resistência contra ataques adaptativos de criptograma escolhido sob o modelo do oráculo aleatório [26]. Nota-se que o sistema de cifração é uma modificação do sistema ID-PKC de Boneh e Franklin [9], agregando as propriedades que caracterizam um sistema sem certificados.

4.1.1 Sistema Básico

Seja k o parâmetro de segurança do sistema e $\mathcal{IG}$ um gerador de parâmetros BDH com entrada k. São definidos os cinco algoritmos para cifração:

Inicializar. Consiste em:

1. Executar $\mathcal{IG}$ com entrada k e obter como saída a tupla $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem q e $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear;
2. Escolher um gerador arbitrário $P \in \mathbb{G}_1$;
3. Selecionar s uniformemente aleatório em $\mathbb{Z}_q^*$ e atribuir $P_0 = sP$;
4. Escolher funções de *hash* criptográficas $H_1 : \{0,1\}^* \to \mathbb{G}_1$ e $H_2 : \mathbb{G}_2 \to \{0,1\}^n$. O parâmetro n é o comprimento em *bits* das mensagens em texto claro.
5. Retornar os parâmetros de sistema $\mathsf{params} = \langle \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_0, H_1, H_2 \rangle$ e a chave mestre $s \in \mathbb{Z}_q^*$. O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$ e o espaço de criptogramas é $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^n$.

Extrair. Recebe como entrada um identificador $\mathsf{ID}_A \in \{0,1\}^*$ e produz a chave privada parcial para a entidade A. Consiste em:

1. Calcular $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$;
2. Retornar a chave privada parcial $D_A = sQ_A \in \mathbb{G}_1^*$.

Por construção, a entidade A pode verificar a correção do algoritmo de extração testando a condição $e(D_A, P) = e(Q_A, P_0)$.

Gerar-chaves. Recebe como entrada params e seleciona aleatoriamente $d_A \in \mathbb{Z}_q^*$ como o valor secreto da entidade A. Utiliza d_A para transformar D_A na chave privada completa S_A, calculando $S_A = d_A D_A = d_A s Q_A \in \mathbb{G}_1^*$. Constrói a chave pública P_A como $P_A = \langle X_A, Y_A \rangle$, onde $X_A = d_A P$ e $Y_A = d_A P_0 = d_A s P$.

Cifrar. Para cifrar uma mensagem $M \in \mathcal{M}$ para a entidade A com identificador $\mathsf{ID}_A \in \{0,1\}^*$ e chave pública $P_A = \langle X_A, Y_A \rangle$, o algoritmo executa os seguintes passos:

1. Verificar que $X_A, Y_A \in \mathbb{G}_1^*$ e que $e(X_A, P_0) = e(Y_A, P)$. Se qualquer das verificações falhar, retornar $\perp$ e abortar a cifração;
2. Calcular $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$;
3. Escolher um valor aleatório $r \in \mathbb{Z}_q^*$;
4. Calcular e retornar o criptograma:

$$C = \langle rP, M \oplus H_2(e(Y_A, Q_A)^r) \rangle.$$

Decifrar. Seja $C = \langle U, V \rangle \in \mathcal{C}$. Para decifrar o criptograma utilizando a chave privada S_A, calcular e retornar:

$$V \oplus H_2(e(U, S_A)).$$

Se $\langle U = rP, V\rangle$ é o criptograma de M para a entidade A com chave pública $P_A = \langle X_A, Y_A\rangle$, o princípio de consistência é satisfeito:

$$\begin{aligned} V \oplus H_2(e(S_A, U)) &= V \oplus H_2(e(rP, x_A s Q_A)) \\ &= V \oplus H_2(e(x_A s P, Q_A)^r) \\ &= V \oplus H_2(e(Y_A, Q_A)^r) \\ &= M. \end{aligned}$$

4.1.2 Sistema Completo

Tendo a versão básica descrita, pode-se especificar o esquema completo de cifração, aplicando a técnica de transformação Fujisaki-Okamoto [25]:

Inicializar. Idêntico à versão básica com a exceção de que são necessárias duas funções de *hash* criptográficas adicionais $H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_n^*$ e $H_4 : \{0,1\}^n \to \{0,1\}^n$. Retorna os parâmetros de sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_0, H_1, H_2, H_3, H_4\rangle$ e a chave mestre $s \in \mathbb{Z}_q^*$. O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$ e o espaço de criptogramas é $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^{2n}$.

Extrair. Idêntico à versão básica.

Gerar-chaves. Idêntico à versão básica.

Cifrar. Para cifrar uma mensagem $M \in \mathcal{M}$ para a entidade A com identificador $\mathsf{ID}_A \in \{0,1\}^*$ e chave pública $P_A = \langle X_A, Y_A\rangle$, o algoritmo executa os seguintes passos:

1. Verificar que $X_A, Y_A \in \mathbb{G}_1^*$ e que $e(X_A, P_0) = e(Y_A, P)$. Se qualquer das verificações falhar, retornar $\perp$ e abortar a cifração;
2. Computar $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$;
3. Escolher um valor aleatório $\sigma \in \{0,1\}^n$;
4. Fazer $r = H_3(\sigma, M)$;
5. Calcular e retornar o criptograma:

$$C = \langle rP, \sigma \oplus H_2(e(Y_A, Q_A)^r), M \oplus H_4(\sigma)\rangle.$$

Decifrar. Seja $C = \langle U, V, W\rangle \in \mathcal{C}$. Decifrar o criptograma utilizando a chave privada S_A requer:

1. Calcular $\sigma' = V \oplus H_2(e(U, S_A))$;
2. Calcular $M' = W \oplus H_4(\sigma')$;
3. Fazer $r' = H_3(\sigma', M')$ e testar se $U = r'P$. Se não for o caso, retornar $\perp$ e rejeitar o criptograma.
4. Retornar M' como a decifração de C.

Quando C é uma cifração legítima de M sob a chave pública P_A e o identificador (ID_A), a decifração de C irá produzir M:

$$\begin{aligned} V \oplus H_2(e(S_A, U)) &= V \oplus H_2(e(rP, x_A s Q_A)) \\ &= V \oplus H_2(e(x_A s P, Q_A)^r) \\ &= V \oplus H_2(e(Y_A, Q_A)^r) \\ &= \sigma; \\ W \oplus H_4(\sigma) &= M. \end{aligned}$$

4.2 Assinatura

Inicializar. Idêntico ao algoritmo de inicialização para o esquema de cifração, com a exceção de que é usada apenas uma função de *hash* $H_1 : \{0,1\}^* \times \mathbb{G}_2 \to \mathbb{Z}_q^*$. Retorna os parâmetros de sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, e, P, P_0, H_1 \rangle$ e a chave mestre $s \in \mathbb{Z}_q^*$. O espaço de assinaturas é $S = \mathbb{G}_1 \times \mathbb{Z}_q^*$.

Extrair. Idêntico ao algoritmo presente no esquema de cifração.

Gerar-chaves. Idêntico ao algoritmo presente no esquema de cifração.

Assinar. Para assinar uma mensagem $M \in \mathcal{M}$ usando a chave privada S_A, executar os passos:

1. Escolher um valor aleatório $k \in \mathbb{Z}_q^*$;
2. Calcular $r = e(kP, P) \in \mathbb{G}_2$;
3. Fazer $h = H_1(M, r) \in \mathbb{Z}_q^*$;
4. Calcular $S = hS_A + kP \in \mathbb{G}_1$;
5. Retornar a assinatura $\langle S, h \rangle$.

Verificar. Para verificar uma suposta assinatura $\langle S, h \rangle$ de uma mensagem $M \in \mathcal{M}$ para a identidade ID_A e a chave pública $P_A = (X_A, Y_A)$, executar os passos:

1. Verificar que a igualdade $e(X_A, P_0) = e(Y_A, P)$ é verdadeira. Se não for o caso, retornar $\perp$ e abortar a verificação;
2. Calcular $r' = e(P, S) \cdot e(-Y_A, Q_A)^h$;
3. Verificar se $h' = H_1(M, r')$. Se a igualdade for verificada, retornar verdadeiro, caso contrário, retornar falso.

Claramente, a consistência é mantida:

$$\begin{aligned} e(P,S)\cdot e(-Y_A,Q_A) &= e(P,hS_A+kP)\cdot e(-d_AsP,Q_A) \\ &= e(P,hS_A)\cdot e(P,kP)\cdot e(P,Q_A)^{-d_As} \\ &= e(P,d_AsQ_A)\cdot e(kP,P)\cdot e(P,Q_A)^{-d_As} \\ &= e(P,Q_A)^{d_As}\cdot e(P,Q_A)^{-d_As}\cdot e(kP,P) \\ &= r; \\ H(M,r) &= h. \end{aligned}$$

5 Implementação

A concretização da primitiva e dos esquemas de cifração e assinatura consistiu na implementação do emparelhamento de Tate [27]. O emparelhamento de Tate foi escolhido por apresentar melhor desempenho que o emparelhamento de Weil [28].

5.1 Emparelhamento de Tate

A formulação do emparelhamento de Tate parte da teoria de divisores [28]. Para uma curva elíptica com característica $p > 3$, utilizamos a notação $E(x,y) = 0$ para indicar o conjunto de valores (x,y), tais que $E(x,y) = y^2 - (x^3 + ax + b) = 0$, com $x, y \in \mathbb{F}_{q^k}$. A mesma notação é utilizada para uma função $f(x,y) = 0$.

5.1.1 Divisores

Um *divisor* é uma soma formal de pontos na curva $E(\mathbb{F}_{q^k})$:

$$\mathcal{D} = \sum_{P\in E} d_P(P), \tag{4}$$

onde d_P é um inteiro e (P) é um símbolo formal. A adição tem as mesmas propriedades aritméticas da adição de inteiros, com a exceção de que é realizada sobre símbolos formais. Divisores podem envolver símbolos formais para vários pontos na curva elíptica, mas limita-se o enfoque a divisores de funções, por possuírem poucos símbolos. O divisor de uma função $f(x,y) = 0$ contém apenas os termos $d_P(P)$ tais que $P = (x,y)$ está na curva $E(x,y) = 0$ e na função $f(x,y) = 0$, ou seja, os pontos em que E e f se encontram. Para qualquer ponto P na curva elíptica tal que E e f não têm intersecção, $d_p = 0$.

O *grau de um divisor* $\mathcal{D}$ é definido como a soma de seus coeficientes:

$$deg(\mathcal{D}) = \sum_{P\in E} d_P \tag{5}$$

O *suporte de um divisor* $\mathcal{D}$ é o conjunto:

$$sup(\mathcal{D}) = \{P \in E | n_P \neq 0\}. \tag{6}$$

Uma estrutura de grupo abeliano aditivo é definida no conjunto dos divisores pela soma dos coeficientes correspondentes em suas somas formais. Assim:

$$\sum_{P\in E} m_P(P) + \sum_{P\in E} n_P(P) = \sum_{P\in E}(m_P + n_P)(P) \quad e \quad n\mathcal{D} = \sum_{P\in E}(nd_P)(P). \tag{7}$$

5.1.2 Funções racionais sobre uma curva elíptica

Uma função $f(x, y)$ em um corpo finito $\mathbb{F}_q$ é dita racional se

$$f(x, y) = \frac{P(x, y)}{Q(x, y)}, \tag{8}$$

e $P(x, y)$ e $Q(x, y)$ são polinômios em $\mathbb{F}_q$. Um função racional está sobre uma curva $E(\mathbb{F}_q)$ se $f(x, y) = 0$ e $E(x, y) = 0$ têm ao menos uma solução em comum. Utiliza-se a notação $P \in f \cap E$ para um ponto $P = (x, y)$ que satisfaz esta condição.

Seja $f(x, y)$ uma função racional sobre $E(x, y) = 0$. Para um ponto $P \in f \cap E$, P é chamado *zero* se $f(P) = 0$ e *pólo* se $f(P) = \infty$.

Para todo ponto $P \in E(\mathbb{F}_q)$, existe uma função racional u com $u(P) = 0$ que satisfaz a seguinte propriedade: toda função racional não-nula f pode ser escrita como $f(P) = u^d s(P)$ para algum inteiro d e uma função racional s, tal que $s(P) \neq 0, \infty$ [28]. Assim, a *ordem* de P, denotada por ord_P é d. Se P é um zero da função f, então a ordem de P é positiva e P tem *multiplicidade* ord_P. Se P é uma pólo de f, então a ordem de P é negativa e P tem *multiplicidade* $-ord_P$.

5.1.3 Divisor de uma função racional

Seja f uma função racional sobre E. O divisor de F é

$$div(f) = \sum_{P\in E} ord_P(P). \tag{9}$$

O divisor de uma função é chamado *divisor principal*. Um divisor $\mathcal{D}$ é principal se e somente se:

$$deg(\mathcal{D}) = 0 \qquad \text{e} \qquad \sum_{P\in E} d_P P = \infty.$$

Dois divisores $\mathcal{C}$ e $\mathcal{D}$ são *equivalentes* ($\mathcal{C} \sim \mathcal{D}$) se a diferença $\mathcal{C} - \mathcal{D}$ é um divisor principal. Para se avaliar uma função racional f em um divisor $\mathcal{D}$ que satisfaz $sup(div(f)) \cap sup(\mathcal{D}) = \varnothing$, calcula-se [28]:

$$f(\mathcal{D}) = \prod_{P\in sup(\mathcal{D})} f(p)^{n_P}. \tag{10}$$

5.1.4 Definição do emparelhamento

Seja $P \in E(\mathbb{E}_{q^k})[r]$ com r e q co-primos, e seja $\mathcal{D}_P$ um divisor equivalente a $(P) - (\infty)$. Sob estas circunstâncias, o divisor $r\mathcal{D}_P$ é principal e existe uma função f_P tal que $div(f_P) = r\mathcal{D}_P = n(P) - n(\infty)$. O *emparelhamento de Tate* de ordem r é a função

$$e_r : E(\mathbb{F}_q)[r] \times E(\mathbb{F}_{q^k})[r] \rightarrow \mathbb{F}^*_{q^k}, \tag{11}$$

dada por:

$$e_r(P, Q) = f_P(\mathcal{D})^{(q^k-1)/r}. \tag{12}$$

para algum divisor $\mathcal{D} \sim (Q) - (\infty)$.

5.1.5 Cálculo do emparelhamento

O emparelhamento de Tate é um *emparelhamento bilinear* e a computação de $f_P(\mathcal{D})$ é obtida pela aplicação do *Algoritmo de Miller* [29], que substitui o cálculo de $f_P(\mathcal{D})$ pelo cálculo de $f_P(Q)$ e cuja saída define uma potência r-ésima em $\mathbb{F}^*_{q^k}$. A exponenciação final em $(q^k - 1)/r$ é necessária para produzir um resultado único, já que $a^{(q^k-1)} = 1$, para qualquer $a \in \mathbb{F}^*_{q^k}$. Conseqüentemente, o valor após esta exponenciação é uma raiz r-ésima da unidade.

5.2 Parâmetros

A escolha de parâmetros foi dominada pelos requisitos principais de segurança e de viabilidade, considerando o suporte disponível em bibliotecas criptográficas livres. Como os protocolos aqui descritos são destinados a uma aplicação real, o requisito de segurança exigiu uma decisão conservadora na escolha do tamanho dos parâmetros e a viabilidade exigiu código portável, de fácil implementação e desempenho razoável. Utilizou-se, portanto, a formulação do emparelhamento de Tate em curvas não-supersingulares sobre corpos primos. Os mesmos requisitos direcionaram a escolha de um grau de mergulho $k = 2$, pelo bom desempenho, possibilidade de otimizações [27], ampla disponibilidade de implementações de curvas elípticas sobre corpos $\mathbb{F}_p$, facilidade na implementação de aritmética no corpo de extensão $\mathbb{F}_{p^2}$ e por ser improvável o aparecimento de novos ataques que sejam poderosos o suficiente para colocar em risco a segurança da implementação, se feita uma escolha cuidadosa no tamanhos dos parâmetros [27]. Nesta instanciação particular, ataques que utilizam o cálculo de índices estão disponíveis, sendo improvável o surgimento de ataques mais poderosos.

5.2.1 Corpo finito

O corpo finito escolhido é o corpo primo $\mathbb{F}_p$, com $p = 3 \bmod 4$. Para esta escolha particular de p, temos que $v = -1$ é sempre não-resíduo quadrático, o que permite a implementação eficiente de operações de quadrado e multiplicação na aritmética da extensão quadrática de $\mathbb{F}_p$ [27]. O tamanho de p foi fixado em 512 *bits*, considerando que o melhor ataque

conhecido (subexponencial) no contexto considerado corresponde ao cálculo do logaritmo discreto no corpo $\mathbb{F}_{p^2}$, onde cada elemento tem 1024 *bits*. Um ataque desta natureza tem poder semelhante a um ataque no sistema criptográfico RSA de 1024 *bits*, que é considerado seguro para o poder computacional vigente.

5.2.2 Curva

A curva escolhida é da forma

$$E : y^2 = x^3 - 3x + b, \tag{13}$$

com $b \in \mathbb{F}_p$. Se $x, y \in \mathbb{F}_p$, a curva tem $\#E(\mathbb{F}_p) = p + 1 - t$ pontos, onde t é o traço de Frobenius. Se $x, y \in \mathbb{F}_{p^2}$, a curva tem $\#E(\mathbb{F}_{p^2}) = (p + 1 - t)(p + 1 + t)$ pontos. Como $p = 3 \bmod 4$, o *twist* $E^t(\mathbb{F}_p)$ da curva é da forma

$$E : y^2 = x^3 - 3x - b. \tag{14}$$

e tem $p + 1 + t$ pontos, se $x, y \in \mathbb{F}_p$. Considerando o nível de segurança de sistemas criptográficos de curvas elípticas não-supersingulares, é desejável que $E(\mathbb{F}_p)$ tenha um subgrupo de ordem r com pelo menos 160 *bits*. Por questões de eficiência, r deve ter baixo peso de Hamming, de preferência um primo de Solinas como $r = 2^{159} + 2^{17} + 1$. Além disso, o grupo de pontos de ordem r em $E(\mathbb{F}_{p^2})$ deve ter grau de mergulho $k = 2$ e a condição $r|p+1$ deve ser mantida para que os parâmetros possibilitem o cálculo do emparelhamento de Tate.

Particularmente, foi escolhida a curva (em base hexadecimal) [27]:

```
 p = 0xDF9BD3ED0034174E54597AA4E2AB033D21C7F6F1AFDD080D4708BC67CAC2AED5
       54FE43F3DA7CD547ED458502C46356BB2A76688DDF064094EBE7785EDE2E413F
 a = 0xDF9BD3ED0034174E54597AA4E2AB033D21C7F6F1AFDD080D4708BC67CAC2AED5
       54FE43F3DA7CD547ED458502C46356BB2A76688DDF064094EBE7785EDE2E413C
 b = 0xCFEC8DDB4E226F34828D4F9B30571BB52E14D1611FA34031423862B3ACB17910
       2A1C152E860FC993A87999CB6A8539516C04950344270037ABC0905175FD47E
#E = 0xDF9BD3ED0034174E54597AA4E2AB033D21C7F6F1AFDD080D4708BC67CAC2AED3
       767AC584178BA7D62E6F13DDC46356BB2A6EEE7C284037F0B03E22219BED5EF6
 t = 0x1DE837E6FC2F12D71BED6712500000000000077A11B6C608A43BA9563D4240E24A
 r = 0x8000000000000000000000000000000020001
```

5.3 Algoritmo

Considerando o grau de mergulho $k = 2$, o emparelhamento de Tate toma dois pontos P e Q na curva elíptica $E(\mathbb{F}_{p^2})$, com P de ordem r, e retorna um elemento em $\mathbb{F}_{p^2}$. O algoritmo consiste em duas etapas: a aplicação do Algoritmo de Miller e uma exponenciação final.

O Algoritmo de Miller inicialmente efetua uma multiplicação implícita de P por r, utilizando o algoritmo padrão de multiplicação. O resultado desta multiplicação é ∞, visto que P tem ordem r. Em cada etapa do algoritmo, um valor $m \in \mathbb{F}_{p^2}$ é calculado a partir de uma relação de distância entre a linha ou tangente atual e o ponto Q. Este valor é acumulado e sua avaliação final é a saída do algoritmo. Para garantir que esta saída seja

única, o resultado final é elevado a $(p^2-1)/r = (p-1)(p+1)/r$. Se $\bar{m}$ é o *conjugado* de m, a potência $m^{(p-1)}$ pode ser calculada como $\bar{m}/m$, já que para $m \in \mathbb{F}_{p^2}$, temos que $m^{(p-1)} = m^p/m = \bar{m}/m$ [30]. A ponto $Q \in E(\mathbb{F}_{p^2})$ pode ser tratado como um ponto no *twist* $E^t(\mathbb{F}_p)$ da curva $E(\mathbb{F}_p)$, porque existe um projeção eficiente $\psi : E(\mathbb{F}_{p^2}) \rightarrow E^t(\mathbb{F}_p)$ dado por $\psi(([-x,0],[0,y])) = (x,y)$ [31]. Isto facilita a implementação de operações sobre Q, como requisitado em protocolos que exigem que Q tenha ordem r [9].

O Algoritmo 1 detalha o cálculo do emparelhamento. A função $f(A,B,Q)$ avalia o divisor da linha que passa por A e B no ponto Q, a contribuição da adição ou duplicação mais recente para a variável m. Esta função encontra-se detalhada no Algoritmo 2. A exponenciação final pode ser calculada a partir de seqüências de Lucas [30], já que o valor de m após a décima linha é unitário.

Algoritmo 1 Emparelhamento de Tate [27].

Entrada: Pontos P e Q linearmente independentes, fator r de $\#E(\mathbb{F}_p)$, ordem p do corpo finito subjacente.
Saída: $e(P,Q)$.

1: $m = 1$
2: $A = P$
3: $n = r - 1$
4: **for** $i = \lfloor lg(n) - 2 \rfloor$ **downto** 0 **do**
5: $\quad m = m^2 \cdot f(A,A,Q)$
6: $\quad$ **if** $n_i = 1$ **then**
7: $\quad\quad m = m \cdot f(A,P,Q)$
8: $\quad$ **end if**
9: **end for**
10: $m = \bar{m}/m$
11: $m = m^{(p+1)/r}$
12: **return** m

Algoritmo 2 Função f [27].

Entrada: Pontos A, B e Q.
Saída: Avaliação do divisor da linha que passa por A e B no ponto Q.
Nota: λ é o coeficiente angular da linha que passa por A e B. O operador $\leftarrow$ indica a extração das coordenadas do ponto. O valor i denota um não-resíduo quadrático tal que $i^2 = -1$.

1: $\lambda = A.add(B)$
2: $(x,y) \leftarrow A$
3: $(a,d) \leftarrow Q$
4: **return** $y - \lambda(a+x) - di$

5.4 Protocolos

Foram implementados protocolos de cifração e assinatura. Ambos os protocolos corrigem a vulnerabilidade ao ataque de um KGC malicioso e têm computação mais eficiente do que os apresentados anteriormente.

5.4.1 Cifração

Este protocolo é seguro contra ataques adaptativos de criptograma escolhido sob o modelo do oráculo aleatório e apresenta uma forma eficiente de cifração, onde apenas um emparelhamento por operação de cifração ou decifração é necessário [32].

Inicializar. Consiste em:

1. Executar $\mathcal{IG}$ com entrada k e obter como saída a tupla $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem q e $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear;
2. Escolher um gerador arbitrário $P \in \mathbb{G}_1^*$;
3. Selecionar s uniformemente aleatório em $\mathbb{Z}_q^*$ e atribuir $P_0 = sP$;
4. Escolher funções de *hash* criptográficas $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1 \to \{0,1\}^n$, $H_3 : \{0,1\}^n \times \{0,1\}^n \to \mathbb{Z}_q^*$ e $H_4 : \{0,1\}^n \to \{0,1\}^n$. A parâmetro n é o comprimento em *bits* das mensagens em texto claro;
5. Retornar os parâmetros de sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, e, n, P, P_0, H_1, H_2, H_3, H_4 \rangle$ e a chave mestre $s \in \mathbb{Z}_q^*$. O espaço de mensagens é $\mathcal{M} = \{0,1\}^n$ e o espaço de criptogramas é $\mathcal{C} = \mathbb{G}_1 \times \{0,1\}^{2n}$.

Extrair. Recebe como entrada um identificador $\mathsf{ID}_A \in \{0,1\}^*$ e executa os seguintes passos para produzir a chave privada parcial para a entidade A:

1. Calcular $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$;
2. Retornar a chave privada parcial $D_A = sQ_A \in \mathbb{G}_1^*$.

Gerar-chaves. Recebe como entrada params e seleciona aleatoriamente $d_A \in \mathbb{Z}_q^*$ como o valor secreto da entidade A. Retorna a chave privada completa $S_A = (d_A, D_A)$ e constrói a chave pública $P_A = d_A P$.

Cifrar. Cifrar uma mensagem $M \in \mathcal{M}$ para a entidade A com identificador $\mathsf{ID}_A \in \{0,1\}^*$ e chave pública P_A requer os seguintes passos:

1. Calcular $Q_A = H_1(\mathsf{ID}_A) \in \mathbb{G}_1^*$;
2. Escolher um valor aleatório $\sigma \in \{0,1\}^n$;
3. Fazer $r = H_3(\sigma, M)$;
4. Calcular e retornar o criptograma:

$$C = \langle rP, \sigma \oplus H_2(rP, e(P_0, Q_A)^r, rP_A), M \oplus H_4(\sigma) \rangle.$$

Decifrar. Seja $C = \langle U, V, W \rangle \in \mathcal{C}$. Para decifrar o criptograma utilizando a chave privada S_A:

1. Calcular $\sigma' = V \oplus H_2(U, e(U, D_A), d_A U)$;
2. Calcular $M' = W \oplus H_4(\sigma')$;
3. Fazer $r' = H_3(\sigma', M')$ e testar se $U = r'P$. Se não for o caso, retornar $\perp$ e rejeitar o criptograma;
4. Retornar M' como a decifração de C.

Se $\langle U = rP, V, W \rangle$ é o criptograma de M para a entidade A com chave pública $P_A = d_A P$, o princípio de consistência é satisfeito:

$$\begin{aligned} V \oplus H_2(U, e(U, D_A), d_A U) &= V \oplus H_2(rP, e(rP, sQ_A), d_A U) \\ &= V \oplus H_2(rP, e(P, sQ_A)^r, d_A U) \\ &= V \oplus H_2(rP, e(P_0, Q_A)^r, d_A U) \\ &= \sigma; \\ W \oplus H_4(\sigma) &= M. \end{aligned}$$

5.4.2 Assinatura

O protocolo de assinatura [33] tem segurança provada sob o modelo do oráculo aleatório. Consiste na adaptação de um sistema eficiente de assinaturas baseado em identidades também provado seguro sob o mesmo modelo [34].

Inicializar. Consiste em:

1. Executar $\mathcal{IG}$ com entrada k e obter como saída a tupla $\langle \mathbb{G}_1, \mathbb{G}_2, e \rangle$, onde $\mathbb{G}_1$ e $\mathbb{G}_2$ são grupos de ordem q e $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ é um emparelhamento bilinear;
2. Escolher dois geradores arbitrários linearmente independentes $P, Q \in \mathbb{G}_1^*$;
3. Calcular $g = e(P, Q) \in \mathbb{G}_2$;
4. Selecionar s uniformemente aleatório em $\mathbb{Z}_q^*$ e atribuir $Q_0 = sQ$;
5. Escolher funções de *hash* criptográficas $H_1 : \{0,1\}^* \to Z_q^*$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}_2 \times \mathbb{G}_2 \to \mathbb{Z}_q^*$;
6. Retornar os parâmetros de sistema $\mathsf{params} = \langle q, \mathbb{G}_1, \mathbb{G}_2, e, g, P, Q, Q_0, H_1, H_2 \rangle$ e a chave mestre $s \in \mathbb{Z}_q^*$. O espaço de mensagens é $\mathcal{M} = \{0,1\}^*$ e o espaço de assinaturas é $\mathcal{S} = \mathbb{G}_1 \times \mathbb{Z}_q^*$.

Extrair. Recebe como entrada um identificador $\mathsf{ID}_A \in \{0,1\}^*$, calcula e retorna a chave privada parcial $D_A = (H_1(\mathsf{ID}_A) + s)^{-1} P \in \mathbb{G}_1^*$;

Gerar-chaves. Recebe como entrada params e seleciona aleatoriamente $d_A \in \mathbb{Z}_q^*$ como o valor secreto da entidade A. Retorna a chave privada completa $S_A = (d_A, D_A)$ e constrói a chave pública $P_A = g^{d_A} \in \mathbb{G}_2$.

Assinar. Assinar uma mensagem $M \in \mathcal{M}$ usando a chave privada S_A da identidade requer os seguintes passos:

1. Escolher um valor aleatório $k \in \mathbb{Z}_q^*$;
2. Calcular $r = g^k \in \mathbb{G}_2$;
3. Fazer $h = H_2(M, \mathsf{ID}_A, P_A, r) \in \mathbb{Z}_q^*$;
4. Calcular $S = (k + hd_A)D_A \in \mathbb{G}_1$;
5. Retornar a assinatura $\langle S, h \rangle$.

Verificar. Verificar uma suposta assinatura $\langle S, h \rangle$ de uma mensagem $M \in \mathcal{M}$ para a identidade ID_A com chave pública P_A requer os seguintes passos:

1. Calcular $r' = e(S, H_1(\mathsf{ID}_A)Q + Q_0)(P_A)^{-h}$;
2. Verificar se $h = H_2(M, \mathsf{ID}_A, P_A, r')$. Se a igualdade for verificada, retornar verdadeiro, caso contrário, retornar falso.

Claramente, o algoritmo de verificação retorna verdadeiro para uma assinatura legítima:

$$\begin{aligned} e(S, H_1(\mathsf{ID}_A)Q + Q_0)(P_A)^{-h} &= e((k + hd_A)D_A, H_1(\mathsf{ID}_A)Q + sQ) \cdot (g^{d_A})^{-h} \\ &= e((H_1(\mathsf{ID}_A) + s)^{-1}P, (H_1(\mathsf{ID}_A) + s)Q)^{k+hda} \cdot (g^{d_A})^{-h} \\ &= e(P, Q)^{(H_1(\mathsf{ID}_A)+s)^{-1}(H_1(\mathsf{ID}_A)+s)(k+hda)} \cdot g^{-hd_A} \\ &= e(P, Q)^{(k+hda)} \cdot e(P, Q)^{-hd_A} \\ &= e(P, Q)^k \\ &= r; \\ H_2(M, \mathsf{ID}_A, P_A, r) &= h. \end{aligned}$$

5.5 Resultados

Na Tabela 1, encontra-se o custo de computação dos protocolos implementados de cifração e assinatura, em termos de operações de emparelhamento bilinear (e), exponenciação no grupo multiplicativo $\mathbb{G}_2$ (a^x), multiplicação de ponto por escalar no grupo aditivo $\mathbb{G}_1$ (mP), multi-plicação no grupo multiplicativo $\mathbb{G}_2$ ($\times$) e aplicações de funções de *hash* (h). As operações encontram-se em ordem decrescente de complexidade. As funções de *hash* foram implementadas a partir da função *SHA-1* e dos Algoritmos 3 e 4.

A implementação propriamente dita foi realizada utilizando a linguagem C, o compilador GCC 3.4.6 e a biblioteca *OpenSSL*[1] 0.9.8d para aritmética em corpos finitos e curvas elípticas. A aritmética na extensão quadrática $\mathbb{F}_{p^2}$ foi implementada a partir das operações no corpo finito $\mathbb{F}_p$ disponibilizadas pela biblioteca. As otimizações implementadas foram:

- Utilização de coordenadas projetivas;

[1] *The OpenSSL Project:* `http://www.openssl.org`

	Operações				
Operação	e	a^x	mP	$\times$	h
Cifração CL-PKC	1	1	2	0	4
Decifração CL-PKC	1	0	2	0	3
Assinatura CL-PKC	0	1	1	0	2
Verificação CL-PKC	1	1	1	1	1

Tabela 1: Custo computacional em operações executadas dos protocolos implementados.

- Aritmética eficiente na extensão quadrática, pela escolha de um grau de mergulho $k = 2$. Cada operação de multiplicação foi implementada com 3 multiplicações modulares e cada operação de quadrado foi implementada com 2 multiplicações modulares. Foi utilizado o método de Karatsuba para a multiplicação e a relação $(a + bi)^2 = (a + b)(a - b) + 2abi$ para o quadrado [27];
- Fator r da ordem da curva com baixo peso de Hamming [27];
- Cálculo de potências do emparelhamento como avaliações de seqüência de Lucas [30];
- Compressão do emparelhamento[30];
- Eliminação do denominador do acumulador [18] no Algoritmo de Miller;
- Cálculo encapsulado do acumulador [35] dentro das operações de duplicação e soma de pontos no Algoritmo de Miller; e
- Isomorfismo entre subgrupo da curva na extensão quadrática $E(\mathbb{F}_{q^2})$ e *twist* $E^t(\mathbb{F}_q)$ [27].

Na Tabela 2, é apresentado o tempo médio de cifrações, decifrações, assinaturas e verificações. O tempo destas operações inclui a aplicação de funções de *hash SHA-1* e chamada da cifra simétrica *AES-128*. O tempo de execução de diversas operações aritméticas implementadas na biblioteca *OpenSSL* é apresentado para referência, em conjunção com o custo de assinaturas e cifras de chave pública comumente utilizados. Todos os tempos apresentados são tomados em uma plataforma Intel Core 2 Duo 2.0 GHz, utilizando apenas um processador. Os resultados obtidos são compatíveis com os resultados publicados das implementações do emparelhamento de Tate com a mesma escolha de parâmetros [27].

A diferença entre os tempos de execução de operações CL-PKC e RSA é justificada pela complexidade dos protocolos de criptografia sem certificados, que exigem maior número de aplicações de funções de *hash* e cifras simétricas, e pelo tempo de execução do emparelhamento. O tempo de execução de operações RSA é claramente dominado pela exponenciação modular. Se compararmos diretamente o tempo de decifração e assinatura RSA com o tempo de execução de um emparelhamento bilinear, um emparelhamento corresponde ao quádruplo do tempo de execução de uma exponenciação modular dos protocolos RSA, sendo viável para aplicações reais. Aprimoramentos podem ser obtidos a partir da utilização de emparelhamentos mais eficientes [37].

Algoritmo 3 *Hash* para um inteiro modulo p [36].

Entrada: Cadeia de caracteres s de comprimento $|s|$ e inteiro p.
Saída: Inteiro no intervalo $[0, p-1]$.
Nota: A função executa l aplicações da função de *hash SHA-1*, com l escolhido tal que, para entrada aleatória, a saída é quase uniforme no intervalo $[0, p-1]$ com uma não-uniformidade estatística não superior a $\frac{1}{\sqrt{p}}$. O operador $||$ representa a concatenação de duas cadeias de caracteres e o operador $\leftarrow$ representa a decodificação de uma cadeia de caracteres para um inteiro positivo em convenção *big-endian*.

1: $v_0 = 0$
2: $h_0 = \texttt{0x}0000000000000000000000000000000000000000$
3: $l = \lceil \frac{3}{5} \lceil \log_2 p \rceil \rceil$
4: **for** $i = 0$ **to** l **do**
5: $\quad t_i = h_{i-1} || s$
6: $\quad h_i =$*SHA-1*(t_i)
7: $\quad a_i \leftarrow h_i$
8: $\quad v_i = 256^{20} \cdot v_{i-1} + a_i \bmod p$
9: **end for**
10: **return** v_l

Algoritmo 4 *Hash* para subgrupo de ordem r de pontos na curva $E(\mathbb{F}_p)$ [36].

Entrada: cadeia de caracteres s, cofator $h = \frac{\#E(\mathbb{F}_p)}{r}$.
Saída: Ponto P de ordem r na curva $E(\mathbb{F}_p)$.
Nota: O operador $||$ representa a concatenação de duas cadeiras de caracteres. A função HASH implementa o Algoritmo 3. O operador $\leftarrow$ descomprime o ponto da curva elíptica armazenado na forma comprimida $(x, \{0, 1\})$.

1: $i = 0$
2: **repeat**
3: $\quad t = s||i$
4: $\quad x =$HASH(t_i, r)
5: $\quad b = i \bmod 2$ $\{bit$ menos significativo de $i\}$
6: $\quad P \leftarrow (x, b)$
7: $\quad i = i + 1$
8: **until** P seja um ponto válido na curva $E(\mathbb{F}_p)$
9: **return** hP

Operação	Tempo de execução (μs)
Quadrado na extensão quadrática	2
Multiplicação na extensão quadrática	4
Inversão na extensão quadrática	134
Duplicação encapsulada de ponto na curva elíptica	28
Adição encapsulada de pontos da curva elíptica	38
Exponenciação final com seqüência de Lucas	720
Hash para um inteiro (Algoritmo 3)	169
Hash para um ponto na curva (Algoritmo 4)	1078
Multiplicação de gerador de subgrupo da curva elíptica†	346
Multiplicação de ponto na curva elíptica†	1690
Emparelhamento	6353
Exponenciação de um emparelhamento comprimido	486
Multiplicação de emparelhamentos comprimidos	25
Geração de Parâmetros (Cifração)	15615
Geração de Parâmetros (Assinatura)	24796
Extração (Cifração)	2727
Extração (Assinatura)	534
Geração de chaves pública e privada	507
Cifração CL-PKC	9654
Decifração CL-PKC	8722
Assinatura CL-PKC	2476
Verificação CL-PKC	8049
Cifração RSA-1024†	45
Decifração RSA-1024†	1099
Assinatura RSA-1024†	1542
Verificação RSA-1024†	87

As rotinas marcadas com (†) são rotinas da biblioteca *OpenSSL*;
As demais rotinas foram implementadas por completo, utilizando aritmética da biblioteca *OpenSSL*.

Tabela 2: Custo computacional em tempo de execução dos protocolos implementados. Os tempos são tomados em um processador da plataforma Intel Core 2 Duo 2.0 GHz e representam a média aritmética dos tempos de execução de 1000 cópias da rotina apresentada, com entradas aleatórias.

6 Conclusões

Neste trabalho, foi apresentado um novo paradigma para definição de sistemas criptográficos de chave pública, chamado Criptografia de Chave Pública sem Certificados (CL-PKC). O objetivo primordial deste novo modelo é utilizar as facilidades advindas de sistemas criptográficos baseados em identidade, porém eliminando as principais desvantagens inerentes a tais sistemas, como a custódia de chaves privadas e os riscos associados. A construção de um sistema genérico CL-PKC foi apresentada em termos da modificação de um sistema ID-PKC típico, e as propriedades do sistema resultante foram discutidas, especialmente a confiança conferida à autoridade central considerando ataques válidos no modelo de adversário definido. Finalmente, esquemas de encriptação e assinatura que utilizam o paradigma foram apresentados utilizando emparelhamentos bilineares de curvas elípticas e detalhes de implementação foram discutidos. Ao compararmos suas características e propriedades com a PKI tradicional e sistemas baseados em identidade, o modelo de Criptografia sem Certificados destaca-se como uma alternativa viável para a construção de infra-estruturas modernas de criptografia de chave pública.

Referências

[1] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22:644–654, November 1976.

[2] L. M. Kohnfelder. Towards a practical public-key cryptosystem. B.S. Thesis, supervised by L. Adleman, May 1978.

[3] P. Gutman. PKI: it's not dead, just resting. *IEEE Computer*, 35(8):41–49, 2002.

[4] A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology (CRYPTO '84)*, pages 47–53, New York, NY, USA, 1984. Springer-Verlag New York, Inc.

[5] M. Girault. Self-certified public keys. In *EUROCRYPT '91*, pages 490–497. Springer, 1991. LCNS vol.547.

[6] Z. Cheng, R. Comley, and L. Vasiu. Remove key escrow from the Identity-Based Encryption System. In *IFIP TCS*, pages 37–50, 2004.

[7] S. S. Al-Riyami and K. G. Paterson. Certificateless Public Key Cryptography. Cryptology ePrint Archive, Report 2003/126, 2003.

[8] C. Cocks. An Identity Based Encryption Scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, pages 360–363, London, UK, 2001. Springer-Verlag.

[9] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In *CRYPTO '01: Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 213–229, London, UK, 2001. Springer-Verlag.

[10] Y. Hanaoka, G. Hanaoka, J. Shikata, and H. Imai. Identity-Based Hierarchical Strongly Key-Insulated Encryption and its application. Cryptology ePrint Archive, Report 2004/338, 2004. `http://eprint.iacr.org/2004/338`.

[11] J. Callas. Identity-based encryption with conventional public-key infrastructure, April 2005. PKI' 05.

[12] S. S. Al-Riyami. *Cryptographic schemes based on elliptic curve pairings*. PhD Thesis, Department of Mathematics, Royal Holloway, University of London, 2005.

[13] D. H. Yum and P. J. Lee. Generic construction of Certificateless Encryption. In *EUROPKI '04*, volume 3043 of *Lecture Notes in Computer Science*, pages 802–811. Springer-Verlag, 2004.

[14] D. H. Yum and P. J. Lee. Generic construction of Certificateless Signature. In *ACISP '04*, volume 3108 of *Lecture Notes in Computer Science*, pages 200–211. Springer-Verlag, 2004.

[15] S. Saeednia. A note on Girault's self-certified model. *Inf. Process. Lett.*, 86(6):323–327, 2003.

[16] C. Gentry. Certificate-Based Encryption and the certificate revocation problem. Cryptology ePrint Archive, Report 2003/183, 2003.

[17] J. Baek, R. Safavi-Naini, and W. Susilo. Certificateless public key encryption without pairing. In *ISC*, pages 134–148, 2005.

[18] P. S. L. M. Barreto, H. Y. Kim, B. Lynn, and M. Scott. Efficient algorithms for pairing-based cryptosystems. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 354–368, London, UK, 2002. Springer-Verlag.

[19] D. M. Gordon. Discrete logarithms in GF(p) using the number field sieve. In *Siam Discrete Math*, volume 6, pages 124–138, 1993.

[20] L. M. Adleman. A subexponential algorithm for the discrete logarithm problem with applications to cryptography. In *Proc. 18th IEEE Symp. on Foundations of Comp. Science*, pages 55–60, Providence, 1977. IEEE.

[21] J. H. Silverman and J. Suzuki. Elliptic curve discrete logarithms and the index calculus. In *Proceedings of the International Conference on the Theory and Applications of Cryptology and Information Security (ASIACRYPT '98)*, pages 110–125, London, UK, 1998. Springer-Verlag.

[22] J. K. Liu, M. H. Au, and W. Susilo. Self-Generated-Certificate Public Key Cryptography and certificateless signature/encryption scheme in the Standard Model. Cryptology ePrint Archive, Report 2006/373, 2006.

[23] A. W. Dent. A survey of Certificateless Encryption schemes and security models. Cryptology ePrint Archive, Report 2006/211, 2006.

[24] M. H. Au, J. Chen, J. K. Liu, Y. Mu, D. S. Wong, and G. Yang. Malicious KGC attack in Certificateless Cryptography. Cryptology ePrint Archive, Report 2006/255, 2006.

[25] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, pages 537–554, London, UK, 1999. Springer-Verlag.

[26] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communications Security*, pages 62–73. ACM, 1993.

[27] M. Scott. Computing the Tate pairing. In *Topics in Cryptology/CT-RSA '05*, volume 3376 of *Lecture Notes in Computer Science*, pages 293–304. Springer Berlin/Heidelberg, 2005.

[28] Martijn Maas. Pairing-based cryptography. Master's thesis, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, 2004.

[29] V. Miller. Short programs for functions on curves. Manuscrito não-publicado, 1986.

[30] P. S. L. M. Barreto and M. Scott. Compressed pairings. In *Advances of Cryptology (CRYPTO '04)*, volume 3152 of *Lecture Notes in Computer Science*, pages 140–156. Springer-Verlag, 2004. `http://eprint.iacr.org/2004/032/`.

[31] P. S. L. M. Barreto, B. Lynn, and M. Scott. Constructing elliptic curves with prescribed embedding degrees. Cryptology ePrint Archive, Report 2002/088, 2002.

[32] Z. Cheng and R. Comley. Efficient Certificateless Public Key Encryption. Cryptology ePrint Archive, Report 2005/12, 2005.

[33] D. H. Goya. Proposta de esquemas de criptografia e assinatura sob modelo de criptografia de chave pública sem certificado. Master's thesis, Instituto de Matemática e Estatística / Universidade de São Paulo, 2006.

[34] P. S. L. M. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In *Advances in Cryptology (ASYACRYPT '05)*, Lecture Notes in Computer Science, pages 515–532. Springer Berlin / Heidelberg, 2005.

[35] S. Chatterjee, P. Sarkar, and R. Barua. Efficient computation of Tate pairing in projective coordinates over general characteristic fields. In *Information Security and Cryptology (ICISC '04)*, Lecture Notes in Computer Science, pages 168–181. Springer Berlin / Heidelberg, 2005.

[36] X. Boyen and L. Martin. Identity-Based Cryptography Standard (IBCS) #1: Supersingular curve implementations of the BF and BB1 cryptosystems. S/MIME Working Group - Internet Draft, June 2006. `http://www.ietf.org/internet-drafts/draft-ietf-smime-ibcs-00.txt`.

[37] P. S. L. M. Barreto, S. Galbraith, C. Ó hÉigeartaigh, and M. Scott. Pairing computation on supersingular abelian varieties. Cryptology ePrint Archive, Report 2004/375, 2004.