PAULO SÉRGIO L. M. BARRETO

CRIPTOGRAFIA ROBUSTA E MARCAS D'ÁGUA FRÁGEIS: CONSTRUÇÃO E ANÁLISE DE ALGORITMOS PARA LOCALIZAR ALTERAÇÕES EM IMAGENS DIGITAIS

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Engenharia.

São Paulo
2003

PAULO SÉRGIO L. M. BARRETO

CRIPTOGRAFIA ROBUSTA E MARCAS D'ÁGUA FRÁGEIS: CONSTRUÇÃO E ANÁLISE DE ALGORITMOS PARA LOCALIZAR ALTERAÇÕES EM IMAGENS DIGITAIS

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Engenharia.

Área de Concentração:

Sistemas Eletrônicos

Orientador:

Prof. Dr. Hae Yong Kim

São Paulo
2003

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 29 de outubro de 2003.

Assinatura do autor

Assinatura do orientador

FICHA CATALOGRÁFICA

Barreto, Paulo Sérgio Licciardi Messeder
Criptografia robusta e marcas d'água frágeis: construção e análise de algoritmos para localizar alterações em imagens digitais/ P. S. L. M. Barreto. – ed. rev. – São Paulo, 2003.
150 p.

Tese (Doutorado) — Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1. Criptologia. 2. Processamento de imagens. 3. Algoritmos. I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos. II. t.

AGRADECIMENTOS

É um esforço fútil tentar listar todas as pessoas a quem estou em dívida pelo apoio e incentivo que me concederam. Mesmo assim, gostaria de expressar meu agradecimento a algumas delas:

- Meus pais, minha esposa e meus filhos, pela compreensão além da conta e ilimitada paciência frente aos meus esforços durante a pós-graduação;
- Meu orientador, Prof. Dr. Hae Yong Kim, por seu incentivo, estímulo e apoio;
- Vincent Rijmen, por nortear-me pacientemente nas complexidades da criptoanálise e nos segredos da redação de artigos sobre criptografia;
- Ben Lynn, Jorge Nakahara Jr. e Mike Scott, pelo frutífero intercâmbio de idéias;
- Waldyr Dias Benits Jr. e Cesar Alison Monteiro Paixão, por sua acurada revisão da tese ainda em estado bruto e por seus valiosos comentários que permitiram refiná-la.

Peço, enfim, perdão a tantos outros que, tendo parte no mérito de me conduzirem à conclusão desta pesquisa, não se encontram aqui elencados. Espero que, de alguma forma, possam sentir-se representados nesta sucinta lista, e igualmente abrangidos pela minha gratidão.

RESUMO

A manutenção da integridade, autenticidade e irretratabilidade de imagens, bem como de outros sinais digitais derivados de informações originalmente analógicas, pode ser obtida através de *marcas d'água digitais*. Em particular, marcas d'água *topológicas* são capazes não só de detectar, mas também de localizar alterações numa imagem marcada com uma resolução previamente estabelecida. A natureza do objetivo desse tipo de marca d'água sugere uma associação com algoritmos criptográficos assimétricos, mais precisamente *assinaturas digitais* organizadas adequadamente. Esta mesma observação indica que, via de regra, será necessário recorrer a técnicas de criptoanálise para avaliar até que ponto um esquema de marca d'água atinge seus objetivos de projeto – infelizmente, esta abordagem não parece ser adotada em muitos esquemas propostos. Adicionalmente, o caráter intrusivo de qualquer marca d'água requer a minimização do volume de dados embutidos na imagem hospedeira (a fim de não deteriorar a qualidade da imagem resultante) e maximização da velocidade de processamento (devido ao número naturalmente elevado, tipicamente vários milhares, de assinaturas que se devem gerar e verificar em imagens realísticas). Em termos criptográficos, as assinaturas inseridas na imagem hospedeira devem ser o mais possível compactas, e seu processamento deve ser tão eficiente quanto for exeqüível obter. Poucos algoritmos reconhecidamente seguros de assinatura digital compacta existem na atualidade; o método mais promissor, chamado esquema BLS, baseia-se no conceito de *emparelhamento bilinear* em certos grupos elípticos. Essa tecnologia, porém, até recentemente era considerada ineficiente demais para aplicações práticas.

Nossa pesquisa tem por foco a criptoanálise e o projeto seguro de marcas d'água topológicas, bem como a elaboração de algoritmos assimétricos eficientes como substrato criptográfico para essas marcas d'água. Apontamos diversas falhas de segurança em esquemas topológicos propostos; em especial, definimos os conceitos de *ataque de transplante* e de *ataque de aniversário avançado*, aos quais sucumbe a quase totalidade das marcas d'água dessa categoria. Em contrapartida, sugerimos um esquema novo (chamado *encadeamento de blocos de hash*, ou HBC) que resiste não só a esses, mas a todos os ataques de nosso conhecimento. Nos aspectos criptográficos, apresentamos uma variante determinística e eficiente (ordens de grandeza mais rápida que as variantes previamente conhecidas) do *algoritmo de Miller* para o cálculo do *emparelhamento de Tate*, bem como um algoritmo geral para a construção de curvas elípticas e geradores de grupos amigáveis ao cálculo desse emparelhamento. Esta parte da nossa pesquisa tem personalidade própria e interesse independente do contexto em que o empregamos, pois torna prática pela primeira vez uma família inteira de algoritmos criptográficos recentes e extremamente úteis – os sistemas baseados em emparelhamentos, que conseguem resolver elegantemente diversos problemas que permaneceram abertos durante décadas. Por fim, mostramos como os próprios resultados originais aqui expostos sugerem outros problemas de pesquisa, quer na área de segurança de imagens, quer na de criptografia enquanto disciplina independente.

ABSTRACT

Digital watermarks are the method of choice for ensuring the integrity, authenticity, and nonrepudiation of images and other digital signals derived from originally analogic information. In particular, *topological* watermarks are capable of not only detecting, but also localizing alterations in marked images with a previously established resolution. The nature of the goals of such watermarks suggests a close association with asymmetric cryptographical algorithms; more precisely, with suitably organized *digital signatures*. This very observation indicates that it will be usually necessary to resort to cryptanalytical techniques to assess to what extent a watermark scheme achieves its design goals – but unfortunately, this approach does not seem to be adopted in the majority of proposed schemes. Additionally, the intrusive character of any watermark requires minimizing the amount of data embedded in the host image (to avoid deteriorating the quality of the resulting image) and maximizing the processing speed (due to the naturally high number, typically several thousands, of signatures one must generate and verify in realistic images). In cryptographical terms, the signatures inserted in a host image must be as compact as possible, and its processing must be as efficient as feasible. There currently exist very few secure algorithms to obtain compact signatures; the most promising method, the so called BLS scheme, is based upon the concept of *bilinear pairing* on certain elliptic groups. However, until recently this technology was considered too inefficient to be used in practice.

The focus of our research is the cryptanalysis and the secure design of topological watermarking schemes, as well as the elaboration of efficient asymmetric algorithms as a cryptographical framework for such schemes. We point out several security breaches in proposed topological schemes; we especially define the concepts of *transplantation attack* and *advanced birthday attack*, to which nearly all watermarks in that category succumb. As a counterpart, we suggest a new watermarking scheme (called *hash block chaining*, or HBC) that resists not only these, but actually all attacks of which we are aware. On the cryptographical side, we present a deterministic, efficient variant (orders of magnitude faster than previously known variants) of *Miller's algorithm* to compute the *Tate pairing*, as well as a general algorithm to construct pairing-friendly elliptic curves and group generators. This part of our research has a character of its own, and its interest is quite independent from the context in which we apply it, since for the first time it makes practical a whole family of recent and extremely useful cryptographical algorithms – the pairing based cryptosystems, which can elegantly solve several problems that remained open for decades. Finally, we show how our original results by themselves suggest other research problems, both in the area of image security and in the realm of cryptography *per se*.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 APRESENTAÇÃO

1.1 Exposição do cenário

O crescimento espetacular dos sistemas multimídia em rede nos últimos anos, particularmente com o advento da *World Wide Web*, impôs desafios enormes à manutenção de determinados aspectos de imagens digitais, tais como reconhecimento de propriedade para estabelecimento de direitos autorais e *copyright* (por exemplo, para fotografias jornalísticas), integridade, autenticidade e irretratabilidade (como é o caso de imagens legais, médicas e similares).

Para fazer frente a esses desafios foi introduzido o conceito de *marcas d'água digitais*, com o mesmo propósito das marcas materiais análogas há muito aplicadas a documentos impressos.

Uma marca d'água digital, porém, deve satisfazer uma série de requisitos para ser efetivamente útil na resolução dos problemas acima. Entre esses requisitos, enfatizamos um aspecto até pouco tempo raramente discutido em esquemas existentes de marca d'água digital: a possibilidade de detecção/verificação *pública* de uma marca d'água.

Esta observação estabelece, como veremos, uma conexão natural (embora ainda não explorada a fundo) entre técnicas de processamento de sinais e algoritmos criptográficos assimétricos (conhecidos também como algoritmos de *chave pública*). Todavia, a natureza peculiar das marcas d'água exige um cuidado especial na aplicação

de assinaturas digitais para esse propósito. Um esquema de marcas d'água baseado em tal princípio precisa ser projetado com todas as técnicas conhecidas de criptoanálise em mente, sob o risco de conter vulnerabilidades sutis que possam enfraquecer ou invalidar a construção.

1.2 Objetivos

Nosso foco no presente trabalho será analisar e projetar marcas d'água destinadas a proporcionar confirmação pública da integridade, autenticidade e irretratabilidade de imagens digitais e, por extensão, a outros tipos de sinais digitais. Nossos algoritmos terão, além disso, a propriedade de localizar eventuais alterações nas imagens marcadas com resolução razoável.

Nesse contexto, um aspecto central de nosso estudo será o da construção e implementação eficiente de assinaturas digitais compactas, que reduzem o volume de informação de controle necessário para obter as propriedades desejadas das marcas d'água, ou o aumento na granularidade da localização de alterações nas imagens marcadas. Manteremos sempre em mente as técnicas modernas de criptoanálise para evitar vulnerabilidades em nossas construções.

Em contrapartida, não será nosso objetivo um estudo das características puramente perceptuais das marcas d'água geradas. Um tal estudo, que tem interesse independente, é deixado como sugestão para ulteriores pesquisas.

Muitas das técnicas desenvolvidas neste trabalho e outras dele derivadas, notavelmente nossos métodos para implementação eficiente de sistemas baseados em emparelhamentos, transcendem o âmbito de marcas d'água e mostram-se relevantes em muitas áreas de criptografia, conforme atestam as citações à nossa pesquisa (AL-RIYAMI; PATERSON, 2002; AL-RIYAMI; PATERSON, 2003; BALFANZ et al., 2003; BERTONI et al., 2003; BONEH; FRANKLIN, 2003; BOYD; MAO; PATERSON, 2003; BOYEN,

2003; BREZING; WENG, 2003; DEGUILLAUME; VOLOSHYNOVSKIY; PUN, 2002; DUPONT; ENGE; MORAIN, 2002; DUURSMA; LEE, 2003; EISENTRAEGER; LAUTER; MONTGOMERY, 2003; GAGNÉ, 2002; GALBRAITH; HARRISON; SOLDERA, 2002; HARRISON; PAGE; SMART, 2002; HU; WU; IRWIN, 2003; IZU; TAKAGI, 2003; KNUDSEN; WAGNER, 2002; LIN; WU, 2003; SUN; HSIEH, 2003; ZHANG; KIM, 2002; ZHANG; KIM, 2003).

1.3 Contribuições originais

As contribuições originais deste trabalho são as seguintes:

1. [Seções 3.6 e 4.2] Conceituação de *ataque de transplante* e de *ataque de aniversário avançado*, aplicáveis à quase totalidade dos esquemas de marca d'água com capacidade de localizar alterações em imagens. Esses ataques são o resultado de uma série de análises sobre vulnerabilidades comuns a muitos esquemas de marca d'água. Apontamos também a ineficácia de diversas medidas propostas contra esses ataques.

2. [Capítulo 4] Definição de *encadeamento de blocos de hash* (HBC), a primeira família de esquemas de marca d'água resistentes ao ataque de transplante e outros. O encadeamento de blocos de *hash* localiza naturalmente alterações em imagens e outros sinais N-dimensionais. Como bônus, o algoritmo HBC possui uma propriedade de proteção de *hash* que possibilita reduzir o tamanho das assinaturas digitais empregadas (particularmente assinaturas Schnorr) sem degradar o nível de segurança. Variantes avançadas permitem localizar inserções e remoções, e podem beneficiar-se de paralelismo, se disponível.

3. [Seção 8.1] Apresentação de um algoritmo eficiente para a extração de raízes quadradas em certos corpos finitos. Esta operação é fundamental em sistemas criptográficos baseados em curvas elípticas e hiperelípticas. O melhor

algoritmo prático previamente conhecido para esse cálculo num corpo $\mathbb{F}_q$ tem complexidade $O(n^3)$, onde $n = \log q$; nosso novo algoritmo tem complexidade $O(n^2 \log n)$.

4. [Seção 8.2] Conceituação da operação de *triplicação de ponto* e aperfeiçoamento da multiplicação por escalar em curvas supersingulares sobre corpos de característica 3. Estas operações, que estendem resultados conhecidos em característica 2, têm complexidades $O(m)$ e $O(m^2)$, respectivamente. Em hardware, a triplicação de ponto pode ser feita até mesmo em tempo $O(1)$. Por comparação, os melhores resultados previamente conhecidos em característica diferente de 2 têm complexidades $O(m^2)$ e $O(m^3)$, respectivamente. Nossos algoritmos foram parcialmente redescobertos pelo trabalho independente de (GALBRAITH; HARRISON; SOLDERA, 2002), e recentemente estendidos em (DUURSMA; LEE, 2003; HARRISON; PAGE; SMART, 2002; PAGE; SMART, 2003; SMART; WESTWOOD, 2003).

5. [Seção 8.4] Descrição de um método para construir curvas elípticas ordinárias (não-supersingulares) contendo subgrupos com grau de imersão arbitrário. Este método resolve um problema anteriormente aberto, proposto em (BONEH; LYNN; SHACHAM, 2002). Nosso algoritmo generaliza o conceito de curvas MNT (MIYAJI; NAKABAYASHI; TAKANO, 2001), permitindo além disso a escolha de certos parâmetros com propriedades favoráveis à otimização (por exemplo, ordens primas com representação binária esparsa).

6. [Seção 8.5] Elaboração de um algoritmo simples e eficiente para *selecionar geradores* de grupos amigáveis a emparelhamentos. Como bônus, nosso algoritmo torna mais eficientes várias operações independentes de emparelhamentos como geração de pares de chaves, e possibilita reduzir a utilização de banda ocupada por chaves e assinaturas.

7. [Seção 8.6] Aperfeiçoamento do algoritmo de Miller para calcular o emparelhamento de Tate. A variante aqui proposta é determinística, e evita muitas operações irrelevantes presentes no algoritmo convencional. Aplicado a curvas supersingulares em características 2 e 3, a nova variante reduz a contribuição da multiplicação escalar subjacente e da exponenciação final à complexidade computacional de $O(m^3)$ para $O(m^2)$. Propomos neste contexto a técnica de *eliminação de denominadores*, aplicável não só a essas curvas supersingulares mas também aos grupos construídos segundo o método de seleção mencionado no item anterior.

Alguns destes resultados encontram-se agora publicados (BARRETO; KIM, 1999; BARRETO et al., 2002; BARRETO; LYNN; SCOTT, 2002; BARRETO; KIM; RIJMEN, 2000; BARRETO; KIM; RIJMEN, 2001; BARRETO; KIM; RIJMEN, 2002; BARRETO; LYNN; SCOTT, 2003b) ou foram submetidos para publicação (BARRETO; KIM, 2001; BARRETO; LYNN; SCOTT, 2003a). O mesmo vale para certos resultados paralelos de nossas pesquisas, nas áreas de processamento de imagens (KIM; BARRETO, 2000), criptoanálise (BARRETO et al., 2002; NAKAHARA-JR. et al., 2002) e algoritmos criptográficos propostos (BARRETO; RIJMEN, 2000a; BARRETO; RIJMEN, 2000b; BARRETO; RIJMEN, 2000c).

1.4 Organização

Esta tese está organizada em duas partes, para não comprometer uma leitura orientada quer aos aspectos relacionados a marcas d'água, quer às contribuições de natureza estritamente criptográfica.

Assim, a primeira parte concentra-se especificamente na construção e segurança de marcas d'água, e comporta os capítulos 2 a 4, a saber:

- O capítulo 2 conceitua e classifica imagens e marcas d'água, levantando os pro-

blemas associados e enumerando algumas tentativas de solução.

- O capítulo 3 relaciona marcas d'água e assinaturas digitais, e aponta vulnerabilidades em combinações existentes desses conceitos.

- O capítulo 4 explora nossa nova construção (denominada HBC, ou *hash block chaining*) de marca d'água usando assinaturas digitais, expondo suas propriedades de segurança e sua eficiência.

A segunda parte focaliza os algoritmos criptográficos subjacentes, cujo interesse e aplicabilidade transcendem ao âmbito das marcas d'água. Esta parte abrange os capítulos 5 a 8.

- O capítulo 5 expõe a teoria matemática em que se fundamentam as técnicas aqui desenvolvidas.

- O capítulo 6 introduz elementos básicos de criptografia assimétrica e assinaturas digitais.

- O capítulo 7 elabora em maior profundidade o conceito de assinaturas digitais compactas e técnicas para obtê-las.

- O capítulo 8 traz nossos novos algoritmos para a implementação eficiente de assinaturas digitais úteis para a construção de marcas d'água, discutindo aspectos de implementação e resultados experimentais.

Finalmente, o capítulo 9 resume e conclui nosso trabalho, e o apêndice A enumera as publicações relevantes do autor.

PARTE I

IMAGENS E MARCAS D’ÁGUA

2 IMAGENS E SEU PROCESSAMENTO

Neste capítulo, recolhemos conceitos formais sobre imagens, seu processamento, e generalidades sobre marcas d'água, que desenvolveremos adiante.

2.1 Noções básicas

Definição 1. *Uma* imagem *de dimensões* $m \times n$*,* c *cores e* k *níveis é uma função* $A : \mathbb{Z}_m \times \mathbb{Z}_n \to \mathbb{Z}_{2^k}^c$.

O número c de cores é geralmente igual a três, mas em certos casos pode ser substancialmente expandido pela inclusão de faixas não visuais do espectro eletromagnético, como ocorre em imagens meteorológicas e astronômicas (BINNEY; MERRIFIELD, 1998, seção 2.3).

Às vezes é conveniente, por motivos de simplicidade algorítmica, considerar extensões de uma imagem A a todo o plano $\mathbb{Z} \times \mathbb{Z}$. Duas extensões apresentam-se naturalmente em determinados contextos. A mais simples é a *extensão com suporte*, em que a imagem A é vista como uma região retangular num plano incolor infinito (chamado suporte de A). A *extensão periódica* replica o conteúdo de A periodicamente sobre todo o plano.

Definição 2. *Seja* $b \in \mathbb{Z}_{2^k}$*. A* extensão com suporte *de uma imagem* $A : \mathbb{Z}_m \times \mathbb{Z}_n \to$

$\mathbb{Z}_{2^k}^c$ *com* cor de fundo b *é a imagem* $A^{sup} : \mathbb{Z}_m \times \mathbb{Z}_n \to \mathbb{Z}_{2^k}^c$ *tal que*

$$A^{sup}(i,j) \equiv \begin{cases} A(i,j), & \textit{se } i \in \mathbb{Z}_m \textit{ e } j \in \mathbb{Z}_n; \\ (b,b,\ldots,b) \in \mathbb{Z}_{2^k}^c, & \textit{caso contrário}. \end{cases}$$

Definição 3. *A* extensão periódica *de uma imagem* $A : \mathbb{Z}_m \times \mathbb{Z}_n \to \mathbb{Z}_{2^k}^c$ *é a imagem* $A^{per} : \mathbb{Z}_m \times \mathbb{Z}_n \to \mathbb{Z}_{2^k}^c$ *tal que*

$$A^{sup}(i,j) \equiv A(i \bmod m, j \bmod n).$$

Também é conveniente considerar regiões retangulares ou *blocos* de uma imagem, isto é, restrições da função A:

Definição 4. *Um* bloco *de uma imagem* $A : \mathbb{Z}_m \times \mathbb{Z}_n \to \mathbb{Z}_{2^k}^c$ *é uma restrição* $X : \{i_1, i_1+1, \ldots, i_2\} \times \{j_1, j_1+1, \ldots, j_2\} \to \mathbb{Z}_{2^k}^c$ *da função* A*, de modo que* $X(i,j) = A(i,j)$ *para* $0 \leqslant i_1 \leqslant i \leqslant i_2 < m$ *e* $0 \leqslant j_1 \leqslant j \leqslant j_2 < n$*.*

Onde necessário, os limites do domínio de X serão concisamente indicados por $X[i_1 : i_2, j_1 : j_2]$.

O valor $A(i,j)$ é chamado *pixel* de A na posição (i,j). Se $c = 1$, A é chamada imagem em *níveis de cinza*, ou monocromática; se, além disso, $k = 1$, a imagem é dita *binária*.

Uma imagem pode ser representada como uma seqüência $A = (A_1, A_2, \ldots, A_c)$, onde cada A_p é uma imagem monocromática de dimensões $m \times n$ e k níveis, chamada um *plano de cor*. Os pixels de A_p são chamados componentes (da p-ésima cor) dos pixels de A.

A *diferença* entre duas imagens A e B com as mesmas dimensões e mesmo número de níveis é a imagem $C \equiv A \oplus B$ tal que $C_p(i,j) = A_p(i,j) \oplus B_p(i,j)$, para todos os valores permitidos de i, j e p.

Dada uma imagem A, denota-se por A^* a imagem obtida pela anulação do bit

menos significativo de cada pixel em todos os planos de cor de A. Mais precisamente, $A^* \equiv (A_1^*, A_2^*, \ldots, A_c^*)$ onde

$$A_p^*(i,j) = \begin{cases} A_p(i,j) & \text{se } A_p(i,j) \text{ é par,} \\ A_p(i,j) \oplus 1 & \text{caso contrário.} \end{cases}$$

A seguinte definição caracteriza a representação primária de marcas d'água que consideraremos adiante.

Definição 5. *O* nível d'água *de uma imagem A é a imagem $A^o \equiv A \oplus A^*$.*

Note-se que todos os planos de cor de A^o são imagens binárias.

Sejam A e B imagens com as mesmas dimensões $m \times n$ e número de cores c, onde todos os planos de cor de B são binários, isto é, $B = B^o$. A *inserção* de B no nível d'água de A é a imagem $C \equiv A^* \oplus B$.

É possível considerar também uma representação de imagens no domínio da freqüência (séries de Fourier e similares), mas por simplicidade — e sem perda substancial de generalidade, por não se tratar de nosso foco de interesse neste trabalho — não nos deteremos mais detalhadamente nessa possibilidade.

2.2 Conceituação de marca d'água.

Definição 6. *Uma* marca d'água digital *é um sinal portador de informação, visualmente imperceptível, embutido numa imagem digital. A imagem que contém a marca d'água é dita* imagem marcada *ou* hospedeira.

A informação contida numa marca d'água é de natureza e quantidade variada, desde um dado elementar como sua mera presença até semânticas complexas como logotipos ou legendas.

Idealmente, uma marca d'água deveria satisfazer as seguintes propriedades:

- Armazenada na própria imagem;
- Visualmente imperceptível quando inserida;
- Visualmente significativa quando extraída;
- Irreproduzível por terceiros não autorizados;
- Capaz de localizar alterações maliciosas na imagem hospedeira com resolução suficiente;
- Publicamente verificável;
- Indelével por manipulação não autorizada;
- Resistente a certas operações de processamento de imagens (como mudar o nível de compressão);
- Aplicável a formatos com e sem perdas, e a imagens binárias, em níveis de cinza e coloridas;
- Eficiente em tempo de processamento e espaço de armazenamento.

Obter todas estas características é muito difícil na prática. A maioria dos algoritmos de marca d'água concentra-se apenas em alguns poucos destes requisitos, e portanto são aplicáveis numa gama restrita de circunstâncias.

Outra dificuldade reside no fato que os requisitos acima podem ser contraditórios, no sentido de que algumas operações em imagens são ora exigidas, ora proibidas.

Por exemplo, é desejável que se possa verificar uma marca d'água depois de uma operação de truncamento (*cropping*) numa disputa judicial de autoria ou propriedade legal de uma imagem (i.e. deve-se ser capaz de mostrar que a marca d'água ainda está na imagem, e determinar inequivocamente o seu autor ou proprietário); entretanto, uma verificação de marca d'água deve falhar (indicando adulteração no conteúdo da

imagem) após o truncamento quando o problema em consideração é o de integridade exclusivamente (como em imagens legais ou médicas).

De fato, o problema da resolução de propriedade e o problema da verificação de integridade e autenticidade são complementares (MACQ; QUISQUATER, 1995), e freqüentemente exigem que técnicas dedicadas sejam aplicadas em sua solução, o primeiro aparecendo no contexto de codificação e representação de informações, e o segundo estando mais diretamente relacionado com técnicas criptográficas.

2.2.1 Marcas d'água robustas e frágeis

Esquemas para a obtenção de marcas d'água classificam-se entre *robustos* e *frágeis* segundo a dificuldade em remover a marca de uma imagem marcada.

Assim, uma marca d'água diz-se robusta se sua remoção de uma imagem marcada deteriora a qualidade da imagem resultante a ponto de destruir seu conteúdo visual. Mais precisamente, a correlação entre uma imagem marcada e uma marca d'água robusta nela inserida permanecerá detectável mesmo após um processamento digital, enquanto a imagem resultante do processamento continuar visualmente reconhecível e identificável com a imagem original. Por esse motivo, marcas d'água robustas são normalmente utilizadas para a verificação de propriedade ou de *copyright* de imagens. Há, porém um resultado negativo que parece indicar a impossibilidade de criar marcas d'água robustas realmente seguras (BARAK et al., 2001).

Uma marca d'água frágil, por outro lado, pode ser removida sem afetar substancialmente o aspecto visual da imagem resultante. Contudo, é possível construir marcas d'água frágeis cuja remoção sempre pode ser detectada. Esta propriedade torna tais marcas d'água úteis para fins de autenticação e atestação de integridade de imagens. Em outras palavras, uma marca d'água frágil fornece uma garantia de que a imagem marcada não seja despercebidamente editada ou adulterada, e também de que efetivamente provém da origem declarada ou assumida. Neste sentido, o termo "frágil" é

infeliz para qualificar esses algoritmos, sendo mantido apenas por motivos históricos.

Exemplos de esquemas frágeis propostos são (FRIDRICH; GOLJAN; BALDOZA, 2000; WONG, 1998; YEUNG; MINTZER, 1997), para nomear apenas alguns.

2.2.2 Detecção privada e pública de marcas d'água

Outro critério de classificação de marcas d'água refere-se à transparência do processo de detecção das marcas, que pode ser *privado* ou *público*. Esta distinção é importante pelas suas implicações de segurança: a capacidade de verificar marcas d'água inerentemente privadas está via de regra associada com a capacidade de produzir outras marcas d'água em nome da mesma entidade.

Naturalmente, marcas d'água publicamente detectáveis têm maior aplicabilidade que marcas d'água exclusivamente privadas. Em contrapartida, marcas d'água privadas são realizáveis com algoritmos notoriamente mais eficientes que marcas d'água públicas, sendo portanto bastante convenientes em determinadas situações.

A necessidade de verificação *pública* de marcas d'água é óbvia. Alegações de autoria ou propriedade de imagens podem ser, nesse cenário, aceitas ou rejeitadas sem que o autor ou proprietário precise revelar informação de caráter privativo. Este comportamento algorítmico é típico de — e facilmente implementável por — sistemas criptográficos assimétricos, como foi pioneiramente indicado em (FRIEDMAN, 1993).

Todavia, muitos esquemas de marca d'água explicitamente ignoram ou excluem essa possibilidade. No método proposto em (CHAE; MUKHERJEE; MANJUNATH, 1998), "a assinatura é recuperável pelo proprietário, que possui a chave para decodificar os dados ocultos [na imagem]" ("*the signature is only recoverable by the owner who has the key to decoding the hidden data*"). O algoritmo descrito em (WU; LIU, 1998) emprega "uma tabela de consulta proprietária" ("*a proprietary look-up table*") que é usada tanto para inserir quanto para extrair (i.e. verificar) marcas d'água. Um

método relacionado é exposto em (YEUNG; MINTZER, 1997), que apresenta vulnerabilidades intrínsecas (FRIDRICH; GOLJAN; MEMON, 2000). As técnicas analisadas em (KALKER; LINNARTZ; DIJK, 1998) são tais que "a detecção de marcas d'água eletrônicas só é exeqüível se o detetor de marcas d'água conhecer [um parâmetro secreto] K" (*"electronic watermark detection is only feasible if the watermark detector is aware of K"*). Em (MARVEL; RETTER; BONCELET-JR., 1998) um método de esteganografia simétrica (chave secreta) é aplicado à geração de marcas d'água. Um trabalho muito interessante é o de (PIVA et al., 1997), que descreve um método de geração de marcas d'água robustas contra a maioria das técnicas de processamento de sinais e distorções geométricas; contudo, esse método não é publicamente verificável.

Poucos esquemas efetivamente levam em consideração a necessidade de verificação pública. A técnica de (HARTUNG; GIROD, 1997) é uma delas (e uma bastante notável), mas mesmo esse é um esquema *a priori*, já que "a idéia subjacente é tornar pública apenas parte da chave pseudo-aleatória" (*"the underlying idea is to make only parts of the pseudo-noise key public"*) em vez de se adotar diretamente criptografia de chave pública (neste contexto, o uso da expressão "public key" pelos autores não é padrão).

Um exemplo de algoritmo de marcas d'água realmente baseado em criptografia assimétrica é o de (BHATTACHARJEE; KUTTER, 1998), mas lá a assinatura digital da imagem é "armazenada numa base de dados ou acrescentada ao cabeçalho da imagem" (*"stored in a database or added to the image header"*), e portanto não satisfaz o requisito de estar embutida na própria imagem hospedeira.

2.2.3 O problema da localização – marcas d'água topológicas

A possibilidade de não apenas detectar que uma imagem foi indevidamente alterada, mas também identificar que região sofreu alterações é uma propriedade desejável de marcas d'água frágeis, porquanto permitiria decidir se a alteração está localizada ou

não numa região importante dos dados.

Um exemplo típico desta situação ocorre em bases de dados contendo imagens médicas ou fotografias de sinistros automobilísticos, onde pode ser elucidativo para rastrear as intenções do falsário saber que parte de uma imagem foi alterada (cfr. figura 1).

A escala de resolução na localização de alterações varia desde pixels individuais (discretização total) até a imagem inteira (mera detecção de manipulação). Por razões algorítmicas ou de eficiência, a abordagem mais comum para esse aspecto da resolução é *topológica*, no sentido de que uma imagem a ser marcada é normalmente particionada em blocos, e cada bloco recebe um fragmento próprio da marca d'água; o esquema será idealmente capaz de identificar que blocos individuais foram alterados, embora não consiga apontar quais pixels desses blocos foram manipulados.

Tendo isso em mente adotaremos por concisão de referência a seguinte definição:

Definição 7. *Chama-se* topológico *qualquer esquema de marca d'água frágil por blocos.*

2.2.4 Representação de marcas d'água frágeis

A representação de marcas d'água frágeis em imagens com perdas, tais como imagens JPEG (WALLACE, 1992), ainda é, em linhas gerais, um problema em aberto. Como já indicamos no início da seção 2.2, explica-se isto pelos requisitos mais ou menos antagônicos de detectar quaisquer alterações na imagem, *exceto* mudanças no nível de compressão, que via de regra envolvem modificações em todos os pixels. Técnicas eficazes para a seleção automática de características relevantes têm-se mostrado pouco satisfatórias, constituindo ainda um problema em aberto. Uma pesquisa interessante nesse sentido é a de Choi e Aizawa (CHOI; AIZAWA, 2001), que, contudo, focaliza-se na manutenção de *copyright* de imagens, não abordando o problema de localizar

Figura 1: Tentativa (frustrada) de adulteração de imagem legal, hospedeira de marca d’água.

alterações.

Não nos propomos sugerir aqui soluções para este problema em particular. Destarte, restringir-nos-emos à representação original de Wong para imagens sem perdas, a saber, à técnica de inserir a marca d’água no nível d’água de uma imagem (cfr. definição 5), adequada para imagens em níveis de cinza e coloridas.

3 MARCAS D'ÁGUA E CRIPTOGRAFIA

Por uniformidade de exposição, assumimos neste capítulo alguma familiaridade com os conceitos criptográficos de função de *hash*, oráculos aleatórios, criptografia assimétrica e assinatura digital. Em benefício do leitor, porém, definições formais apropriadas para esses conceitos são dadas nas seções 6.1, 6.2, 6.3 e 6.4, respectivamente.

3.1 Construções topológicas baseadas em assinaturas digitais

Assinaturas digitais convencionais são capazes de *detectar* alterações em dados assinados, mas não de *localizá-las*, o que seria uma propriedade desejável, conforme discutimos na seção 2.2.3.

Abordagens ingênuas a este problema – como particionar os dados em blocos e assinar cada bloco individualmente (WONG, 1998) – podem carecer da própria capacidade de meramente detectar alterações, como ocorre, por exemplo, se os blocos forem simplesmente rearranjados. Até algumas técnicas de fornecer informação posicional apresentam essa vulnerabilidade; por exemplo, alimentar as coordenadas dos blocos à função de *hash* impossibilita ataques de rearranjo conforme descritos acima, porém falha na detecção da troca de um bloco por outro com coordenadas e conteúdo idênticos, mas proveniente de outro conjunto de dados legitimamente assinados, particionados de maneira semelhante.

Um problema associado à capacidade de localização de alterações é o *trade-off* entre o volume de informação inserido na imagem hospedeira e a granularidade dos blocos assinados. Um bom esquema de marca d'água topológica procurará minimizar a quantidade de informações inseridas em cada bloco, possibilitando a adoção de blocos menores e a conseqüente maximização da capacidade de localização de alterações na imagem.

Começaremos nosso estudo de construções topológicas com o primeiro algoritmo qualificável como topológico *público* (isto é, projetado para permitir a verificação pública da marca d'água topológica).

3.2 O algoritmo de Wong

Wong (WONG, 1998) é um dos pioneiros da utilização de técnicas de assinatura digital para a obtenção de marcas d'água.

O método de Wong para inserção de marcas d'água numa imagem em níveis de cinza pode ser sumarizado como segue.

1. Seja Z uma imagem de dimensões $M \times N$ na qual se deseja inserir uma marca d'água. Particiona-se Z em n blocos $Z_t (0 \leqslant t < n)$ de tamanho 8×8 pixels (no máximo; blocos das bordas podem ser menores). Cada bloco Z_t será marcado separadamente.

2. Seja A uma imagem binária visualmente significativa a ser usada como base da marca d'água (por exemplo, um logotipo). Esta imagem é replicada periodicamente para obter uma imagem grande o bastante para cobrir Z inteiramente. Para cada bloco Z_t haverá um bloco binário correspondente A_t.

3. Usando uma função de *hash* criptograficamente segura H, calcula-se o código de integridade $H_t \equiv H(M, N, Z_t^*)$.

4. Calcula-se o ou-exclusivo de H_t com o bloco A_t, obtendo-se o código marcado $\hat{H}_t$.

5. Encripta-se $\hat{H}_t$ com a chave privada k, gerando assim a assinatura digital $S_t \equiv E_k(\hat{H}_t)$.

6. Insere-se S_t nos bits menos significativos dos pixels de Z_t^*, obtendo-se o bloco marcado $X_t = Z_t^* \oplus S_t$.

O algoritmo correspondente para a verificação da marca d'água é imediato:

1. Seja X uma imagem marcada de dimensões $M \times N$. Particiona-se esta imagem em n blocos X_t, como antes.

2. Usando a mesma função de *hash* H escolhida para a inserção da marca d'água, calcula-se o índice de integridade $H_t \equiv H(M, N, X_t^*)$.

3. Extrai-se o conteúdo do nível d'água X^o de X_t e decifra-se o resultado usando a chave pública do suposto signatário (o originador da mensagem), obtendo-se o bloco decifrado D_t.

4. Calcula-se o ou-exclusivo de H_t com o bloco D_t, obtendo-se o bloco de verificação C_t.

5. Se C_t e A_t forem iguais, declara-se genuína a imagem marcada. Caso contrário, declara-se que a imagem marcada X foi alterada no bloco X_t.

3.3 Ataques simples

Indicaremos agora algumas vulnerabilidades criptoanalíticas do método de Wong, e sugeriremos adiante meios de torná-lo robusto. Não é demais lembrar que um esquema de autenticação só é realmente seguro se *qualquer* alteração na imagem marcada for detectável, mesmo que essas mudanças não pareçam imediatamente aprovei-

Figura 2: Permutação de planos de cor (RGB → GBR)

táveis para propósitos maliciosos. A mera existência de alguma falha notável indica uma vulnerabilidade no esquema que não foi levada em consideração durante o projeto, e que pode ser usada em ataques futuros[1].

Por exemplo, técnicas de marca d'água para imagens em níveis de cinza são comumente generalizadas para imagens coloridas pela simples aplicação dessas técnicas aos planos de cor independentemente (o próprio algoritmo de Wong é um exemplo). Neste caso, a marca d'água não detectará permutações dos planos de cor. Embora possa ser difícil imaginar como este ataque possa ser usado maliciosamente (exceto em casos de vandalismo), é mais prudente que mesmo esta categoria de alteração não passe despercebida. Este problema concreto, ilustrado na figura 2, pode ser facilmente remediado submetendo todos os planos de cor juntos e em ordem fixa à função de *hash*.

[1]Exemplos abundantes existem de anomalias detectadas em algoritmos criptográficos que, sem constituir vulnerabilidades em si, foram posteriormente convertidas em ataques efetivos: criptoanálise linear a partir da linearidade peculiar de uma das S-boxes do DES, curvas elípticas anômalas onde o logaritmo discreto (seção 6.3.2) é computável em tempo polinomial, colisões na função de compressão do MD4 levando a colisões da função completa e muitos outros cenários

Figura 3: Ataque de mosaico com blocos da mesma imagem.

3.3.1 Ataque de mosaico

Há outro ataque simples e indetectável pelo esquema de marca d'água de Wong, mas que pode ser realmente usado com intenções maliciosas; a ele damos o nome *ataque de mosaico* ou *cut-and-paste*, e baseia-se na vulnerabilidade inerente à independência das assinaturas dos blocos. Suponhamos que um falsário possua uma coleção de imagens legitimamente marcadas, todas do mesmo tamanho e contendo a mesma imagem A embutida na marca d'água. Dado que cada bloco é marcado separadamente sem qualquer informação adicional sobre a imagem hospedeira exceto suas dimensões, é possível para esse falsário selecionar blocos das imagens autênticas e construir com eles uma nova imagem cuja marca d'água será erroneamente verificada pelo algoritmo de Wong, e aceita como legítima. Assumimos aqui que as coordenadas originais de cada bloco são mantidas inalteradas na imagem forjada. Contudo, em alguns casos (por exemplo, se o tamanho da imagem A for da forma $2^r \times 2^s$) pode ser possível montar um ataque de mosaico dentro de uma mesma imagem marcada, mantendo a marca d'água lá inserida intacta. A figura 3 ilustra este tipo de ataque. O ataque de mosaico foi descoberto independentemente de nossa pesquisa por Holliman e Memon (HOLLIMAN; MEMON, 2000).

3.4 O paradoxo do aniversário

Antes de analisarmos os próximos ataques contra marcas d'água topológicas, faremos uma breve digressão sobre certos comportamentos estatísticos não intuitivos de funções aleatórias. Os ataques abordados adiante serão baseados nesses comportamentos.

Seja f uma função sobrejetora cujo contra-domínio consista em n valores distintos. Suponhamos que uma seqüência $(f_1, f_2, \ldots, f_k)$ de valores retornados por f seja aleatória e uniformemente distribuída. A probabilidade de encontrar nessa seqüência dois valores iguais (isto é, dois índices $i \neq j$ tais que $f_i = f_j$) torna-se maior que $1/2$ quando $O(\sqrt{n})$ valores estiverem presentes; mais precisamente, quando $k > c\sqrt{n}$, onde $c = \sqrt{2 \ln 2}$. Um tal par de valores é chamado *colisão* de f.

Este fenômeno puramente estocástico é chamado *paradoxo do aniversário* (STINSON, 2002, seção 4.2.2) (MENEZES; OORSCHOT; VANSTONE, 1999, seção 9.7.1), e independe dos detalhes de f. A importância do paradoxo do aniversário está em que mesmo função computacionalmente unidirecional (isto é, uma função de *hash* criptograficamente segura) produzirá colisões espontaneamente quando chamada um número apropriado de vezes. Sempre que se precise obter com alta probabilidade uma seqüência de valores sem repetição, será necessário usar uma função cujo contra-domínio seja grande o suficiente.

Uma generalização do paradoxo do aniversário é considerada por Nishimura e Sibuya (NISHIMURA; SIBUYA, 1990). Seja f uma função sobrejetora cujo contra-domínio consista em n valores distintos. Suponhamos que duas seqüências $(f_1, f_2, \ldots, f_r)$ e $(f_1, f_2, \ldots, f_s)$ de valores retornados por f, com r e s elementos respectivamente, sejam aleatória e uniformemente distribuídas. O número esperado de valores coincidentes entre eles é $c \approx rs/n$.

3.5 Ataques de aniversário

Ataques de aniversário constituem um meio requintado e poderoso de subverter assinaturas digitais. Esses ataques apóiam-se diretamente no paradoxo do aniversário (ou a generalização de Nishimura e Sibuya), sendo portanto independentes da estrutura detalhada dos algoritmos de assinatura. Qualquer função de *hash* que assuma n valores possíveis é suscetível a um ataque de aniversário que encontra colisões (isto é, pares de mensagens cujos índices de integridade sejam iguais, produzindo assinaturas equivalentes) com complexidade $O(n^{1/2})$, o que deve ser comparado com os $O(n)$ passos necessários a uma abordagem exaustiva.

O esquema original de Wong utiliza uma função de *hash* com tamanho não superior a 64 bits; portanto, espera-se encontrar colisões quando o adversário tiver coletado meramente cerca de 2^{32} blocos. Um cenário plausível é uma companhia de seguros que mantém uma base de dados de sinistros automobilísticos. Uma base de dados típica de uma companhia de grande porte pode conter mais de um milhão de imagens de, digamos, 640×480 pixels, de modo que cada imagem seria, com o esquema de Wong, particionada em 4800 blocos individualmente assinados de 8×8 pixels. Isto resulta em mais de 2^{32} assinaturas, o suficiente para montar um ataque simples de aniversário.

A mecânica dos ataques de aniversário baseia-se na seguinte observação. Seja H uma função de *hash* que assume n valores distintos. Dados dois conjuntos de valores de H gerados aleatoriamente, com r e s elementos respectivamente, o número esperado de valores coincidentes entre eles é $c \approx rs/n$ segundo a análise de Nishimura e Sibuya. Portanto, um ataque bem sucedido contra um bloco individual B_t pode ser montado obtendo-se uma base de dados com $r \approx n^{1/2}$ assinaturas válidas e gerando-se uma coleção de $s \approx n^{1/2}$ blocos forjados (que na prática seriam variantes de B_t ligeiramente diferentes entre si, mas visualmente equivalentes). Já que $c \approx 1$ para esses conjuntos, uma assinatura válida será provavelmente encontrada (com probabilidade

superior a 1/2) para algum bloco forjado B'_t, que pode então substituir B_t. Para construir s variantes de B_t, escolhem-se $k \approx \log_2 s$ pixels e varia-se um bit de cada um (preferencialmente um bit pouco significativo, mas obviamente não o menos significativo se a marca d'água for aí armazenada). Se este processo for repetido um número suficiente de vezes, uma imagem inteira pode ser forjada.

Em geral, a única proteção contra ataques de aniversário é aumentar o tamanho dos índices de integridade (ou seja, indiretamente aumentar o valor de n). Isto, porém, reduz a resolução na localização de alterações na imagem, porque os blocos precisam ser aumentados para comportar o volume maior de dados.

O esquema de Wong e Memon (WONG; MEMON, 2001) é outro exemplo de marca d'água suscetível a ataques simples de niversário. Usando a nossa notação, a fórmula de contextos de Wong e Memon é a seguinte:

$$H_t \equiv H(M, N, t, ID, X_t^*),$$

onde ID é um identificador globalmente único da imagem marcada, conhecido por todas as partes envolvidas. Infelizmente, não há dependência de conteúdo entre quaisquer blocos da imagem, abrindo caminho para o ataque de aniversário simples.

3.6 O ataque geral do transplante

Descreveremos agora um ataque novo, de nossa autoria.

Qualquer técnica de particionamento que acrescenta aos argumentos da função de *hash* um contexto local à imagem e determinístico é suscetível ao que chamamos um *ataque de transplante*. Para ver por que isto acontece, denotemos por $X \rightarrow Y$ a circunstância em que o contexto de *hash* usado para o subconjunto Y de um conjunto de dados depende apenas do conteúdo e das coordenadas do subconjunto X do mesmo conjunto. Suponhamos que um adversário obtenha dois conjuntos de dados, S e T,

juntamente com as assinaturas correspondentes, estando cada conjunto particionado em quatro regiões segundo se mostra abaixo:

$$A_S \rightarrow U_S \rightarrow B_S \rightarrow C_S,$$
$$A_T \rightarrow V_T \rightarrow B_T \rightarrow C_T,$$

onde os conteúdos e coordenadas dos blocos na região A_S são idênticos aos da região A_T, e o mesmo vale para B_S e B_T e para C_S e C_T, mas *não* para U_S e V_T. Então o par de regiões (U_S, B_S) é intercambiável com o par (V_T, B_T):

$$A_S \rightarrow V_T \rightarrow B_T \rightarrow C_S,$$
$$A_T \rightarrow U_S \rightarrow B_S \rightarrow C_T,$$

assumindo que as assinaturas correspondentes são igualmente permutadas. Uma vez que esta operação não afeta o contexto de nenhuma região, os índices de integridade são mantidos constantes, evitando a detecção das mudanças efetuadas.

Um exemplo de marca d'água suscetível a este ataque é o esquema Li-Lou-Chen (LI; LOU; CHEN, 2000). Este esquema quebra cada bloco em duas metades, e assina a metade direita de cada bloco concatenada com a metade esquerda do bloco seguinte, percorrendo a imagem em zigue-zague ciclicamente (a metade direita do último bloco é combinada com a metade esquerda do primeiro bloco). Como no algoritmo de Wong, a marca d'água é aqui armazenada no nível d'água da imagem hospedeira. Usando nossa notação, o algoritmo Li-Lou-Chen baseia-se essencialmente na seguinte

fórmula de contextos para o cálculo de *hash* dos blocos da imagem:

$$H_t \equiv H(M, N, \text{right}(X_t^*), \text{left}(X_{(t+1) \bmod MN}^*)).$$

A existência do ataque de transplante mostra ser incorreta a afirmação de que esse esquema detecta quaisquer manipulações locais nas imagens (na verdade, esse esquema é suscetível a ataques mais simples, por exemplo, a simples permutação cíclica dos blocos ao longo da trilha em zigue-zague, ou a remoção de blocos ao longo dessa trilha se dois blocos idênticos ocorrerem na imagem, isto é, se $X_i^* = X_j^*$, todos os blocos $X_i^* \ldots X_{j-1}^*$ ao longo do zigue-zague podem ser removidos sem possibilidade de detecção). Por outro lado, os autores desse esquema têm o mérito de reconhecer que a necessidade de inserir assinaturas RSA (RIVEST; SHAMIR; ADLEMAN, 1978) impõe limites mínimos ao tamanho dos blocos, com uma conseqüente perda na capacidade de resolução de alterações na imagem hospedeira.

4 ENCADEAMENTO DE BLOCOS DE HASH

Apresentaremos agora nossas propostas de marca d'água topológica. A primeira versão, HBC1, foi elaborada antes de nossa descoberta do ataque de transplante (seção 3.6), além de sucumbir a um ataque avançado baseado no paradoxo do aniversário (seção 4.2). A segunda versão, HBC2, foi desenvolvida especificamente com esses ataques em mente, e constitui o primeiro esquema topológico resistente a todos os tipos de ataque do nosso conhecimento.

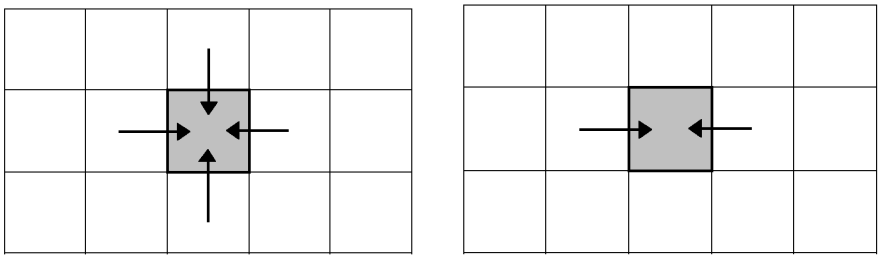
Assumiremos nas duas versões que as funções de *hash* adotadas são aproximações aceitáveis de oráculos aleatórios (seção 6.2), no sentido de que sua estrutura interna não seja aproveitável em ataques, a despeito da organização dos vários argumentos alimentados a essas funções.

4.1 *Hash Block Chaining*, versão 1 (HBC1)

Conforme indicado em (BARRETO; KIM, 1999; BARRETO; KIM; RIJMEN, 2000; BARRETO; KIM; RIJMEN, 2001; HOLLIMAN; MEMON, 2000), a solução para evitar muitos ataques simples é introduzir informação contextual na marca d'água, isto é, no cálculo do índice de integridade H_t (usado para gerar a assinatura do bloco X_t), alimenta-se a função de *hash* H com o conteúdo de blocos vizinhos, além do conteúdo do próprio bloco t. A figura 4 ilustra esta idéia.

Se um bloco X_t for alterado, a verificação de assinatura falhará para esse bloco e todos os blocos que dependam do conteúdo de X_t. Por isso, um número de dependên-

Figura 4: Uso de informação contextual em marcas d'água. Para calcular a assinatura do bloco mostrado em cinza, leva-se em conta o conteúdo desse bloco e de certos blocos vizinhos, conforme o número desejado de dependências (neste exemplo, 4 ou 2 dependências por bloco, respectivamente).



cias tão pequeno quanto possível é desejável para manter uma localização precisa das alterações numa imagem; idealmente, uma única dependência por bloco.

Nossa construção inicial, o modo HBC1, baseia-se na seguinte fórmula de contextos aplicada ao esquema básico de Wong:

$$H_t \equiv H(M, N, t, X^*_{(t-1) \bmod MN}, X^*_t).$$

Como no esquema de Wong, as dimensões M e N da imagem são usadas para detectar truncamento (*cropping*) da imagem. O índice de bloco t é inserido para detectar permutação cíclica de blocos.

Com HBC1, um ataque de mosaico simples não pode mais ser perpetrado, porque se um bloco espúrio substituir o bloco X_t, com probabilidade extremamente alta essa alteração induzirá uma mudança em H_{t+1} (a probabilidade de que essa mudança *não* ocorra é apenas $O(2^{-m})$ para funções de *hash* de m bits). Uma mudança assim invalida a assinatura do bloco X_{t+1}. Similarmente, se um ataque de aniversário for tentado contra o bloco X_t, o suposto falsário teria que forjar não só a assinatura de X_t, mas também a de X_{t+1} devido ao efeito da alteração de X_t sobre H_{t+1}. Mas a mudança obrigatória de X_{t+1} induziria uma mudança em H_{t+2}, e assim sucessivamente. Portanto, o falsário enfrentaria uma propagação cíclica do problema sobre todos os blocos, que finalmente destruiria a assinatura forjada do primeiro bloco adulterado.

Figura 5: Diagrama de dependência contextual para o modo HBC1. Este modo usa uma única dependência por bloco, em varredura zigue-zague ou *raster*.

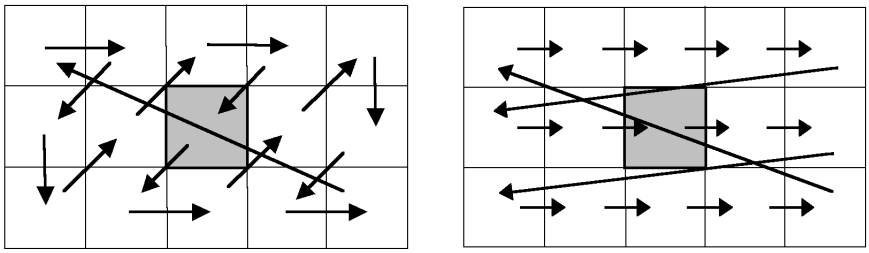


Figura 6: Localização efetiva de alterações e fronteira da região alterada de acordo com o modo HBC1 em varredura *raster*.

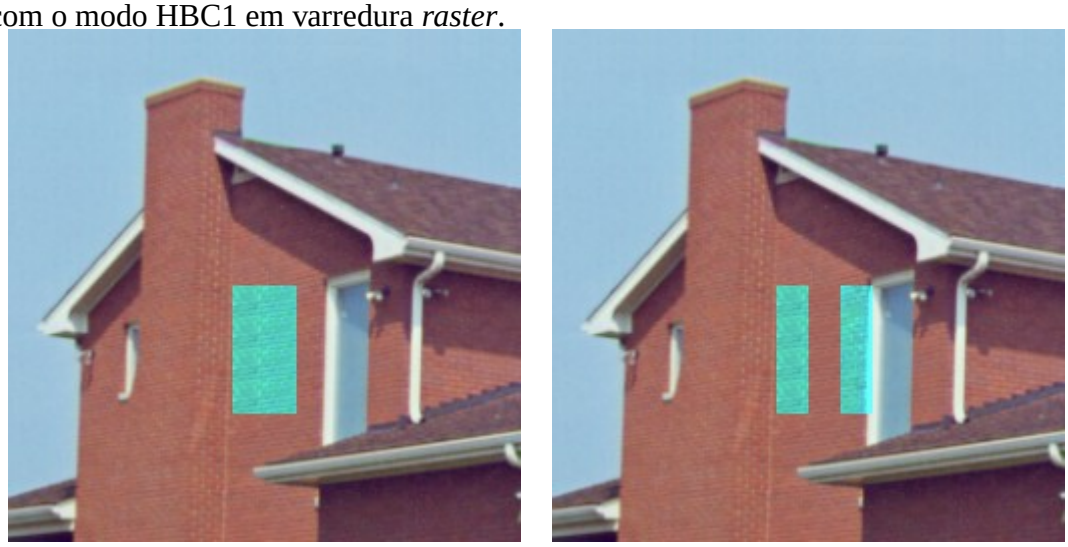


Tabela 1: Notação para o ataque de aniversário avançado

	originais	forjados	coletados
blocos	X_i	B_j	Y_k
códigos de integridade	H_i	F_j	P_k
assinaturas	S_i	L_j	R_k

4.2 Ataque de aniversário avançado

Adotaremos nesta seção a notação da tabela 1 para atribuir rótulos precisos a cada quantidade relevante, e suporemos que a função de *hash* assume m valores possíveis.

No ataque avançado, o adversário constrói variantes de B_t^* e B_{t+1}^*, e então tenta encontrar, entre os códigos de integridade coletados, três deles P_u, P_v, e P_w (com assinaturas associadas R_u, R_v, E R_w) tais que:

$$\begin{aligned} F_t &\equiv H(M, N, t, X_{t-1}^*, B_t^*) &&= P_u, \\ F_{t+1} &\equiv H(M, N, t+1, B_t^*, B_{t+1}^*) &&= P_v, \\ F_{t+2} &\equiv H(M, N, t+2, B_{t+1}^*, X_{t+2}^*) &&= P_w; \end{aligned}$$

e então impõe $L_t = R_u$, $L_{t+1} = R_v$ e $L_{t+2} = R_w$. É importante ter em mente que a base de assinaturas coletadas contém cerca de s pares (P_i, R_i); logo, se um código de integridade ocorre nessa base, certamente existe uma assinatura válida para ele. Com cerca de p variantes de B_t e q variantes de B_{t+1}, o paradoxo do aniversário provê os meios de achar cerca de ps/m soluções B_t para a primeira equação e qs/m soluções B_{t+1} para a terceira equação, onde para cada B_t existe um R_u correspondente e para cada B_{t+1} um R_w correspondente. Espera-se por isso encontrar cerca de $(ps/m)(qs/m)s/m$ soluções para a segunda equação entre os $(ps/m)(qs/m)$ pares (B_t, B_{t+1}), juntamente com o R_v correspondente. Isto conduz, com alta probabilidade, a um par (B_t, B_{t+1}) *e também* a uma tripla (R_u, R_v, R_w) que satisfazem em conjunto as três equações acima. Assumindo que $p \approx q$ e $s \approx m^{1/2}$, e impondo $(ps/m)(qs/m)s/m \approx 1$, obtém-se $p \approx q \approx m^{3/4}$.

Obter-se-ia o mesmo resultado impondo $p \approx q \approx s \approx m^{3/5}$. Esta variante do ataque, porém, tem um custo, na parte dependente do signatário, maior que $O(\sqrt{m})$, que é a complexidade de quebrar o algoritmo assimétrico de assinatura. Na verdade, pelo menos um dentre os parâmetros p, q e s será sempre, no mínimo, $O(m^{3/5})$.

4.3 *Hash Block Chaining*, versão 2 (HBC2)

Conforme dissemos, o modo HBC2 é um aprimoramento de HBC1 que evita ataques de transplante e ataques de aniversário (inclusive o ataque avançado). A nova variante faz uso de certos esquemas não determinísticos de assinatura digital; mais precisamente, esquemas onde o cálculo do índice de *hash* e, paralelamente, da assinatura a partir do *hash* dependem de um valor aleatório *privado*. Quando se usa um algoritmo de assinatura dessa natureza, até mesmo mensagens idênticas produzirão valores diferentes de *hash*, e a fonte dessa diferença não estará acessível a um fraudador potencial. Esta propriedade impede efetivamente ataques de transplante.

O modo HBC2 baseia-se na seguinte fórmula de assinatura:

$$H_t \equiv H(M, N, t, X^*_{(t-1) \bmod MN}, S_{t-1}, X^*_t, r_t), S_t = \text{sign}(H_t, r_t),$$

onde r_t é o valor aleatório privado usado pelo signatário para calcular o *hash* H_t e a assinatura S_t do bloco X_t. Convencionamos aqui $S_{-1} \equiv \varnothing$; note-se que não se pode usar $S_{(t-1) \bmod MN}$, pois a assinatura S_{MN-1} não é conhecida quando se calcula H_0.

Assumimos também que, embora o valor r_t apareça explicitamente no algoritmo de assinatura como um argumento adicional, o algoritmo de verificação não faz uso direto dele. Além disso, dentre as operações de cálculo do *hash* e da assinatura correspondente, é possível tornar apenas uma diretamente dependente de r_t em sua forma privativa. Um exemplo concreto de assinatura desse tipo é o esquema de Schnorr (seção 6.5.2), em que o valor r_t fica apenas implícito no cálculo do *hash* e no algoritmo

de verificação, permanecendo protegido em forma de logaritmo discreto[1].

O ataque de aniversário avançado é completamente inefetivo contra o modo HBC2. Com efeito, suponhamos que um fraudador potencial tenha conseguido substituir dois blocos válidos consecutivos X_t e X_{t+1} por dois blocos forjados B_t e B_{t+1}, e três assinaturas S_t, S_{t+1} e S_{t+2} por três assinaturas ilegítimas (mas válidas) L_t, L_{t+1} e L_{t+2}, mantendo intacto o conteúdo do bloco X_{t+2}. Uma tal substituição já é muito mais difícil para HBC2 que para HBC1 devido ao contexto das assinaturas e ao caráter não determinístico destas. Mesmo neste cenário improvável, o modo HBC2 relata uma alteração na imagem, porque H_{t+3} depende não só do conteúdo de X_{t+2}, que é deixado intacto, mas também de sua assinatura, que se modifica com alta probabilidade (especificamente, a probabilidade de que essa mudança *não* ocorra é apenas $O(2^{-m})$ para funções de *hash* de m bits).

É interessante notar que alguns algoritmos não determinísticos de assinatura digital *não* são adequados para uso com HBC2, a saber, aqueles em que a aleatoriedade está desvinculada do cálculo do *hash*, como o algoritmo DSA[2]. Um problema ainda em aberto é determinar se algum esquema moderno de *padding* (como a construção OAEP+ proposta em (SHOUP, 2001)) pode estabelecer uma ponte entre assinaturas determinísticas e marcas d'água HBC2.

4.4 Propriedades de segurança

4.4.1 Proteção de *hash*

O uso do modo HBC2 tem um efeito colateral surpreendente. Tipicamente, ataques de aniversário podem ser montados contra funções de *hash* que assumem m valores distintos com um esforço computacional de $\sqrt{m}$ passos. Entretanto, contra o HBC2 não se conhece nenhum ataque de complexidade inferior a $O(m)$ passos. A este efeito

[1] Usando a notação da seção 6.5.2, o valor r_t é o inteiro k.

[2] Uma variante recente que corrige este problema é descrita em (MALONE-LEE; SMART, 2003).

damos o nome *proteção de hash*. Conseqüentemente, é aparente que, num cenário otimista, o comprimento de *hash* poderia ser reduzido pela metade, mantendo contudo o nível de segurança original. Esta é a *conjectura da redução*.

Por outro lado, não recomendamos reduzir o tamanho de *hash* até que esta conjectura seja submetida a um escrutínio mais profundo, uma vez que tal redução poderia afetar de maneira adversa a segurança do próprio algoritmo de assinatura. Este receio, conquanto não embasado num ataque efetivo, é inspirado por considerações do ataque de van Oorschot e Wiener (OORSCHOT; WIENER, 1994), que se beneficia de restrições no domínio da chave privada, e também pela observação de como o código de integridade aparece nas equações de assinatura, particularmente nos algoritmos de Schnorr e Nyberg-Rueppel (dual em relação à chave privada).

4.4.2 Relação entre tamanho de *hash* e ordem de grupo

Suponhamos que o *hash* possa assumir m valores distintos e que o grupo onde se efetuam as operações de asinatura contenha n elementos. O ataque avançado de aniversário tem complexidade mínima de $m^{3/5}$, e o logaritmo discreto tem complexidade $n^{1/2}$. Se m e n puderem ser desvinculados (por exemplo, usando assinaturas Schnorr), podemos tornar o ataque de aniversário avançado computacionalmente equivalente ao cálculo do logaritmo discreto escolhendo esses parâmetros de modo que $m^{3/5} \approx n^{1/2}$, ou seja, $m \approx n^{5/6}$. Se a única restrição ao oponente for o acúmulo de assinaturas válidas estar limitado a $n^{1/2}$ valores, embora acarretando um trabalho *offline* de complexidade $m^{3/4}$, os parâmetros podem ser até mesmo escolhidos de modo que $m \approx n^{4/5}$. Esta possibilidade é semelhante à proteção de *hash*, embora quantitativamente mais restrita.

Se m e n forem iguais (por exemplo, se a função de *hash* produzir um elemento do grupo), o próprio ataque de aniversário simples torna-se computacionalmente equivalente ao cálculo do logaritmo discreto, e o ataque de aniversário avançado passa a

ser computacionalmente mais pesado, sendo portanto irrelevante para a segurança do sistema. Com isso, até algoritmos determinísticos como o esquema Wong-Memon tornam-se seguros. A impossibilidade de reduzir o tamanho de m neste caso pode ser compensada pelo uso de assinaturas digitais naturalmente compactas (capítulo 7).

Nesse panorama, a estratégia a seguir é substituir a dependência não determinística na fórmula de contextos de *hash* por uma identificação pública, global e única da mensagem hospedeira, uma idéia originalmente proposta por Wong e Memon (WONG; MEMON, 2001):

$$H_t \equiv H(M, N, t, X^*_{(t-1) \bmod MN}, S_{t-1}, X^*_t, id),$$

onde id é o identificador da imagem hospedeira. Por ser um contexto externo à imagem, id não pode ser modificado por um oponente, impedindo naturalmente ataques de transplante. Ao contrário, porém, do esquema de Wong e Memon, esta construção resiste completamente ao ataque de aniversário simples, devido à dependência entre os blocos.

4.4.3 Optimalidade de HBC2

É legítimo questionarmos se seria possível projetar um algoritmo de marca d'água que não apresentasse nenhum dos problemas de eficiência do modo HBC, e ao mesmo retivesse sua capacidade de localizar alterações e sua resistência a todos os ataques conhecidos de fraude.

O modo HBC2 usa uma fórmula recursiva para encadear valores de *hash* e assinaturas, que devem ser calculadas para cada bloco da imagem. A idéia óbvia de otimização seria procurar uma construção com um código simétrico de autenticação de mensagem para marcar cada bloco, substituindo a chave simétrica por um valor efêmero e descartável derivado da chave pública do signatário (de modo que esse valor estivesse pública e inequivocamente associado com o signatário) e um *nonce* em

contexto específico de cada bloco.

Infelizmente, essa abordagem – como também qualquer outra que procure reduzir o número de assinaturas digitais para menos de uma por bloco – está essencialmente incorreta, conforme estabelece o seguinte resultado, simples mas até o momento inédito:

Teorema 1. *Qualquer algoritmo de marca d'água topológica pública necessariamente efetuará, sob pena de perder a capacidade de localizar alterações, pelo menos N operações assimétricas independentes, seja para marcar, seja para verificar uma imagem particionada em N blocos.*

Demonstração. Suponhamos que o signatário encripte uma semente de *offsets* (qualquer que seja a sua origem) com sua chave privada, de modo que a semente possa ser publicamente recuperada. Se a semente não puder ser inequivocamente associada com a imagem marcada, o que significa depender do conteúdo e da estrutura dessa imagem, então nada poderá impedir que um fraudador use essa mesma semente inalterada para forjar marcas d'água para outras imagens. Contudo, se a semente tornar-se dependente da imagem inteira (por exemplo, usando o *hash* da imagem inteira), então qualquer mudança na imagem irá danificar a semente recuperada para efeitos de verificação, causando que todos os blocos sejam relatados como alterados ou inválidos, e em conseqüência destruindo a capacidade de localizar as alterações na imagem. Portanto, cada bloco deve receber uma semente independente. Mas assumimos no início que a semente é cifrada sob a chave privada do signatário; assim, o signatário deve executar uma operação assimétrica para cada bloco. □

Em resumo, parece extremamente difícil, se não impossível, combinar um algoritmo eficiente de autenticação simétrica (que seria responsável pela maior parte do trabalho) com uma assinatura digital assimétrica (que tornaria o esquema publicamente verificável) sem perder resolução na localização de alterações.

Por fim, cabe notar que não pudemos encontrar uma demonstração formal de segurança (cfr. (BELLARE; ROGAWAY, 1993)) para o modo HBC2 por redução à segurança das primitivas criptográficas envolvidas. Em termos simples, uma tal demonstração garantiria que qualquer vulnerabilidade no esquema HBC2 remeteria a uma vulnerabilidade intrínseca de uma das primitivas, por exemplo, uma brecha de segurança no algoritmo de assinatura digital ou uma falha da função de *hash* utilizada. Infelizmente, esse tipo de demonstração esbarra em certas dificuldades conceituais, como o conceito exato de segurança de uma marca d'água *topológica* – não basta a capacidade de detectar alterações, mas de localizá-las, e a resolução de localização não é fixa em nenhum esquema conhecido (nem mesmo HBC2). Um problema interessante de pesquisa é, portanto, a definição de uma métrica de segurança em termos dessa resolução, e a formulação de uma prova formal de segurança quantificada por essa métrica.

4.5 Detecção de inserções e remoções

Como vimos, a fórmula básica de contexto do modo HBC2 é a seguinte:

$$H_t \equiv H(M, N, t, X^*_{(t-1) \bmod MN}, S_{t-1}, X^*_t),$$

onde $S_t = \text{sign}(H_t)$ e $S_{-1} \equiv \varnothing$.

Todos os itens parecem necessários: X_t é o próprio conteúdo do bloco; M, N e t detectam rearranjos que mantêm a seqüência de varredura inalterada e remoções no fim da seqüência; $X_{t-1) \bmod MN}$ fornece contexto de conteúdo, S_{t-1} introduz unicidade no contexto e evita ataques de transplante.

Por outro lado, HBC2 perde a resolução se um bloco for inserido ou removido, pois isso altera não só o valor de t para alguns blocos, mas principalmente M e N, que afetam todos os blocos.

Embora não tenhamos encontrado uma solução definitiva para esse problema, é

possível definir variantes do modo HBC que fazem frente parcial a essas limitações.

Na primeira variante, introduzimos uma assinatura adicional S'_0 para o primeiro bloco, e descartamos os argumentos M, N e t:

$$\begin{aligned} H_0 &\equiv H(X^*_0), \\ H_t &\equiv H(X^*_t, X^*_{t-1}, S_{t-1}),\ 1 \leqslant t \leqslant N-1, \\ H'_0 &\equiv H(X^*_0, X^*_{N-1}, S_{N-1}), \end{aligned}$$

onde $S_t \equiv S_K(H_t)$ e $S'_0 \equiv S_K(H'_0)$. Contudo, a assinatura adicional potencialmente complica a inserção da marca d'água: o primeiro bloco precisa ser duas vezes maior que os outros, ou cada bloco precisa conter alguns bits da assinatura adicional.

Na segunda variante, troca-se o par de assinaturas do primeiro bloco por um sentinela (um bloco fictício anterior ao primeiro bloco) e uma fórmula especial de assinatura para o último bloco:

$$\begin{aligned} H_0 &\equiv H(X^*_0, X^*_{-1}, S_{-1}, 0), \\ H_t &\equiv H(X^*_t, X^*_{t-1}, S_{t-1}, 1),\ 0 < t < N-1, \\ H_{N-1} &\equiv H(X^*_{N-1}, X^*_{N-2}, S_{N-2}, 2), \end{aligned}$$

onde $S_t \equiv S_K(H_t)$ e X^*_{-1} é um bloco fictício, com uma assinatura igualmente fictícia (por exemplo, todos os bits nulos). A presença do bloco fictício dificulta ataques de aniversário, pois o *hash* é sempre calculado sobre dados do mesmo tamanho. O argumento final torna únicos os contextos do primeiro e do último bloco.

Na terceira variante, introduzimos uma assinatura interna adicional para cada bloco, obtendo um algoritmo trivialmente paralelizável:

$$\begin{aligned} H_t &\equiv H(X^*_t), \\ H'_t &\equiv H(X^*_t, X^*_{(t-1) \bmod MN}, S_{(t-1) \bmod MN}), \end{aligned}$$

onde $S_t \equiv S_K(H_t)$ e $S'_t \equiv S_K(H'_t)$ (apenas a segunda assinatura é mantida na marca d'água).

Aparentemente, todas essas variantes resistem aos mesmos ataques que o algoritmo HBC2 (por raciocínios análogos aos da análise de segurança daquela variante), e também são sensíveis a inserções e remoções, sem perda completa de resolução na localização dessas alterações.

4.6 Inserção de dados semânticos

Conforme mencionamos na seção 2.2, é freqüentemente interessante, vantajoso ou necessário inserir dados adicionais numa imagem além das informações sobre sua origem, integridade e autenticidade. Este é o caso, por exemplo, de marcas d'água visualmente reconhecíveis após sua extração, como logotipos e legendas.

É plausível que a etapa final da verificação esteja condicionada precisamente a esse tipo de reconhecimento visual, particularmente se o volume de verificações for baixo demais para justificar um sistema automático em grande escala: uma marca d'água somente seria aceita se sua aparência visual determinar inequivocamente a origem da imagem. Até mesmo a localização de alterações poderia proceder dessa maneira, se houver uma relação simples entre regiões da imagem hospedeira e áreas corrompidas na apresentação visual da marca d'água (por exemplo, um logotipo repetido em cada bloco de um esquema topológico, como é o caso do algoritmo original de Wong (WONG, 1998)).

Alguns algoritmos de assinatura digital permitem naturalmente não só a verificação da assinatura, mas a recuperação de informações associadas a ela. Exemplos desse tipo de algoritmo são as assinaturas RSA 6.5.5 e as assinaturas Nyberg-Rueppel e Pintsov-Vanstone 6.5.3.

Muitos esquemas de assinatura digital baseados no problema do logaritmo dis-

creto não foram projetados para possibilitar recuperação de dados como nos algoritmos Nyberg-Rueppel e Pintsov-Vanstone. Exemplos de tais esquemas são DSA e Schnorr. No entanto, a maioria deles admite canais subliminares (SIMMONS, 1994) que podem ser usados para embutir uma marca d'água visualmente significativa numa assinatura digital. O canal subliminar mais simples do DSA é a própria escolha do fator aleatório k necessário à confecção de uma assinatura. O possuidor da chave privada x pode usá-la para recuperar k da assinatura, portanto k pode ser escolhido de maneira a conter informações adicionais. Isto deve ser feito com cuidado, uma vez que todos os esquemas de assinatura baseados no problema do logaritmo discreto assumem (e suas propriedades de segurança baseiam-se nesta hipótese) que k é um inteiro estatisticamente único e imprevisível[3].

Existem canais subliminares que podem ser lidos sem o conhecimento direto da chave privada de assinatura x (SIMMONS, 1994). Usar canais subliminares em assinaturas tem a vantagem de que uma marca d'água baseada neles pode ser publicamente verificada sem a recuperação das informações subliminares, enquanto o possuidor da imagem ou um terceiro acreditado podem não só verificá-la mas também recuperar a informação adicional aí embutida.

Infelizmente, a presença de informações adicionais na marca d'água é incompatível com a minimização das informações inseridas. Algoritmos de assinatura que permitem a recuperação de mensagem e algoritmos que admitem canais subliminares invariavelmente produzem assinaturas de tamanho suboptimal, no sentido de existirem outros algoritmos que, sem apresentarem essas propriedades de recuperação de informações, todavia produzem assinaturas proporcionalmente mais compactas para o mesmo nível de segurança, como acontece com o esquema BLS (BONEH; LYNN; SHACHAM, 2002). Assim, uma escolha precisará ser feita de antemão entre um reco-

[3]Idealmente, a informação subliminar deveria ser cifrada com *one-time pad* (STINSON, 2002, seção 2.1) para produzir k; na prática, a informação será cifrada com um algoritmo simétrico usando uma chave estatisticamente única e imprevisível.

nhecimento visual de marcas d'água ou a minimização do volume de informações a ser inserido na imagem hospedeira. O projeto de esquemas balanceados que procurem atingir da melhor maneira possível esses dois objetivos simultaneamente ainda é, em traços gerais, um problema aberto de pesquisa, e nele não nos deteremos em maior detalhe.

4.7 Extensões

Os esquemas de marca d'água aqui delineados podem ser estendidos para dados N-dimensionais. Aplicações possíveis são marcas d'água em sinais de áudio ($N = 1$) e vídeo ($N = 3$). Contudo, o processamento de marcas d'água em vídeo provavelmente precisaria ser realizado em tempo real, um requisito geralmente incompatível com o uso de assinaturas digitais devido ao custo computacional dos algoritmos assimétricos. Uma possibilidade natural para fazer frente a essa situação é diminuir o número de blocos do esquema topológico empregado, restringindo a capacidade de localização de alterações (é plausível que uma localização individual apenas de *frames* alterados seja suficiente para muitas aplicações). Isso indica que um aspecto importante em futuras pesquisas neste campo é o da definição de métricas para a localizabilidade de alterações, no sentido de *quantificar* que tipos de detalhes são importantes em cada tipo de sinal marcado, e qual a relação entre esses detalhes e o formato do sinal, de modo a minimizar o volume de dados assinados e maximizar a resolução na localização de alterações relevantes.

A utilização de *hardware* dedicado pode também mitigar o impacto da verificação de marcas d'água, mas limita sua aplicabilidade aos ambientes onde esse tipo de recurso estiver disponível. Ainda é um problema em aberto a definição de assinaturas digitais com tempos de verificação comparáveis aos de algoritmos simétricos (códigos de autenticação de mensagem).

Outro tema de pesquisa na área de marcas d'água é como alterar a transformada discreta de Fourier de modo a preservar alguma propriedade dos coeficientes durante a compressão de uma imagem. A propriedade conservada poderia servir como base para as assinaturas digitais associadas, tornando a marca d'água menos sensível a mudanças no nível de compressão da imagem.

Finalmente, um problema em aberto óbvio é o da construção de *marcas d'água universais* (simultaneamente robustas e frágeis). Conforme já mencionamos, este problema é especialmente difícil em face do resultado parcial de Barak et al. (BARAK et al., 2001).

PARTE II

ALGORITMOS CRIPTOGRÁFICOS

5 FUNDAMENTOS MATEMÁTICOS

Adotaremos no que segue as seguintes notações, convenções e definições básicas, e apresentaremos um resumo do ferramental matemático de que nos valemos no corpo da tese.

5.1 Notações e definições

Denota-se $A \oplus B$ o ou-exclusivo das representações binárias de A e B, independentemente da semântica própria desses operandos.

O *peso de Hamming* de um inteiro k, denotado $\omega(k)$, é o número de bits não nulos na representação binária de k.

$\mathbb{Z}_n$ denota o conjunto de inteiros módulo n, isto é, $\{0, 1, \ldots, n-1\}$, onde o resultado das operações aritméticas elementares é reduzido módulo n.

$\mathbb{Z}_n^c$ representa o produto Cartesiano $\mathbb{Z}_n \times \mathbb{Z}_n \times \cdots \times \mathbb{Z}_n$ com c fatores.

$\mathbb{Z}_n[x]$ denota o conjunto dos polinômios com coeficientes em $\mathbb{Z}_n$. Um polinômio $r(x) \in \mathbb{Z}_n[x]$ diz-se *irredutível* se $r(x)$ não pode ser escrito como produto de outros polinômios em $\mathbb{Z}_n[x]$ de grau positivo mas menor que o de $r(x)$.

Sejam m e n dois inteiros. Escreve-se $n \mid m$ se m é um múltiplo de n (lê-se "n divide m"), e $n \nmid m$ caso contrário.

5.1.1 Complexidade de algoritmos

Definição 8. *Dada uma função* $g : \mathbb{N} \to \mathbb{R}$ *tal que* $\exists n_0 \in \mathbb{N}, \forall n > n_0 : g(n) > 0$*, diz-se que uma função* $f : \mathbb{N} \to \mathbb{R}$ *tem a* ordem *de* g*, escrita* $O(g)$*, se, e somente se,* $\exists c \geqslant 0, n_0 \in \mathbb{N}, \forall n \in \mathbb{N} : n > n_0 \Rightarrow 0 \leqslant f(n) \leqslant cg(n)$*, e tem a* ordem exata[1] *de* g*, escrita* $\Theta(g)$*, se, e somente se,* $\exists c, d \geqslant 0, n_0 \in \mathbb{N}, \forall n \in \mathbb{N} : n > n_0 \Rightarrow 0 \leqslant cg(n) \leqslant f(n) \leqslant dg(n)$.

Uma função f é de *ordem polinomial* se existe algum polinômio $p : \mathbb{N} \to \mathbb{R}$ tal que f é $O(p)$; de *ordem exponencial* se existe alguma constante $c > 0$ tal que f é $O(e)$ para a função $e : \mathbb{N} \to \mathbb{R}$, $e(n) = \exp(cn)$; finalmente, de *ordem subexponencial* (ou *superpolinomial*) se f é $O(L[\alpha, n])$ onde $L[\alpha, n] \equiv \exp(cn^{\alpha} \ln^{1-\alpha} n)$, $0 < \alpha < 1$, para algum $c > 0$.

5.2 Grupos e corpos finitos

Os conceitos algébricos delineados nesta seção são tratados com profundidade em (HOFFMAN; KUNZE, 1971; LIDL; NIEDERREITER, 1997; MACLANE; BIRKHOFF, 1993).

Definição 9. *Um* grupo abeliano *é um par* $(\mathbb{G}, \circ)$*, onde* $\mathbb{G}$ *é um conjunto não vazio e* $\circ : \mathbb{G} \times \mathbb{G} \to \mathbb{G}$ *é uma operação binária satisfazendo as seguintes propriedades:*

- *(Existência do elemento neutro)* $\exists n \in \mathbb{G} : \forall a \in \mathbb{G} : a \circ n = a$.
- *(Existência do elemento inverso)* $\forall a \in \mathbb{G} : \exists \bar{a} \in \mathbb{G} : a \circ \bar{a} = n$.
- *(Associatividade)* $\forall a, b, c \in \mathbb{G} : (a \circ b) \circ c = a \circ (b \circ c)$.
- *(Comutatividade)* $\forall a, b \in \mathbb{G} : a \circ b = b \circ a$.

[1] Por simplicidade, no restante deste trabalho (exceto quando explicitamente indicado) usaremos a noção de ordem (notação O) para expressar a complexidade de algoritmos, mas em muitos casos seria mais apropriado usar a noção de ordem exata (notação Θ).

O símbolo $\circ$ é geralmente substituído pelo sinal de adição $+$ (em cujo caso o elemento neutro é escrito 0 e o inverso de a é escrito $-a$) ou pelo sinal de multiplicação $\cdot$ (em cujo caso o elemento neutro é escrito 1 e o inverso de a é escrito a^{-1}). Em grupos aditivos, dado um inteiro positivo k define-se a *multiplicação por escalar* como a operação $[k]a \equiv a+a+\cdots+a$, onde a soma consiste em k termos. Por extensão, define-se $[0]a \equiv 0$ (elemento neutro aditivo do grupo) e $[-k]a \equiv [k](-a)$. Similarmente, em grupos multiplicativos define-se a *potenciação* como a operação $a^k = a \cdot a \cdot \cdots \cdot a$ com k fatores. Havendo numa estrutura algébrica uma única operação binária, a escolha entre a notação aditiva e a notação multiplicativa é irrelevante, freqüentemente motivada por motivos históricos.

Um grupo $\mathbb{G}$ diz-se *cíclico* se existe um elemento $G \in \mathbb{G}$ tal que qualquer elemento $P \in \mathbb{G}$ possa ser escrito como múltiplo escalar de G, isto é, $P = [\lambda]G$ para algum inteiro λ. Um elemento G com essa propriedade chama-se *gerador* de $\mathbb{G}$. Escreve-se também $\langle G \rangle \equiv \mathbb{G}$ para ressaltar qual gerador está sendo usado.

Definição 10. *Um* corpo (comutativo) *é uma estrutura algébrica* $(\mathbb{F}, +, \cdot)$*, onde* $(\mathbb{F}, +)$ *é um grupo abeliano (com elemento neutro 0) e* $(\mathbb{F} - \{0\}, \cdot)$ *é um grupo abeliano (com elemento neutro* $1 \neq 0$*), satisfazendo adicionalmente a seguinte propriedade:*

- *(Distributividade)* $a \cdot (b + c) = a \cdot b + a \cdot c$*.*

Um corpo pode ser finito ou infinito. Nosso interesse estará centrado em corpos finitos, para os quais vale a seguinte propriedade notável:

Teorema 2 (Kronecker). *O número de elementos de um corpo finito é sempre da forma* p^m*, onde* p *é um número primo* m *é um inteiro positivo. Além disso, para todo primo* p *e todo inteiro positivo* m *existe um e um só corpo finito com* p^m *elementos.*

O primo p é chamado *característica* do corpo, e m é seu *grau de extensão*. O único corpo finito com p^m elementos é denotado $\mathbb{F}_{p^m}$. $\mathbb{F}_p$ coincide com $\mathbb{Z}_p$, o conjunto

dos inteiros módulo p. Vale ressaltar que o conjunto $\mathbb{Z}_n$ é um corpo se, e somente se, n for um número primo (de modo geral, $\mathbb{Z}_n$ constitui uma estrutura algébrica de *anel*, semelhante à de um corpo mas desprovida da propriedade da existência de inversos multiplicativos para todos os elementos não nulos).

Escreve-se simplesmente $\mathbb{F}_q$ com $q = p^m$ quando a característica e/ou o grau de extensão são claros pelo contexto ou irrelevantes para a discussão. Denota-se também $\mathbb{F}_q^* \equiv \mathbb{F}_q - \{0\}$.

Outra propriedade extremamente útil dos corpos finitos é o *Pequeno Teorema de Fermat*:

Teorema 3 (Pequeno Teorema de Fermat). *$x^q = x$ para todo $x \in \mathbb{F}_q$.*

Demonstração. (LIDL; NIEDERREITER, 1997, lema 2.3) □

É conveniente definir, no contexto do Pequeno Teorema de Fermat, o conceito de *mapa de Frobenius*:

Definição 11. *O* mapa de Frobenius *no corpo finito $\mathbb{F}_{p^m}$ é a função $\Phi : \mathbb{F}_{p^m} \to \mathbb{F}_{p^m}$, $\Phi(x) = x^p$.*

Devido ao Pequeno Teorema de Fermat, o mapa de Frobenius coincide com a identidade se o grau de extensão de um corpo for unitário. Uma das propriedades do mapa de Frobenius é a linearidade: $\Phi(x + y) = \Phi(x) + \Phi(y)$, e $\Phi(\alpha x) = \alpha\Phi(x)$, para todo $x, y \in \mathbb{F}_{p^m}$ e $\alpha \in \mathbb{F}_p$.

A despeito da unicidade, o corpo $\mathbb{F}_{p^m}$ pode ser representado de várias maneiras. A mais comum é a *representação polinomial*: dado um polinômio irredutível $r(x)$ de grau m, o corpo $\mathbb{F}_{p^m}$ é isomorfo ao conjunto $\mathbb{F}_p[x]/r(x)$ dos polinômios módulo $r(x)$ com coeficientes em $\mathbb{F}_p$.

Além da representação polinomial, consideraremos também a representação em *base normal*, que é particularmente eficiente para implementações em hardware.

Definição 12. *Uma* base normal *de $\mathbb{F}_{p^m}$ é um conjunto linearmente independente $\{\theta^{p^i} \mid 0 \leqslant i < m\}$ onde θ é uma raiz em $\mathbb{F}_{p^m}$ de um polinômio irredutível de grau m em $\mathbb{F}_p[x]$.*

Assim, $\mathbb{F}_{p^m}$ é visto como espaço vetorial sobre $\mathbb{F}_p$, expresso quer na base polinomial $\{x^i \mid 0 \leqslant i < m\}$, quer na base normal $\{\theta^{p^i} \mid 0 \leqslant i < m\}$.

O cálculo do mapa de Frobenius é particularmente eficiente em base normal, pois consiste em simples rotação de coeficientes: $\Phi(\sum_i \alpha_i \theta^{p^i}) = \sum_i \alpha_i \Phi(\theta^{p^i}) = \sum_i \alpha_i \theta^{p^{(i+1) \bmod m}} = \sum_i \alpha_{(i-1) \bmod m} \theta^{p^i}$.

5.3 Curvas elípticas

Definição 13. *Uma* curva elíptica *E sobre um corpo finito $\mathbb{F}_q$ é o conjunto $E(\mathbb{F}_q)$ das soluções (x, y) sobre $\mathbb{F}_q$ de uma equação da forma $E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$, onde $a_i \in \mathbb{F}_q$, juntamente com um ponto adicional chamado* ponto no infinito, *denotado O. Em outras palavras, $E(\mathbb{F}_q) \equiv \{(x, y) \in \mathbb{F}_q \times \mathbb{F}_q \mid y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \{O\}$. A mesma equação define curvas em extensões $\mathbb{F}_{q^m}$ para $m > 0$.*

Existe uma lei de grupo abeliano em E de origem geométrica (conhecida como lei das "secantes e tangentes"). Fórmulas explícitas para calcular as coordenadas de um ponto $P_3 = P_1 + P_2$ a partir das coordenadas de P_1 e P_2 são dadas em (SILVERMAN, 1986, algoritmo 2.3); apresentaremos adiante um subconjunto criptograficamente relevante dessas fórmulas. Apenas corpos finitos com grau de extensão m unitário ou primo serão usados para evitar ataques descendentes de Weil (GALBRAITH; HESS; SMART, 2002; GALBRAITH; SMART, 1999; GAUDRY; HESS; SMART, 2002).

A partir da lei de grupo, define-se o produto de um escalar $k \in \mathbb{Z}$ por um ponto P como o ponto $V = [k]P \equiv P + P + \cdots + P$ (com k termos) se $k > 0$, e por extensão $[0]P = O$ e $[-k]P = [k](-P) = -([k]P)$.

5.3.1 Endomorfismo de Frobenius

O *endomorfismo de Frobenius* é a função $\Phi : E(\mathbb{F}_{q^k}) \rightarrow E(\mathbb{F}_{q^k})$, $(X, Y) \mapsto (X^q, Y^q)$. Note-se que um ponto $P \in E(\mathbb{F}_{q^k})$ está definido sobre $\mathbb{F}_{q^i}$ se, e somente se, $\Phi^i(P) = P$; em particular, $\Phi^k(P) = P$ para qualquer $P \in E(\mathbb{F}_{q^k})$. O polinômio característico de Φ é $\pi(u) = u^2 - tu + q$. O valor t chama-se o *traço do endomorfismo de Frobenius* (ou simplesmente *traço de Frobenius*).

5.3.2 Ordem de um ponto e de uma curva

O número de pontos de uma curva elíptica $E(\mathbb{F}_q)$, denotado $\#E(\mathbb{F}_q)$, é chamado *ordem* da curva sobre o corpo $\mathbb{F}_q$. O *teorema de Hasse* (MENEZES, 1993, teorema 2.8) estabelece que $\#E(\mathbb{F}_q) = q + 1 - t$, onde $|t| \leqslant 2\sqrt{q}$ é o traço de Frobenius. De interesse particular (mas não exclusivo) são as curvas *supersingulares*, que são curvas cujo traço de Frobenius t é múltiplo da característica p. Curvas não supersingulares são comumente denominadas *ordinárias*.

Seja $n \equiv \#E(\mathbb{F}_q)$. A ordem de um ponto $P \in E$ é o menor inteiro $r > 0$ tal que $[r]P = O$. Dado um inteiro $r > 0$, o conjunto de todos os pontos $P \in E$ tais que $[r]P = O$ denomina-se conjunto de r-torção de E, escrito $E[r]$ ou $E(K)[r]$ (para reforçar que se trata do grupo dos pontos da curva E definida sobre o corpo K). A ordem de um ponto sempre divide a ordem da curva. Segue daí que $\langle P \rangle$ é um subgrupo de $E[r]$.

5.3.3 Fórmulas da lei de grupo

A tabela 2 lista algumas curvas supersingulares de interesse criptográfico; a semântica de k (o *grau de imersão* da curva) é discutida na seção 5.3.4.

Efetuam-se operações aritméticas nas curvas da tabela 2 e em curvas ordinárias $E(\mathbb{F}_p) : y^2 = x^3 + ax + b$ com $p > 3$ de acordo com as seguintes regras. Sejam

Tabela 2: Curvas elípticas supersingulares amigáveis a emparelhamentos

equação da curva	corpo finito	ordem da curva	k
$E_{1,b}: \quad y^2 = x^3 + (1-b)x + b$, $b \in \{0,1\}$	$\mathbb{F}_p$, $p > 3$ primo, $p \equiv 2 \bmod 3$	$p+1$	2
$E_{2,b}: \quad y^2 + y = x^3 + x + b$, $b \in \{0,1\}$	$\mathbb{F}_{2^m}$, m primo	$2^m + 1 \pm 2^{(m+1)/2}$	4
$E_{3,b}: \quad y^2 = x^3 - x + b$, $b \in \{-1,1\}$	$\mathbb{F}_{3^m}$, m primo	$3^m + 1 \pm 3^{(m+1)/2}$	6

$P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2)$, $P_3 = P_1 + P_2 = (x_3, y_3)$. Por definição, $-O = O$, $-P_1 = (x_1, -y_1)$, $P_1 + O = O + P_1 = P_1$, e $P_3 = O$ se $P_1 = -P_2$. Além disso,

- Curva $E_{1,b}$:

$$\begin{aligned} P_1 = P_2 \quad \Rightarrow \quad \lambda &\equiv \frac{3x_1^2 + 1 - b}{2y_1}, \\ x_3 &= \lambda^2 - 2x_1, \\ y_3 &= \lambda(x_1 - x_3) - y_1. \end{aligned}$$

$$\begin{aligned} P_1 \neq P_2, -P_2 \quad \Rightarrow \quad \lambda &\equiv \frac{y_2 - y_1}{x_2 - x_1}, \\ x_3 &= \lambda^2 - (x_1 + x_2), \\ y_3 &= \lambda(x_1 - x_3) - y_1. \end{aligned}$$

- Curva $E_{2,b}$:

$$\begin{aligned} P_1 = P_2 \quad \Rightarrow \quad x_3 &= x_1^4 + 1, \\ y_3 &= x_1^4 + y_1^4. \end{aligned}$$

$$\begin{aligned} P_1 \neq P_2, -P_2 \quad \Rightarrow \quad \lambda &\equiv \frac{y_1 + y_2}{x_1 + x_2}, \\ x_3 &= \lambda^2 + x_1 + x_2, \\ y_3 &= \lambda(x_1 + x_3) + y_1 + 1. \end{aligned}$$

- Curva $E_{3,b}$:

$$\begin{aligned} P_1 = P_2 \quad \Rightarrow \quad \lambda &\equiv 1/y_1, \\ x_3 &= x_1 + \lambda^2, \\ y_3 &= -y_1 - \lambda^3. \end{aligned}$$

$$\begin{aligned} P_1 \neq P_2, -P_2 \quad \Rightarrow \quad \lambda &\equiv \frac{y_2 - y_1}{x_2 - x_1}, \\ x_3 &= \lambda^2 - x_1 - x_2, \\ y_3 &= y_1 + y_2 - \lambda^3. \end{aligned}$$

- Curvas ordinárias:

$$\begin{aligned} P_1 = P_2 \quad \Rightarrow \quad \lambda &\equiv \frac{3x_1^2 + a}{2y_1}, \\ x_3 &= \lambda^2 - 2x_1, \\ y_3 &= \lambda(x_1 - x_3) - y_1. \end{aligned}$$

$$\begin{aligned} P_1 \neq P_2, -P_2 \quad \Rightarrow \quad \lambda &\equiv \frac{y_2 - y_1}{x_2 - x_1}, \\ x_3 &= \lambda^2 - (x_1 + x_2), \\ y_3 &= \lambda(x_1 - x_3) - y_1. \end{aligned}$$

As regras para aritmética com os pontos de uma curva elíptica são a base do método da *decomposição binária* para calcular múltiplos escalares. Dado $k > 0$, seja $k = (k_t k_{t-1} \ldots k_1 k_0)_2$ onde $k_i \in \{0, 1\}$ e $k_t \neq 0$ sua representação binária. O cálculo de $V = [k]P$ procede da seguinte maneira.

Multiplicação por escalar via decomposição binária:

$V \leftarrow P$

para $i \leftarrow t-1, t-2, \ldots, 1, 0$ **faça** {

  $V \leftarrow [2]V$

  se $k_i = 1$ **então** $V \leftarrow V + P$

}

devolva V

A validade deste algoritmo prova-se por indução no número de dígitos binários de k. Várias otimizações deste esquema básico são discutidas em (BLAKE; SEROUSSI; SMART, 1999; MENEZES; OORSCHOT; VANSTONE, 1999). Em todos os casos, a complexidade é $O(\log^3 q)$ passos para curvas sobre $\mathbb{F}_q$, as diversas variantes diferindo numa constante de proporcionalidade.

5.3.4 Grau de imersão

Seja P um ponto em E de ordem prima r onde $r^2 \nmid n$. Diz-se que o subgrupo $\langle P \rangle$ tem *grau de imersão* k para algum inteiro $k > 0$ se $r \mid q^k - 1$ e $r \nmid q^s - 1$ para todo $0 < s < k$. Estaremos interessados em curvas cujo grau de imersão é grande o suficiente para manter um nível elevado de segurança, mas pequeno o bastante para permitir que certas operações sejam efetuadas eficientemente. O grau de imersão de curvas elípticas ordinárias é, via de regra, enorme (BALASUBRAMANIAN; KOBLITZ, 1998; MIYAJI; NAKABAYASHI; TAKANO, 2001). Entretanto, se E é supersingular, o valor de k é limitado por $k \leqslant 6$ (MENEZES; OKAMOTO; VANSTONE, 1993). Este limite é atingido por curvas definidas sobre corpos de característica 3 mas não sobre corpos de característica 2, onde o valor máximo atingível é $k = 4$ (MENEZES, 1993, seção 5.2.2).

O grupo $E(\mathbb{F}_q)$ é isomorfo a um subgrupo de $E(\mathbb{F}_{q^k})$ (justificando o nome grau de imersão). Seja $P \in E(\mathbb{F}_q)$ um ponto de ordem r tal que $\langle P \rangle$ tenha grau de imersão k. Então $E(\mathbb{F}_{q^k})$ contém um ponto Q da mesma ordem r mas linearmente independente de P, no sentido de que a única combinação linear da forma $[\alpha]P + [\beta]Q = O$ é aquela onde $\alpha = \beta = 0$.

5.3.5 Traço de uma curva

O *traço* de uma curva elíptica (que não deve ser confundido com o traço de Frobenius) é a função $\mathrm{tr} : E(\mathbb{F}_{q^k}) \to E(\mathbb{F}_q)$ definida como $\mathrm{tr}(P) = P + \Phi(P) + \Phi^2(P) + \cdots + \Phi^{k-1}(P)$. Segue daí que $\mathrm{tr}(\Phi(P)) = \Phi(\mathrm{tr}(P)) = \mathrm{tr}(P)$ para todo $P \in E(\mathbb{F}_{q^k})$, comprovando que a imagem de tr é realmente $E(\mathbb{F}_q)$.

Assumindo que k e r são primos entre si, os autovalores de tr sobre $E[r]$ são k e 0. Descreveremos agora os auto-espaços de tr associados a esses autovalores.

Lema 1. *O auto-espaço associado ao autovalor k de tr é $E(\mathbb{F}_q)[r]$.*

Demonstração. Claramente, todos os pontos $R \in E(\mathbb{F}_q)$ satisfazem $\mathrm{tr}(R) = [k]R$, de modo que só precisamos mostrar que todos os pontos $R \in E[r]$ tais que $\mathrm{tr}(R) = [k]R$ são definidos sobre $\mathbb{F}_q$. Com efeito, se $\mathrm{tr}(R) = [k]R$, então $\Phi(\mathrm{tr}(R)) = \Phi([k]R) = [k]\Phi(R)$, mas como $\Phi(\mathrm{tr}(R)) = \mathrm{tr}(R)$, segue que $[k]\Phi(R) = \mathrm{tr}(R) = [k]R$ e assim $[k](\Phi(R) - R) = O$. Como $\Phi(R) - R \in E[r]$ e k é coprimo a r, necessariamente $\Phi(R) - R = O$, e assim R deve estar definido sobre $\mathbb{F}_q$, isto é, $R \in E(\mathbb{F}_q)[r]$. Portanto, $E(\mathbb{F}_q)[r]$ é o auto-espaço de tr correspondente ao autovalor k. □

É fácil verificar que para qualquer $R \in E(\mathbb{F}_{q^k})$ o ponto $Q = R - \Phi(R)$ satisfaz $\mathrm{tr}(Q) = O$. Isto fornece um meio de gerar pontos de traço zero. Uma vez que pelo menos um ponto $Q \neq O$ pode ser construído dessa maneira (supondo $k > 1$), o outro autovalor é necessariamente zero. Este espaço deve ter dimensão 1, já que a outra dimensão está associada a $E(\mathbb{F}_q)[r]$.

Estamos agora em condições de descrever os auto-espaços do endomorfismo de Frobenius. Conforme mencionamos na seção 5.3.1, o polinômio característico de Φ é $\pi(u) = u^2 - tu + q$, que se decompõe como $\pi(u) = (u-1)(u-q) \pmod r$, e portanto seus autovalores são 1 e q.

Lema 2. *O auto-espaço de Φ associado ao autovalor 1 é $E(\mathbb{F}_q)[r]$.*

Demonstração. Imediata: um ponto de $E(\mathbb{F}_{q^k})$ é fixo sob Φ se, e somente se, esse ponto estiver definido sobre $E(\mathbb{F}_q)$. □

Lema 3. *O auto-espaço de Φ associado ao autovalor q consiste de todos os pontos $R \in E[r]$ satisfazendo $\mathrm{tr}(R) = O$.*

Demonstração. Se um ponto R satisfaz $\mathrm{tr}(R) = (1 + \Phi + ... + \Phi^{k-1})R = O$, então $\mathrm{tr}(\Phi(R)) = (\Phi + ... + \Phi^k)R = O$. Em outras palavras, os pontos de traço zero são mapeados a pontos de traço zero por Φ e portanto constituem um auto-espaço. Como o auto-espaço associado a 1 já foi levado em consideração, o conjunto dos pontos de traço zero deve ser o auto-espaço de Φ associado ao autovalor q. □

5.3.6 *Twist* de uma curva

O *twist* de uma curva dada em forma compacta de Weierstraß $E : y^2 = x^3 + ax + b$ é a curva $E' : y^2 = x^3 + a'x + b'$ com $a' = v^2a$ e $b' = v^3b$ para algum não-resíduo quadrático $v \in \mathbb{F}_q$. As ordens dos grupos de pontos racionais dessas curvas satisfazem a relação $\#E(\mathbb{F}_q) + \#E'(\mathbb{F}_q) = 2q + 2$ (BLAKE; SEROUSSI; SMART, 1999, section III.3).

5.3.7 Mapa de distorção

Um *mapa de distorção* (VERHEUL, 2001) é um isomorfismo $\phi : E(\mathbb{F}_q) \to E(\mathbb{F}_{q^k})$ não $\mathbb{F}_q$-racional, isto é, uma função que associa a cada ponto $P \in E(\mathbb{F}_q)$ um ponto $Q \in E(\mathbb{F}_{q^k})$ linearmente independente de P. Mapas de distorção só existem para curvas supersingulares (VERHEUL, 2001, teorema 4.1). A tabela 3 lista os mapas de distorção que utilizaremos para as curvas supersingulares descritas na tabela 2. Note-se que não há uma entrada para $E_{1,1}$.

Tabela 3: Escolha de mapas de distorção

curva	corpo finito	mapa de distorção	condições
$E_{1,0}$	$\mathbb{F}_p : p > 3$, $p \equiv 3 \bmod 4$	$\phi_1(x, y) = (-x, iy)$	$i \in \mathbb{F}_{p^2}$: $i^2 = -1$
$E_{2,b}$, $b \in \{0, 1\}$	$\mathbb{F}_{2^m} : m$ primo	$\phi_2(x, y) = (x + s^2, y + sx + t)$	$s, t \in \mathbb{F}_{2^{4m}}$: $s^4 + s = 0$, $t^2 + t + s^6 + s^2 = 0$
$E_{3,b}$, $b \in \{-1, 1\}$	$\mathbb{F}_{3^m} : m$ primo	$\phi_3(x, y) = (-x + r_b, iy)$	$r_b, i \in \mathbb{F}_{3^{6m}}$: $r_b^3 - r_b - b = 0$, $i^2 = -1$

5.3.8 Multiplicação complexa

Se $E(\mathbb{F}_q)$ é uma curva elíptica ordinária de ordem n, o teorema de Hasse (cfr. seção 5.3.2) garante que a quantidade $4q - t^2$ (com $t = q + 1 - n$) é positiva, e existe uma fatoração única

$$DV^2 = 4q - t^2 = 4n - (t - 2)^2$$

onde D é livre de quadrados (isto é, não contém fatores quadrados). Diz-se que $E(\mathbb{F}_q)$ admite *multiplicação complexa* (CM) por $\sqrt{-D}$. A equação acima é chamada *equação CM* de $E(\mathbb{F}_q)$, e D é chamado *discriminante CM* para q.

Se o discriminante D não for excessivamente grande (na prática, $D < 10^8$), a partir da equação CM pode-se construir uma curva sobre um corpo arbitrário $\mathbb{F}_q$ com a ordem n desejada. A técnica para esse efeito é chamada método Lay-Zimmer (LAY; ZIMMER, 1994), Atkin-Morain (MORAIN, 1991), ou simplesmente método CM.

Uma descrição minuciosa do método CM foge ao escopo deste documento; o leitor interessado pode consultar as referências acima, ou as excelentes exposições em (BLAKE; SEROUSSI; SMART, 1999, capítulo VIII) ou (IEEE P1363 Working Group, 2000, seções A.13 e A.14). A idéia essencial é calcular uma raiz j do *polinômio de Hilbert* $H_D(x)$ (cujos coeficientes são obtidos em função de D), e montar a equação

da curva através das relações:

$$\begin{aligned} j = 0 &\quad : \quad y^2 = x3 + 1; \\ j = 1728 &\quad : \quad y^2 = x^3 + x; \\ j \neq 0, 1728 &\quad : \quad y^2 = x^3 + 3cx + 2c; \end{aligned}$$

onde $c = j/(1728 - j)$. A raiz j utilizada é chamada j-*invariante* da curva resultante[2].

5.4 Teoria de divisores

Seja $E(\mathbb{F}_q)$ uma curva elíptica contendo um subgrupo de ordem prima r com grau de imersão k. Um *divisor*[3] sobre E é uma soma formal $\mathcal{D} = \sum_{P \in E(\mathbb{F}_{q^k})} a_P(P)$ onde $a_P \in \mathbb{Z}$. Em outras palavras, um divisor é uma associação de um coeficiente inteiro a_P a cada ponto $P \in E(\mathbb{F}_{q^k})$, isto é, uma função $\mathcal{D} : E(\mathbb{F}_{q^k}) \to \mathbb{Z}$, $P_i \mapsto a_i$, para todos os pontos $P_i \in E(\mathbb{F}_{q^k})$, $i = 1, \ldots, n$, onde $\#E(\mathbb{F}_{q^k}) = n$. Os parênteses ao redor dos pontos P_i só são usados para lembrar que não se refere aqui ao valor da soma, isto é, $[a_1]P_1 + \cdots + [a_n]P_n$, mas aos coeficientes dos pontos. Em particular, a notação (P) é uma abreviação para o divisor em que os coeficientes de todos os pontos são nulos, exceto o coeficiente do ponto P, que é $a_P = 1$. Assim, $(P) \equiv 0(P_1) + 0(P_2) + \cdots + 1(P) + \cdots + 0(P_n)$.

O conjunto dos pontos $P \in E(\mathbb{F}_{q^k})$ tais que $n_P \neq 0$ chama-se suporte de $\mathcal{D}$. O *grau* de $\mathcal{D}$ é o valor da soma dos coeficientes $\deg(\mathcal{D}) = \sum_P a_P$. O divisor nulo, denotado 0, tem todos os coeficientes nulos, $n_P = 0$. A soma de dois divisores $\mathcal{D} = \sum_P n_P(P)$ e $\mathcal{D}' = \sum_P n'_P(P)$ é o divisor $\mathcal{D} + \mathcal{D}' = \sum_P (n_P + n'_P)(P)$. Induz-se assim uma estrutura de grupo abeliano sobre o conjunto de divisores; em particular,

[2]De modo geral, o j-invariante relaciona-se com os coeficientes da equação da curva $E : y^2 = x^3 + ax + b$ através da relação $j = -1728(4a)^3/\Delta$, onde $\Delta = -16(4a^3 + 27b^2)$ é o discriminante da equação da curva (MENEZES, 1993, Seção 2.4).

[3]Divisores são geralmente definidos sobre o fecho algébrico $\overline{\mathbb{F}_q}$ de $\mathbb{F}_q$ sujeito à condição adicional de que apenas um número finito de coeficientes a_P são não nulos. Contudo, restringiremos nossa atenção a divisores definidos sobre $\mathbb{F}_{q^k}$, de modo que o número total de coeficientes a_P seja finito.

$r\mathcal{D} = \sum_P (ra_P)(P)$.

Uma *função racional* $f : \mathbb{F}_{q^k} \times \mathbb{F}_{q^k} \to \mathbb{F}_{q^k}$ é uma função da forma $f(x,y) = N(x,y)/D(x,y)$, onde $N, D \in \mathbb{F}_{q^k}[x,y]$. Um zero de f é qualquer ponto $(x,y) \in \mathbb{F}_{q^k} \times \mathbb{F}_{q^k}$ tal que $N(x,y) = 0$ e $D(x,y) \neq 0$, e um pólo de f é qualquer ponto $(x,y) \in \mathbb{F}_{q^k} \times \mathbb{F}_{q^k}$ tal que $N(x,y) \neq 0$ e $D(x,y) = 0$ (note-se que f não está propriamente definida em seus pólos). A multiplicidade de f em P, escrita $\text{ord}_P(f)$, é definida como $\deg N$ se P for um zero de f, como $\deg D$ se P for um pólo de f, e como 0 de P for um ponto ordinário de f (i.e. nem um zero nem um pólo de f). Por extensão, define-se $f : E(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}$ como $f(P) = f(x,y)$ para $P = (x,y)$.

Dada uma função racional $f : E(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}$, define-se o *divisor de* f como sendo o divisor $(f) = \sum_P \text{ord}_P(f)(P)$. Segue desta definição que $(fg) = (f) + (g)$ e $(f/g) = (f) - (g)$ para quaisquer duas funções f e g definidas sobre $E(\mathbb{F}_{q^k})$; além disso, $(f) = 0$ se, e somente se, f for uma constante não nula. Em conseqüência, duas funções distintas por um fator constante $c \neq 0$ têm o mesmo divisor, isto é, $(cf) = (f)$ para todo $c \in \mathbb{F}_{q^k}^*$.

Um divisor $\mathcal{D}$ é dito *principal* se $\mathcal{D} = (f)$ para alguma função (f). Sabe-se (MENEZES, 1993, teorema 2.25) que um divisor $\mathcal{D} = \sum_P a_P(P)$ é principal se, e somente se, o grau de $\mathcal{D}$ é zero e $\sum_P a_P P = O$. Dado um ponto $P \in E[r]$, um exemplo importante de divisor principal é $r(P) - r(O)$.

Diz-se que dois divisores $\mathcal{D}$ e $\mathcal{D}'$ são equivalentes, $\mathcal{D}' \sim \mathcal{D}$, se existe uma função g tal que $\mathcal{D}' = \mathcal{D} + (g)$, isto é, se a diferença entre eles for um divisor principal. Em particular, $(P + R) - (R) \sim (P) - (O)$ para quaisquer pontos P e R.

Para qualquer função f e qualquer divisor $\mathcal{D} = \sum_P a_P(P)$ de grau zero, define-se $f(\mathcal{D}) = \prod_P f(P)^{a_P}$. Se f for constante quando calculada sobre os pontos da curva, seu valor será 1 quando calculada sobre um divisor de grau zero, isto é, $f(\mathcal{D}) = 1$ independentemente do valor da constante $f(P)$.

6 CONCEITOS DE ASSINATURA DIGITAL

Neste capítulo examinaremos como os sistemas criptográficos ditos assimétricos possibilitam a criação de assinaturas digitais.

6.1 Funções de *hash*

O conceito de função de *hash* é essencial à obtenção de assinaturas digitais compactas e eficientes.

Intuitivamente, uma função de *hash* calcula, rápida, segura e univocamente, representantes adequadamente curtos (chamados *resumos*) para mensagens arbitrariamente longas. Esses resumos são assinados em lugar das próprias mensagens, mantendo em nível aceitável o esforço computacional comumente encontrado durante a operação de algoritmos assimétricos.

Formalmente, temos:

Definição 14 ((MENEZES; OORSCHOT; VANSTONE, 1999)). *Uma* função de hash *de tamanho n é uma função $H : \{0,1\}^* \rightarrow \{0,1\}^n$ satisfazendo as seguintes propriedades:*

- (Resistência ao cálculo de primeira pré-imagem) *Dada uma cadeia binária h de comprimento n, é computacionalmente intratável encontrar uma mensagem M tal que $H(M) = h$.*
- (Resistência ao cálculo de segunda pré-imagem) *Dada uma cadeia binária h de*

comprimento n e uma mensagem M tal que $H(M) = h$, é computacionalmente intratável encontrar outra mensagem M' tal que $H(M') = h$.

- (Resistência a colisões) *É computacionalmente intratável encontrar duas mensagens M e M' tais que $H(M) = H(M')$, independentemente do valor de $H(M)$.*

Por intratável entende-se aqui que o esforço para obter uma primeira ou segunda pré-imagem seja $O(2^n)$ passos computacionais, e que o esforço para encontrar uma colisão genérica seja $O(2^{n/2})$.

Em outras palavras, uma função de *hash* é uma função computacionalmente unidirecional e resistente a colisões, mapeando cadeias binárias de tamanho arbitrário a cadeias binárias de tamanho fixo n.

Uma família bem conhecida de funções de *hash* para aplicações criptográficas constitui-se das funções SHA-1 e SHA-2 (NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY – NIST, 2002), que produzem valores de 160, 256, 384 ou 512 bits. Nossa própria função de *hash*, agora acolhida na norma ISO/IEC 10118-3:2003(E) como função padrão de *hash* #7, é WHIRLPOOL (BARRETO; RIJMEN, 2000c), que produz valores de 512 bits.

6.2 Oráculos aleatórios

Uma extensão "estilizada" do conceito de função de *hash* é a noção de *oráculo aleatório*. Intuitivamente, um oráculo aleatório é uma função de *hash* perfeitamente opaca; não há maneira de um agressor distinguir entre a saída de um oráculo aleatório e uma cadeia aleatória verdadeira associada univocamente ao parâmetro de entrada. Formalizando:

Definição 15. *Um* oráculo aleatório (BELLARE; ROGAWAY, 1993) *é um mapeamento*

$R : \{0,1\}^* \rightarrow \{0,1\}^\infty$ *onde cada bit de* $R(x)$ *é escolhido uniforme e independentemente, para todo possível argumento* $x \in \{0,1\}^*$. *Interpreta-se a notação* $\{0,1\}^\infty$ *para o conjunto de valores como significando que a saída do oráculo aleatório é "suficientemente longa" para os propósitos de utilização do oráculo.*

Funções de *hash* comumente usadas como SHA-1 e SHA-2 não podem ser consideradas aproximações razoáveis de oráculos aleatórios, pois possuem uma estrutura interna que pode ser aproveitada por um agressor, principalmente quando a entrada dessas funções tem um certo grau de organização (por exemplo, vários argumentos de natureza nitidamente distinta) e pode ser adaptativamente controlada num ataque a algum protocolo. Contudo, várias construções existem que potencialmente simulam um oráculo aleatório a partir de uma função de *hash* comum (BELLARE; ROGAWAY, 1993, seção 1.1).

6.3 Criptografia assimétrica

6.3.1 Problemas matemáticos subjacentes

O fundamento de todo e qualquer algoritmo criptográfico é uma conjectura que há mais de três décadas permanece como problema aberto (COOK, 1971), a saber, a de que certos problemas (denominados $\mathcal{NP}$-completos) solúveis em tempo polinomial por um algoritmo não determinístico *não* podem ser resolvidos em tempo polinomial por nenhum algoritmo determinístico. Esta é a chamada conjectura $\mathcal{P} \neq \mathcal{NP}$.

Vários problemas $\mathcal{NP}$-completos foram propostos como base para definir sistemas criptográficos assimétricos. Dentre eles, os mais conhecidos são:

- Cálculo do logaritmo discreto (vide adiante).
- Fatoração inteira:
  - RSA (RIVEST; SHAMIR; ADLEMAN, 1978),

  - Rabin-Williams (WILLIAMS, 1980).

- Problema da mochila:

  - Hellman-Merkle (MERKLE; E.HELLMAN, 1978),
  - Chor-Rivest (CHOR; RIVEST, 1988).

- Aproximação de vetores em reticulados:

  - Ajtai-Dwork (AJTAI; DWORK, 1997),
  - NTRU (HOFFSTEIN; PIPHER; SILVERMAN, 2001).

- Equações ocultas em corpo finito:

  - HFE (PATARIN, 1996; PATARIN; COURTOIS; GOUBIN, 2000).

Um contra-exemplo é o *problema da mochila*. Este problema originou o sistema Hellman-Merkle, o mais antigo sistema assimétrico completo, capaz de produzir assinaturas e cifrações. Vários outros sistemas criptográficos baseados no problema da mochila foram propostos a partir de então. Contudo, nenhum deles resistiu a técnicas algébricas de criptoanálise; o último sobrevivente da família, o sistema de Chor-Rivest (CHOR; RIVEST, 1988), foi severamente enfraquecido em 1995 (SCHNORR; HÖRNER, 1995) e quebrado completamente em 1998 (VAUDENAY, 1998).

De maneira semelhante, o sistema de Ajtai-Dwork foi quebrado após menos de um ano de criptoanálise. Um ataque devido a Nguyen e Stern (NGUYEN; STERN, 1998) quebra o sistema Ajtai-Dwork com chaves de até 20 MB, que em si mesmo já não seria prático por causar uma expansão dos dados cifrados por um fator 6144 sobre os dados em claro.

O primeiro algoritmo de assinatura digital associado ao sistema NTRU, chamado NSS, foi proposto em (HOFFSTEIN; PIPHER; SILVERMAN, 2001) e quebrado em (GENTRY et al., 2002). Após o ataque, o esquema foi revisado (a versão final do

artigo (HOFFSTEIN; PIPHER; SILVERMAN, 2001) contém a revisão do algoritmo) e novamente quebrado em (GENTRY; SZYDLO, 2002). A variante atual desse algoritmo, chamada NTRUSign (HOFFSTEIN et al., 2001), continua mais ou menos intacta, embora uma análise preliminar em (GENTRY; SZYDLO, 2002) revele que "NTRUSign não pode ter qualquer propriedade formal de segurança, já que não é seguro contra adversários passivos" (*"NTRUSign cannot have any formal security property, since it is not secure against passive adversaries"*). Na verdade, NTRUSign não se baseia exclusivamente na intratabilidade do problema da aproximação de vetores em reticulados, dificultando uma avaliação efetiva de sua segurança.

O problema matemático mais útil para os propósitos deste trabalho é o do *logaritmo discreto* e suas variantes, que abordaremos a seguir.

6.3.2 O problema do logaritmo discreto

Dado um grupo $\mathbb{G}$, o problema do logaritmo discreto (DLP) é definido da seguinte maneira:

- *Problema do Logaritmo Discreto (DLP)* — Dados P, $[a]P \in \mathbb{G}$ onde a ordem de P é r, calcular $a \pmod r$.

O nome "logaritmo discreto" provém da notação multiplicativa para a operação de grupo, isto é, P^a em vez de $[a]P$, sugerindo que a seja o "logaritmo" de P^a na "base" P.

A complexidade dos algoritmos para a resolução do DLP é geralmente expressa em termos do número n de bits do fator a ou da ordem r de $\mathbb{G}$.

O DLP pode ser resolvido, evidentemente, por busca exaustiva (força bruta). Para tal, calculam-se sistematicamente todos os pares $(t,\ [t]P)$ até encontrar um par onde $[t]P = Q$. Esse algoritmo funciona em espaço $O(1)$, mas infelizmente em tempo $O(r) = O(2^n)$ (exponencial).

O método da força bruta é um exemplo de algoritmo genérico para o DLP. Algoritmos desse tipo procuram calcular logaritmos discretos num grupo $\mathbb{G}$ sem fazer uso da natureza peculiar de $\mathbb{G}$. Outros métodos desse tipo incluem o algoritmo de Pohlig e Hellman (POHLIG; HELLMAN, 1978), os algoritmos λ e ρ de Pollard (POLLARD, 1978) e o algoritmo de Oorschot e Wiener (OORSCHOT; WIENER, 1994). Todos eles funcionam em tempo essencialmente exponencial.

Outras técnicas fazem uso da estrutura do grupo onde se quer calcular o logaritmo discreto. Essas técnicas podem resultar em menor complexidade algorítmica (menor número de passos em relação a algoritmos genéricos), mas via de regra não se aplicam a todos os grupos de interesse. Um dos algoritmos dessa classe é o chamado *cálculo de índices* (MENEZES; OORSCHOT; VANSTONE, 1999, seção 3.6.5), aplicável na solução do DLP sobre o grupo multiplicativo de um corpo finito. A complexidade do algoritmo básico do cálculo de índices é subexponencial, $O(L[1/2, n])$, sendo muito mais eficiente do que os algoritmos genéricos exponenciais (conquanto ainda intratável para valores suficientemente grandes de n). O algoritmo mais rápido conhecido para resolver o DLP é uma variante do cálculo de índices chamada *crivo de corpo numérico generalizado* (em inglês, *generalized number field sieve*) (COPPERSMITH, 1984; GORDON, 1993), e possui complexidade heurística assintótica de operação $O(L[1/3, n])$.

A dificuldade do DLP sobre grupos elípticos parece ser essencialmente maior que a dificuldade sobre outros grupos (SILVERMAN, 1986), exceto em alguns casos bastante específicos, facilmente detectáveis e evitáveis (GALLANT; LAMBERT; VANSTONE, 2000; MENEZES; OKAMOTO; VANSTONE, 1993; SATOH; ARAKI, 1998; SMART, 2002; GAUDRY; HESS; SMART, 2002; GALBRAITH; HESS; SMART, 2002; GALBRAITH; SMART, 1999; JACOBSON; MENEZES; STEIN, 2001; MENEZES; QU, 2001; WIENER; ZUCCHERATO, 1999).

Dois desses métodos dedicados para resolver o DLP em grupos elípticos são particularmente poderosos: o ataque descendente de Weil (GALBRAITH; HESS; SMART,

2002; GALBRAITH; SMART, 1999; GAUDRY; HESS; SMART, 2002), e o ataque Menezes-Okamoto-Vanstone (MENEZES; OKAMOTO; VANSTONE, 1993) (ou a generalização conhecida como ataque Frey-Rück(FREY; RÜCK, 1994)).

O ataque descendente de Weil mapeia o grupo dos pontos de uma curva elíptica a um subgrupo de uma curva hiperelíptica de gênero topológico mais alto, onde o DLP é solúvel por algoritmos subexponenciais. Este ataque não se aplica se o grau de extensão m do corpo finito $\mathbb{F}_{p^m}$ sobre o qual a curva é definida for unitário ou primo.

O ataque Menezes-Okamoto-Vanstone (MOV) ou Frey-Rück (FR) mapeia o DLP sobre uma curva elíptica E a um problema análogo no grupo multiplicativo de um corpo finito (a saber, uma extensão $\mathbb{F}_{q^k}$ do corpo $\mathbb{F}_q$ sobre o qual a curva é definida). A resistência contra esse ataque pode ser quantificada pelo grau de imersão (cfr. seção 5.3.4) k da curva: um sistema criptográfico elíptico resiste ao ataque MOV-FR se o DLP em $\mathbb{F}_{q^k}$ for computacionalmente inviável.

Finalmente, cabe lembrar que, embora a dificuldade de calcular um logaritmo discreto seja sempre um limitante superior de segurança para os sistemas criptográficos baseados no DLP, certos esquemas podem ser suscetíveis a outros tipos de ataque. Entre eles encontram-se o ataque de subgrupo reduzido (IEEE P1363 Working Group, 2000, seção D.5.1.6), ataque de Vaudenay contra DSA (VAUDENAY, 1996), e ataque de Bleichenbacher (BLEICHENBACHER, 2001). Não nos deteremos nessas considerações de segurança, mas qualquer implementação efetiva de algoritmos criptográficos baseados no DLP necessariamente precisará levá-los em conta.

6.4 Assinaturas digitais

Uma *assinatura digital* (MENEZES; OORSCHOT; VANSTONE, 1999, seção 1.6) é um valor inequivocamente associado a uma informação digital e ao seu originador, proprietário ou responsável (chamado genericamente *signatário*), e visa a garantir a in-

tegridade, autenticidade e irretratabilidade dessa informação através de um mecanismo público de verificação.

Tipicamente, um algoritmo de assinatura digital calcula um código digital de integridade (*message digest*) dos dados assinados através de uma função de *hash*, e então aplica uma cifra assimétrica para encriptar esse código com a chave privada do signatário; a assinatura digital é o próprio código de integridade cifrado. Na fase de verificação, a função de *hash* é calculada sobre os dados recebidos e a assinatura associada é decifrada com a chave pública do signatário (recuperando, em princípio, o código de integridade originalmente calculado pelo signatário). Os resultados são comparados e devem coincidir, exceto se os dados recebidos ou a assinatura associada estiverem corrompidos ou tiverem sido forjados.

6.4.1 Assinaturas com apêndice e com recuperação de mensagem

Há basicamente duas maneiras de vincular uma assinatura digital aos dados assinados: anexar a assinatura aos dados, ou embutir os dados na assinatura. O primeiro caso, onde os dados e a assinatura estão meramente justapostos, caracteriza as *assinaturas com apêndice* ou *unidirecionais*. Para este tipo de assinatura, sempre é necessário transmitir os dados juntamente com a assinatura para possibilitar ulterior verificação. O segundo caso define as *assinaturas com recuperação de mensagem* ou *bidirecionais*, e a verificação pode ser efetuada por simples comparação (se a mensagem original estiver aposta à assinatura) ou por validação de redundância anexa (tag). As figuras 7, 8 e 9 comparam os dois tipos de assinatura.

6.5 Algoritmos de assinatura

Concentraremos nossa exposição em algoritmos de assinatura baseados no DLP, que operam num grupo cíclico $\mathbb{G}$ de ordem prima r. No que segue, G denota um

Figura 7: Assinaturas com apêndice.

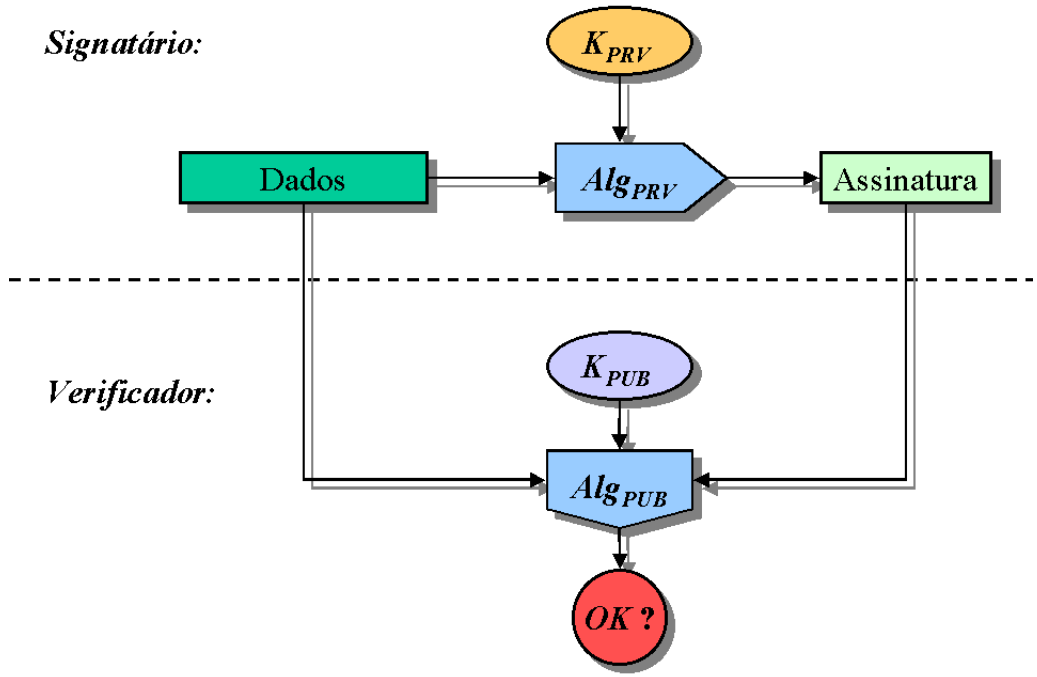


Figura 8: Assinaturas com recuperação e comparação de mensagem.

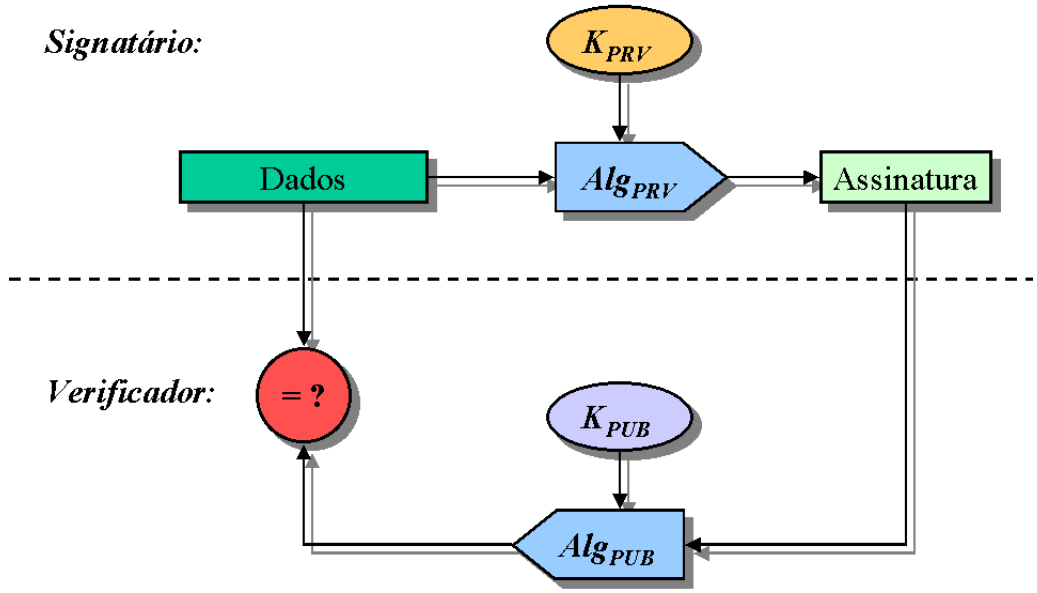
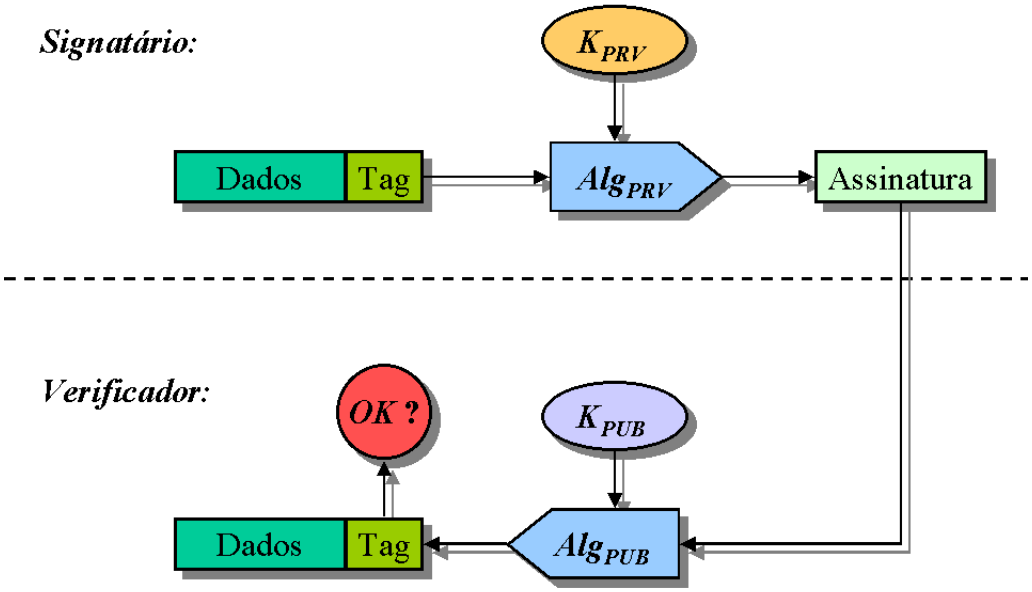


Figura 9: Assinaturas com recuperação de mensagem e validação de redundância anexa (tag).



gerador de $\mathbb{G}$, onde a operação de grupo é escrita aditivamente. Esta notação é natural para grupos elípticos, mas em nada afeta a implementação em outros grupos onde o DLP seja intratável, como o grupo multiplicativo $\mathbb{Z}_p^*$ dos inteiros módulo um número primo p, ou o grupo XTR (LENSTRA; VERHEUL, 2000).

6.5.1 DSA

O algoritmo DSA (*Digital Signature Algorithm*), quer em sua forma convencional (baseada em aritmética modular), quer em sua variante elíptica, foi proposto e adotado como padrão para assinaturas digitais pelo governo americano (NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY – NIST, 2000).

A chave privada do signatário é um inteiro x escolhido aleatoriamente no intervalo $[1 \ldots q-1]$, e a chave pública correspondente é o elemento $Y = [x]G$ do grupo $\mathbb{G}$.

Para assinar uma mensagem M, o signatário gera aleatoriamente um valor inteiro privado k no intervalo $[1 \ldots q-1]$, calcula o valor público correspondente $R = [k]G$ e mapeia o resultado para um inteiro r no intervalo $[1 \ldots q-1]$. O modo exato de mapeamento depende da natureza do grupo específico $\mathbb{G}$. Por exemplo, tratando-se de um subgrupo de ordem q do grupo multiplicativo $\mathbb{Z}_p^*$ (de modo que R é um inteiro no intervalo $[1 \ldots p-1]$), adota-se $r = R \bmod q$, ao passo que, numa curva elíptica sobre $\mathbb{Z}_p$, r é a abscissa do ponto R, reduzida módulo q. Em qualquer caso, se o valor obtido de r for nulo, escolhe-se um novo valor k. Em seguida, o signatário calcula o código de integridade $h = H(M)$ de M (mais exatamente, o código de integridade é mapeado para um inteiro no intervalo $[0 \ldots q-1]$: $h = H(M) \bmod q$) e obtém o valor $s = k^{-1}(h + rx) \bmod q$; se s for nulo (isto é, se $h + rx \equiv 0 \pmod p$), escolhe-se um novo valor de k. A assinatura de M é o par (r, s).

Para verificar uma assinatura, o verificador recalcula h a partir da mensagem recebida, inverte o componente s da assinatura associada obtendo $w = s^{-1} \bmod q$, calcula o elemento $U = [wh]G + [wr]Y$ do grupo $\mathbb{G}$ e mapeia-o para um inteiro u no intervalo $[1 \ldots q-1]$ segundo o mesmo procedimento convencionado para o signatário derivar r a partir de R. Finalmente, o verificador testa se $u = r$, em cujo evento (e somente nele) aceitará a assinatura como válida. A razão é que, pela definição, $w = k(h + rx)^{-1} \bmod q$, donde $k = w(h + rx) = (wh) + (wr)x \bmod q$, e conseqüentemente $R = [k]G = [wh]G + [wr]([x]G) = U$, assumindo que a assinatura foi realmente gerada com a chave privada x correspondente à chave pública Y.

As restrições $r \neq 0$ e $s \neq 0$ justificam-se porque, no primeiro caso, a assinatura não dependeria da chave privada (note-se a multiplicação de r por x), e no segundo, a assinatura não seria verificável (o valor $s^{-1} \bmod q$ é usado na equação de verificação).

6.5.2 Schnorr

Assume-se para a assinatura de Schnorr (SCHNORR, 1991) a existência de uma função de *hash* com chave (isto é, um código de autenticação de mensagem) $H(K, M)$.

A chave privada do signatário é um inteiro x escolhido aleatoriamente no intervalo $[1 \ldots q-1]$, e a chave pública correspondente[1] é o elemento $Y = [x]G$ do grupo $\mathbb{G}$.

Para assinar uma mensagem M, o signatário gera aleatoriamente um valor inteiro privado k no intervalo $[1 \ldots q-1]$, calcula o valor público correspondente $R = [k]G$ e mapeia o resultado para uma seqüência de bits r. Por exemplo, num subgrupo de ordem q do grupo multiplicativo $\mathbb{Z}_p^*$, r pode ser a representação binária de R (um inteiro no intervalo $[1 \ldots p-1]$, sem necessidade de redução módulo q, como ocorre no DSA), ao passo que, numa curva elíptica sobre $\mathbb{Z}_p$, r pode ser a representação binária da abscissa do ponto R. Em seguida, o signatário calcula o código de autenticação de mensagem $h = H(r, M)$ de M sob a chave r (mais exatamente, o código de autenticação de mensagem é mapeado para um inteiro no intervalo $[1 \ldots q-1]$: $h = H(r, M) \bmod q$); se h for nulo, escolhe-se um novo valor de k. Finalmente, o signatário obtém o valor $z = (k - hx) \bmod q$. A assinatura é o par (h, z).

Para verificar uma assinatura, o verificador calcula o elemento $U = [z]G + [h]Y$ do grupo $\mathbb{G}$ e mapeia-o para uma seqüência de bits u segundo o mesmo procedimento convencionado para o signatário derivar r a partir de R. Em seguida, o verificador recalcula $t = H(u, M)$ a partir da mensagem recebida, e finalmente testa se $t = h$, em cujo evento (e somente nele) aceitará a assinatura como válida. A razão é que, pela definição, $k = (z + hx) \bmod q$, donde $R = [k]G = [z]G + [h]([x]G) = U$, assumindo que a assinatura foi realmente gerada com a chave privada x correspondente à chave pública Y.

[1] É comum encontrar a convenção $Y = -[x]G$, que leva a uma equação de assinatura com uma adição em vez de uma subtração. Este detalhe, porém, é irrelevante para o funcionamento e a segurança do algoritmo.

A restrição $h \bmod q \neq 0$ justifica-se porque, sem ela, a assinatura não dependeria da chave privada (note-se a multiplicação de h por x).

6.5.3 Nyberg-Rueppel e Pintsov-Vanstone

Os algoritmos de assinatura de Nyberg-Rueppel (NYBERG; RUEPPEL, 1993) e de Pintsov-Vanstone (PINTSOV; VANSTONE, 2001) derivam do algoritmo de Schnorr, mas possuem a propriedade de *recuperação de mensagem*. A diferença que proporciona esta característica é a substituição, relativa à assinatura de Schnorr, do código de autenticação de mensagem $H(K, M)$ por uma cifra simétrica $E(K, M)$. Assume-se que M possui alguma redundância verificável; por exemplo, M pode ser a concatenação dos dados reais e um *hash* criptograficamente seguro calculado sobre esses dados (variante Pintsov-Vanstone), ou consistir apenas desse *hash*, com os dados efetivos sendo transmitidos à parte (variante Nyberg-Rueppel).

Assim, adotando a mesma notação do algoritmo de Schnorr, a assinatura gerada pelo signatário passa a ser o par (C, z') onde $C = E(r, M)$, $z' = (k - h'x) \bmod q$, e $h' = H(C)$.

O verificador, por sua vez, após obter a seqüência de bits u como numa assinatura Schnorr após calcular $h' = H(C)$, utiliza-a para decifrar C recuperando o valor $M' = E^{-1}(u, C)$, e testar a redundância supostamente presente na mensagem. A assinatura é aceita, e a mensagem em si é considerada como recuperada íntegra e autenticamente ($M' = M$), se, e somente se, o teste de redundância for bem sucedido.

Dados semânticos em qualquer quantidade podem ser incluídos na assinatura, particularmente na forma de marcas d'água visualmente significativas, como se vê na seção 4.6. A única restrição aqui é o próprio espaço disponível para armazenar a assinatura completa na imagem sem acarretar deterioração apreciável na qualidade visual da imagem marcada.

6.5.4 BLS

Assinaturas Boneh-Lynn-Shacham, ou BLS (BONEH; LYNN; SHACHAM, 2002), são um desenvolvimento recente em tecnologia de assinaturas digitais. Um nome mais apropriado poderia ser Okamoto-Pointcheval-Boneh-Lynn-Shacham (OPBLS), uma vez que esse tipo de assinatura já estava essencialmente definido por esses dois primeiros autores em (OKAMOTO; POINTCHEVAL, 2001). Contudo, o mérito de propor um esquema efetivamente compacto é dos três últimos autores, e o nome BLS é agora corrente.

Para expor mais detalhadamente os aperfeiçoamentos e variantes que desenvolvemos para este algoritmo, devotaremos a ele o capítulo 7.

6.5.5 RSA

Embora nosso foco esteja em assinaturas baseadas no DLP, descreveremos resumidamente, para possibilitar comparações, aquele que é talvez o algoritmo mais amplamente utilizado na atualidade, o algoritmo RSA (RIVEST; SHAMIR; ADLEMAN, 1978). Assim chamado por causa das iniciais dos pesquisadores que o inventaram (Rivest, Shamir e Adleman), o RSA pode ser essencialmente descrito de maneira muito simples.

Sejam p e q dois números primos distintos de tamanhos aproximadamente iguais, seja $n = pq$, e seja e um inteiro inversível módulo $(p-1)(q-1)$, com inverso $d \equiv e^{-1} \bmod (p-1)(q-1)$ (isto é, $ed \equiv 1 \pmod{(p-1)(q-1)}$). A chave pública é o par (e, n), e a chave privada é o inteiro d (os primos p e q são também mantidos secretos, e podem até ser descartados, pois o conhecimento deles não é essencial para operações RSA). Seja $M \in \mathbb{Z}_n$ a mensagem a ser assinada. Uma assinatura RSA para M é definida como $C = M^d \bmod n$. A verificação da assinatura procede com a recuperação de M a partir de C: $M = C^e \bmod n$, e comparação com a mensagem M'

Tabela 4: Comparação ilustrativa de tamanhos de chaves e assinaturas

algoritmo	chave privada	chave pública	assinatura
RSA	1024 bits	1024 bits	1024 bits
DSA	160 bits	1024 bits	320 bits
BLS	160 bits	160 bits	160 bits

recebida juntamente com assinatura; esta só é aceita se $M = M'$.

Na prática, uma mensagem precisa ser processada com uma função de *hash* e formatada adequadamente antes de ser efetivamente assinada. A discussão de técnicas seguras de formatação, porém, foge ao escopo desta discussão.

Para fins de comparação, a tabela 4 traça um paralelo entre os tamanhos de chaves e assinaturas para diversos algoritmos, todos com nível de segurança conjecturalmente equivalente.

7 ASSINATURAS DIGITAIS COMPACTAS

Neste capítulo descreveremos a teoria por trás das assinaturas compactas Boneh-Lynn-Shacham, ou BLS (BONEH; LYNN; SHACHAM, 2002). Embora se conheçam outros tipos de algoritmos que produzem assinaturas mais curtas (PATARIN; COURTOIS; GOUBIN, 2000; COURTOIS; FINIASZ; SENDRIER, 2002), o esquema BLS é muito mais rápido na prática, de duas a três ordens de grandeza.

Contrariamente a todos os outros esquemas conhecidos e vistos no capítulo anterior, o algoritmo BLS é estritamente elíptico: não existe um análogo para assinaturas baseadas no problema do logaritmo discreto convencional. Para entendermos o porquê, é necessária uma consideração mais detalhada dos problemas matemáticos em que se apóiam o algoritmo BLS e outros relacionados.

7.1 Digressão sobre problemas do tipo Diffie-Hellman

Sistemas criptográficos designados genericamente como "baseados no logaritmo discreto" (seção 6.3.2) fundamentam-se, mais precisamente, em diversos problemas matemáticos distintos, mas estreitamente relacionados, definidos num grupo $\mathbb{G}$. São eles:

- *Problema Diffie-Hellman Computacional (CDHP)* Dados P, $[a]P$, $[b]P \in \mathbb{G}$, calcular $[ab]P$.
- *Problema Diffie-Hellman Decisional (DDHP)* Dados P, $[a]P$, $[b]P$, $[c]P \in \mathbb{G}$

onde a ordem de P é r, decidir se $c \equiv ab \pmod{r}$.

Obviamente, se o DLP é tratável, então o CDHP também é tratável, e se o CDHP é tratável, então o DDHP também é tratável.

Existem grupos onde o DDHP pode ser eficientemente resolvido. Este é o caso de grupos onde se pode definir a noção de *emparelhamento*:

Definição 16. *Sejam $\mathbb{G}_1$ e $\mathbb{G}_2$ grupos escritos aditivamente, $\mathbb{G}_3$ um grupo escrito multiplicativamente. Um* emparelhamento *é uma função bilinear, não degenerada, efetivamente computável $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. Em outras palavras, as seguintes propriedades são válidas para a função e:*

- (Bilinearidade) $e(P_1+P_2, Q) = e(P_1, Q)\cdot e(P_2, Q)$ *e* $e(P, Q_1+Q_2) = e(P, Q_1)\cdot e(P, Q_2)$ *para todo* $P, P_1, P_2 \in \mathbb{G}_1$ *e todo* $Q, Q_1, Q_2 \in \mathbb{G}_2$. *Segue daí que* $e([a]P, [b]Q) = e(P, Q)^{ab}$ *para todo* $a, b \in \mathbb{Z}$.

- (Não-degeneração) *Se* $e(P, Q) = 1$ *para todo* $Q \in \mathbb{G}_2$, *então* $P = O$. *Alternativamente, para todo* $P \neq O$ *existe* $Q \in \mathbb{G}_2$ *tal que* $e(P, Q) \neq 1$.

- (Computabilidade) *O cálculo de* $e(P, Q)$ *é computacionalmente tratável (tempo polinomial).*

Suponhamos que exista um emparelhamento $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}'$. Nesse caso, dados P, $[a]P$, $[b]P$, $[c]P \in \mathbb{G}$ onde a ordem de P é r, claramente $c \equiv ab \pmod{r} \Leftrightarrow e([a]P, [b]P) = e(P, [c]P)$. Para alguns desses grupos, conjectura-se que o CDHP permaneça intratável.

Seja $\mathbb{G}$ um grupo onde o problema Diffie-Hellman decisional é tratável. As observações acima levam às definições dos seguintes problemas:

- *Problema Diffie-Hellman lacunar (GDHP)* — Resolver o CDHP em $\mathbb{G}$, possivelmente com o auxílio do oráculo que resolve o DDHP em $\mathbb{G}$.

- *Problema Diffie-Hellman bilinear (BDHP)* — Suponhamos que exista um emparelhamento $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}'$. Dados P, $[a]P$, $[b]P$, $[c]P \in \mathbb{G}$, calcular $e(P,P)^{abc} \in \mathbb{G}'$.

Para cada problema acima existe um *co-problema* envolvendo dois grupos $\mathbb{G}_1$ e $\mathbb{G}_2$ em vez de um único grupo:

- *Problema co-Diffie-Hellman decisional (co-DDHP)* — Dados $P, [a]P \in \mathbb{G}_1$ e $Q, [b]Q \in \mathbb{G}_2$ onde a ordem de P e Q é r, decidir se $a \equiv b \pmod{r}$.

- *Problema co-Diffie-Hellman computacional (co-CDHP)* — Dados $P, [a]P \in \mathbb{G}_1$ e $Q \in \mathbb{G}_2$, calcular $[a]Q$.

- *Problema co-Diffie-Hellman lacunar (co-GDHP)* — Resolver o co-CDHP nos grupos $\mathbb{G}_1$, $\mathbb{G}_2$, possivelmente com o auxílio do oráculo que resolve o co-DDHP nesses grupos.

- *Problema co-Diffie-Hellman bilinear (co-BDHP)* — Suponhamos que exista um emparelhamento $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. Dados $P, [a]P, [b]P \in \mathbb{G}_1$ e $Q \in \mathbb{G}_2$, calcular $e(P,Q)^{ab}$.

Notamos que o co-DDHP pode ser resolvido em grupos onde existe um emparelhamento $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$, uma vez que, dados $P, [a]P \in \mathbb{G}_1$ e $Q, [b]Q \in \mathbb{G}_2$ onde a ordem de P e Q é r, $a \equiv b \pmod{r} \Leftrightarrow e([a]P, Q) = e(P, [b]Q)$.

A importância dos co-problemas está na possibilidade de resolver o co-GDHP num par de grupos quando não for possível resolver o GDHP num único grupo, ou definir esquemas baseados na dificuldade do co-BDHP num par de grupos quando não for possível encontrar um grupo adequado onde o BDHP seja intratável.

A segurança de alguns sistemas criptográficos baseados em emparelhamentos depende da dificuldade do BDHP (ou co-BDHP), como o esquema Boneh-Franklin (IBE)

de cifração baseada em identidade (BONEH; FRANKLIN, 2003); outros dependem do GDHP (ou co-GDHP), como as assinaturas compactas BLS (BONEH; LYNN; SHACHAM, 2002).

A intensa pesquisa nesse ramo da criptografia resultou em novas análises das propriedades associadas de segurança, bem como a extensões para curvas algébricas mais gerais, como curvas hiperelípticas e superelípticas (GALBRAITH, 2002; RUBIN; SILVERBERG, 2002).

7.1.1 Emparelhamentos

Dois emparelhamentos são especialmente adequados para abordar a resolução do DDHP: o emparelhamento de Weil (JOUX; NGUYEN, 2001; MENEZES; OKAMOTO; VANSTONE, 1993; MILLER, 1986; SILVERMAN, 1986) e o emparelhamento de Tate (FREY; MÜLLER; RÜCK, 1999; GALBRAITH; HARRISON; SOLDERA, 2002). Destes, o mais simples e flexível é o emparelhamento de Tate, e a ele dedicaremos nossa atenção[1].

Definição 17. *Seja $E(\mathbb{F}_q)$ uma curva elíptica contendo um subgrupo de ordem prima r e grau de imersão k, e seja ℓ um múltiplo de r que divide $q^k - 1$. Sejam $P \in E(\mathbb{F}_q)[\ell]$, $Q \in E(\mathbb{F}_{q^k})$, f_ℓ uma função racional cujo divisor satisfaça $(f_\ell) = \ell(P) - \ell(O)$ e $\mathcal{D} \sim (Q) - (O)$ um divisor com suporte disjunto do suporte de f_ℓ. O* emparelhamento de Tate *de ordem ℓ é a função racional $e_\ell : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k}) \to \mathbb{F}_{q^k}^*$ definida por $e_\ell(P, Q) \equiv f_\ell(\mathcal{D})^{(q^k-1)/\ell}$.*

Note-se que, como $P \in E(\mathbb{F}_q)$, f_ℓ é uma função racional com coeficientes em $\mathbb{F}_q$.

O emparelhamento de Tate satisfaz as seguintes propriedades (FREY; RÜCK, 1994):

[1] *Nossa definição do emparelhamento de Tate é um pouco diferente daquelas dadas em (FREY; MÜLLER; RÜCK, 1999; GALBRAITH, 2002), mas captura as propriedades essenciais necessárias em sistemas criptográficos baseados no problema GDH.*

- *(Bilinearidade)* $e_\ell(P_1 + P_2, Q) = e_\ell(P_1, Q) \cdot e_\ell(P_2, Q)$ e $e_\ell(P, Q_1 + Q_2) = e_\ell(P, Q_1) \cdot e_\ell(P, Q_2)$ para todo $P, P_1, P_2 \in E(\mathbb{F}_q)[\ell]$ e todo $Q, Q_1, Q_2 \in E(\mathbb{F}_{q^k})$. Segue daí que $e_\ell([a]P, Q) = e_\ell(P, [a]Q) = e_\ell(P, Q)^a$ para todo $a \in \mathbb{Z}$.

- *(Não-degeneração)* Se $e_\ell(P, Q) = 1$ para todo $Q \in E(\mathbb{F}_{q^k})$, então $P = O$. Alternativamente, para todo $P \neq O$ existe $Q \in E(\mathbb{F}_{q^k})$ tal que $e_\ell(P, Q) \neq 1$.

- *(Compatibilidade)* Se $\ell = h\ell'$, $P \in E(\mathbb{F}_q)[\ell]$, e $Q \in E(\mathbb{F}_{q^k})$, então $e_\ell(P, Q) = e_{\ell'}([h]P, Q) = e_{\ell'}(P, Q)^h$, isto é, $f_\ell(\mathcal{D})^{(q^k-1)/\ell} = f_{\ell'}(\mathcal{D})^{(q^k-1)/\ell'}$.

A propriedade de compatibilidade permite escrever simplesmente $e(P, Q)$, subentendendo-se $f_\ell(\mathcal{D})^{(q^k-1)/\ell}$ para qualquer múltiplo ℓ de r que divida $q^k - 1$.

De modo geral, é inviável representar $e(P, Q)$ diretamente como a razão de dois polinômios. Contudo, calculando o valor do emparelhamento sob demanda mantém a complexidade computacional do emparelhamento de Tate (ou de Weil) igual à de uma multiplicação por escalar sobre a extensão da curva E para o corpo finito $\mathbb{F}_{q^k}$. Esta é a idéia básica do algoritmo de Miller (cfr. seção 8.6).

Recentemente, um terceiro emparelhamento foi proposto por Boneh, Mironov e Shoup (BONEH; MIRONOV; SHOUP, 2003), sem qualquer relação com os emparelhamentos de Tate e Weil – na verdade, este emparelhamento não é sequer baseado em aritmética elíptica e no DLP, mas em aritmética modular e no problema da fatoração inteira. Trata-se da função $r : \mathbb{Z}_n^* \times \mathbb{Z}_{\phi(n)}^+ \to \mathbb{Z}_n^*$ dada por $r(g, h) \equiv g^h \bmod n$. Conjectura-se que este emparelhamento seja seguro dentro da hipótese RSA forte, isto é, desde que seja computacionalmente inviável encontrar, para um dado $x \in \mathbb{Z}_n^*$ aleatório, valores $y \in \mathbb{Z}_n^*$ e $r > 1$ tais que $y^z = x$.

Sejam $P \in E(\mathbb{F}_q)[r]$ e $Q \in E(\mathbb{F}_{q^k})$ pontos linearmente independentes, de modo que o emparelhamento de Tate satisfaz $e(P, Q) \neq 1$. O emparelhamento de Tate resolve o co-DDHP definido por $P, [a]P, Q, [b]Q$. Suponhamos que exista um mapa de distorção eficientemente calculável $\phi : \langle P \rangle \to \langle Q \rangle$. Podemos definir o emparelha-

mento derivado $\hat{e} : \langle P \rangle \times \langle P \rangle \rightarrow \mathbb{F}_{q^k}^*$ tal que $\hat{e}(P, P) = e(P, \phi(P))$. Esse emparelhamento resolve o DDHP definido por $P, [a]P, [b]P, [c]P$.

Infelizmente, mapas de distorção só existem para curvas supersingulares. Mesmo assim, podemos definir outro emparelhamento derivado $\bar{e} : \langle Q \rangle \times \langle Q \rangle \rightarrow \mathbb{F}_{q^k}^*$ tal que $\hat{e}(Q, Q) = e(\mathrm{tr}(Q), [k]Q - \mathrm{tr}(Q))$. Este outro emparelhamento derivado resolve o DDHP definido por $Q, [a]Q, [b]Q, [c]Q$.

7.2 O algoritmo de assinatura digital BLS

Apresentaremos duas variantes do algoritmo BLS. A primeira, baseada na dificuldade do GDHP, é otimizada para curvas supersingulares e emprega um mapa de distorção ϕ na curva utilizada (cfr. tabela 2 para as curvas em questão e seus respectivos mapas de distorção). A segunda, baseada no co-GDHP, é mais geral e pode ser utilizada com qualquer grupo onde o problema DDHP seja viável e o problema CDHP intratável. Assumiremos nos dois casos que a curva utilizada tenha grau de imersão k. A verificação baseia-se no emparelhamento de Tate $e(P, Q)$. O algoritmo também necessita uma função de *hash* h que mapeia mensagens a pontos da curva. Uma construção apropriada em termos de uma função de *hash* convencional é descrita adiante.

7.2.1 Variante supersingular

7.2.1.1 Geração de par de chaves

Seja $E(\mathbb{F}_q)$ uma das curvas supersingulares da tabela 2, seja r a ordem prima do subgrupo com grau de imersão k, e seja $P \in E(\mathbb{F}_q)[r]$. A chave privada de assinatura é um elemento secreto, estatisticamente único e uniformemente distribuído $s \in \mathbb{Z}_r^*$, e a chave pública correspondente é a quádrupla $(E(\mathbb{F}_q), r, P, V)$ onde $V = [s]P$.

7.2.1.2 Assinatura

Para assinar uma mensagem $M \in \{0,1\}^*$, mapeia-se M a um ponto $P_M = h(M) \in \langle P \rangle$. Seja $S_M = [s]P_M = (x_S, y_S)$. A assinatura σ é a abscissa x_S de S_M. Note-se que $\sigma \in \mathbb{F}_q$.

7.2.1.3 Verificação

Seja ϕ o mapa de distorção associado à curva $E(\mathbb{F}_q)$. Dado $P \in E(\mathbb{F}_q)$ de ordem r, o ponto $Q = \phi(P) \in E(\mathbb{F}_{q^k})$ tem a mesma ordem r e é linearmente independente de P.

Dada a chave pública $(E(\mathbb{F}_q), r, P, V)$, uma mensagem M e uma assinatura alegada σ, executam-se os seguintes passos:

1. Encontrar um ponto $S \in E(\mathbb{F}_q)$ de ordem r cuja abscissa seja σ e cuja ordenada seja y para algum elemento $y \in \mathbb{F}_q$. Se nenhum ponto assim existir, rejeitar a assinatura.

2. Calcular os emparelhamentos de Tate $u \leftarrow e(P, \phi(S))$ e $v \leftarrow e(V, \phi(h(M)))$.

3. Aceitar assinatura se, e somente se, $u = v$ ou $u^{-1} = v$.

Note-se que (σ, y) e $(\sigma, -y)$ são ambos pontos em $E(\mathbb{F}_q)$ com abscissa σ. Qualquer um desses dois pontos pode ser tomado como o ponto S_M usado para produzir a assinatura. De fato, da relação $(\sigma, -y) = -(\sigma, y)$ resulta que $e(P, \phi(-S)) = e(P, \phi(S))^{-1}$. Portanto, $u = v$ testa se $(P, V, h(M), S)$ é uma quádrupla de Diffie-Hellman, enquanto que $u^{-1} = v$ perfaz o mesmo teste para a quádrupla $(P, V, h(M), -S)$. O fato que a assinatura consiste exclusivamente na abscissa se S causa uma degradação (insubstancial) na capacidade de detectar tentativas de fraude, já que S e $-S$ são indistinguíveis do ponto de vista do algoritmo de verificação.

7.2.2 Variante geral

7.2.2.1 Geração de par de chaves

Seja $E(\mathbb{F}_q)$ uma curva elíptica contendo um subgrupo de ordem prima r e grau de imersão k. Sejam $P \in E(\mathbb{F}_q)[r]$ e $Q \in E(\mathbb{F}_{q^k})$ pontos[2] linearmente independentes de ordem r. A chave privada de assinatura é um elemento secreto, estatisticamente único e uniformemente distribuído $s \in \mathbb{Z}_r^*$, e a chave pública correspondente é a quádrupla $(E(\mathbb{F}_q), r, Q, V)$ onde $V = [s]Q$.

7.2.2.2 Assinatura

Para assinar uma mensagem $M \in \{0,1\}^*$, mapeia-se M a um ponto $P_M = h(M) \in \langle P \rangle$. Seja $S_M = [s]P_M = (x_S, y_S)$. A assinatura σ é a abscissa x_S de S_M. Note-se que $\sigma \in \mathbb{F}_q$.

7.2.2.3 Verificação

Dada a chave pública $(E(\mathbb{F}_q), r, Q, V)$, uma mensagem M e uma assinatura alegada σ, executam-se os seguintes passos:

1. Encontrar um ponto $S \in E(\mathbb{F}_q)$ de ordem r cuja abscissa seja σ e cuja ordenada seja y para algum elemento $y \in \mathbb{F}_q$. Se nenhum ponto assim existir, rejeitar a assinatura.

2. Calcular os emparelhamentos de Tate $u \leftarrow e(Q, S)$ e $v \leftarrow e(V, h(M))$.

3. Aceitar assinatura se, e somente se, $u = v$ ou $u^{-1} = v$.

Por raciocínio análogo ao caso supersingular, o algoritmo de verificação testa se uma das quádrupla $(Q, V, h(M), S)$ ou $(Q, V, h(M), -S)$ é co-Diffie-Hellman.

[2] O ponto P só é usado implicitamente.

É interessante notar que, na variante geral, a chave pública V é um elemento de $E(\mathbb{F}_{q^k})$, e portanto k vezes maior que no caso supersingular. A assinatura, porém, continua sendo um elemento de $E(\mathbb{F}_q)$.

7.3 *Hash* sobre curvas

Para completar os esquemas acima é necessário especificar uma função de *hash* para mapear mensagens em elementos do grupo $\langle P \rangle$, de tal maneira que o logaritmo discreto do valor de *hash* permaneça desconhecido. No que segue, assumiremos por simplicidade que $\langle P \rangle$ coincide com a própria curva, isto é, que a ordem desse grupo é prima; o caso de um subgrupo estrito é fácil de derivar, está explicado com detalhes em (BONEH; LYNN; SHACHAM, 2002, seção 3.3), e não contribui substancialmente para a flexibilidade do sistema (pelo contrário, demanda cuidados adicionais para não introduzir vulnerabilidades, como ataques de subgrupo reduzido).

A função *Map2Group* proposta pelos autores do algoritmo BLS mapeia uma mensagem M a um ponto da curva (x, y) usando uma função de *hash* convencional $h : \mathbb{Z} \times \{0,1\}^* \to \{0,1\}^n$. Essa construção é probabilística, no sentido de que existe uma probabilidade não nula (mas arbitrariamente pequena) de que uma mensagem não possa ser mapeada a um ponto da curva.

Seja $y^2 = f(x)$ a equação da curva elíptica utilizada sobre $\mathbb{F}_q$, onde $f(x)$ é um polinômio cúbico[3], e seja I um parâmetro inteiro pequeno. Calcula-se *Map2Group* como segue:

Algoritmo *Map2Group*$_h(M)$

1. Inicializar um contador $i = 0$.

[3]Curvas em característica 2 têm uma equação ligeiramente diferente que, estritamente falando, exige um tratamento distinto, mas muito semelhante ao exposto aqui. Por simplicidade, consideraremos apenas características ímpares.

2. Mapear o par (i, M) ao par $h(i, M) = (x, t)$ onde $x \in \mathbb{F}_q$ e $t \in \{0, 1\}$.

3. Calcular $u = f(x)$.

4. Resolver a equação quadrática $y^2 = u$ em $\mathbb{F}_q$.

5. Se nenhuma solução for encontrada:

   (a) Se $i < I$, incrementar i e tentar de novo a partir do passo 2.

   (b) Caso contrário, declarar que M não pode ser mapeada num ponto da curva.

6. Caso contrário, usar t para escolher entre as soluções y_0 e y_1, e retornar (x, y_t).

A escolha entre as raízes no último passo acima requer uma convenção determinística arbitrária (e irrelevante para a presente discussão) de rotular as soluções da equação quadrática como y_0 e y_1.

O contador incrementado no passo 5 é limitado pelo valor máximo I. Se esse limite for atingido, a mensagem é declarada não mapeável a um ponto da curva. A probabilidade de falha δ é feita arbitrariamente pequena escolhendo I suficientemente grande, a saber, $I = \lceil \lg \lg(1/\delta) \rceil$.

7.4 Eliminação de ordenada

Não é realmente necessário escolher explicitamente entre as raízes y_0 e y_1 no passo 6 da função *Map2Group*. De fato, t só é necessário para distinguir entre os dois pontos $P_M = (x, y)$ e $-P_M = (x, -y)$, que levam à mesma assinatura porque os pontos $S_M = [s]P_M$ e $-S_M = [s](-P_M)$ compartilham a mesma abscissa x_S. Portanto, o valor de t tem pouca relevância, e uma escolha arbitrária entre y e $-y$ não afeta a segurança do algoritmo.

Da mesma forma, a chave pública $V = (x_V, y_V)$ poderia ser representada apenas por sua abscissa x_V, e a reconstrução de V para uma verificação de assinatura esco-

lheria arbitrariamente entre os pontos (x_V, y_V) e $(x_V, -y_V) = -V$, uma vez que o algoritmo de verificação já compara emparelhamentos inversos.

7.5 Detalhes de utilização

Em certas circunstâncias, pode ser indesejável transmitir o contador completo usado pela função *Map2Group* juntamente com a assinatura. Em teoria, o contador poderia ser completamente omitido da assinatura, mas isto aumentaria a carga imposta ao verificador, a quem já cabe a parte mais pesada do trabalho, com o mesmo esforço que o signatário deve despender. Um balanço simples pode ser anexar apenas alguns bits do contador à assinatura (digamos, os bits menos significativos) e recalcular apenas os bits restantes por ocasião da verificação.

Um aspecto importante a ser levado em conta é que, por serem determinísticas, as assinaturas BLS precisam ser combinadas com o esquema HBC2 através da fórmula de contextos descrita na seção 4.4.2.

Assinaturas BLS são excelentes em situações onde pouco espaço está disponível para inserir uma marca d'água, por exemplo, imagens em níveis de cinza com maior resolução na localização de alterações. Por outro lado, a natureza decisional do algoritmo não deixa margem à anexação de dados semânticos. A opção entre o uso do BLS e a inserção de informação semântica deve basear-se na natureza da aplicação, bem como dos objetivos que se espera alcançar com o uso da marca d'água.

8 ALGORITMOS EFICIENTES PARA SISTEMAS DE EMPARELHAMENTO

Neste capítulo, descrevemos vários algoritmos eficientes para implementar sistemas criptográficos baseados em emparelhamentos, particularmente o emparelhamento de Tate. Nossas técnicas aumentam a velocidade do cálculo de um emparelhamento por um fator de 100 ou mais em relação a métodos previamente conhecidos. Propomos também algoritmos muito mais rápidos de multiplicação por escalar e extração de raízes quadradas, operações essenciais na manipulação de curvas elípticas.

8.1 Extração de raízes quadradas

Pode-se usar a própria equação da curva elíptica $E : y^2 = f(x)$ sobre $\mathbb{F}_q$, onde $f(x)$ é um polinômio cúbico, para obter uma representação compacta de pontos da curva. A idéia é usar um único bit da ordenada y como seletor entre as duas soluções da equação $y^2 = f(x)$ para um dado x. Esta propriedade é importante também para construir funções de *hash* sobre curvas, segundo a técnica de mapear o argumento de *hash* a um elemento $x \in \mathbb{F}_q$, e resolver a equação da curva para obter um ponto completo $(x, y) \in E$. Obviamente, uma operação essencial nessas circunstâncias é o cálculo de raízes quadradas num corpo finito.

Num corpo finito $\mathbb{F}_{p^m}$ onde $p \equiv 3 \pmod 4$ e m é ímpar, o melhor algoritmo previamente conhecido (COHEN, 1993, seção 1.5) para extrair uma raiz quadrada executa em $O(n^3)$ operações em $\mathbb{F}_p$, com $n \equiv m \log p$. Por aquele método, uma solução de

$x^2 = a$ é dada por $x = a^{(p^m+1)/4}$, supondo que a seja um resíduo quadrático. Assim, obtém-se o valor da raiz através de um algoritmo de exponenciação com janela deslizante(MENEZES; OORSCHOT; VANSTONE, 1999, algoritmo 14.82).

Para o nosso método, notamos inicialmente que, se $m = 2k + 1$ para algum k:

$$\frac{p^m + 1}{4} = \frac{p+1}{4}\left[p(p-1)\sum_{i=0}^{k-1}(p^2)^i + 1\right],$$

de modo que

$$a^{(p^m+1)/4} = \left[(a^{\sum_{i=0}^{k-1}(p^2)^i})^{p(p-1)} \cdot a\right]^{(p+1)/4}.$$

Estas relações podem ser verificadas diretamente por indução finita.

A quantidade $a^{\sum_{i=0}^{k-1} u^i}$ onde $u = p^2$ pode ser eficientemente calculado da maneira análoga à inversão de Itoh-Teechai-Tsujii (ITOH; TEECHAI; TSUJII, 1988), tendo por base o mapa de Frobenius em característica p:

$$a^{1+u+\cdots+u^{k-1}} = \begin{cases} (a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}}) \cdot (a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}})^{u^{\lfloor k/2\rfloor}}, & k \text{ par}, \\ ((a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}}) \cdot (a^{1+u+\cdots+u^{\lfloor k/2\rfloor-1}})^{u^{\lfloor k/2\rfloor}})^u \cdot a, & k \text{ ímpar}. \end{cases}$$

Note-se que o mapa de Frobenius é uma operação linear, e quase gratuita numa representação em base normal. Pode-se verificar facilmente por indução que este método requer $\lfloor \lg k \rfloor + \omega(k) - 1$ multiplicações no corpo finito, onde $\omega(k)$ é o peso de Hamming de k. $O(\log p)$ multiplicações são adicionalmente necessárias para completar o cálculo da raiz quadrada devido à multiplicação extra por a e às exponenciações em $p-1$ e $(p+1)/4$, que pode ser efetuadas com um algoritmo convencional de exponenciação[1]. O custo total é $O(n^2 \log n)$ operações em $\mathbb{F}_p$ por extração de raiz quadrada. Se a característica p for fixa e pequena em comparação com m, a complexidade é apenas $O(m^2 \log m)$ operações em $\mathbb{F}_p$.

Relações similares de recorrência valem para o algoritmo de Atkin (IEEE P1363

[1] Se p for grande, pode ser vantajoso calcular z^{p-1} como z^p/z, trocando $O(\log p)$ multiplicações por uma inversão.

Working Group, 2000, seção A.2.5) para o cálculo de raízes quadradas em $\mathbb{F}_{p^m}$ quando $p \equiv 5 \pmod 8$ e m for ímpar, com a mesma complexidade $O(n^2 \log n)$. Infelizmente, o caso geral não é tão fácil. Nem o algoritmo de Tonelli-Shanks (COHEN, 1993) nem o de Lehmer (IEEE P1363 Working Group, 2000, seção A.2.5) beneficiam-se inteiramente da técnica acima, embora aperfeiçoamentos parciais que não alteram a complexidade global sejam possíveis.

Esta técnica de extração de raízes é útil não apenas para sistemas criptográficos baseados em emparelhamentos, mas também para sistemas convencionais (por exemplo, cfr. (KOBLITZ, 1998, seção 6)). Uma extensão do algoritmo básico para raízes quadradas permite o cálculo de raízes superiores (BARRETO; VOLOCH, 2003), mas não entraremos aqui em detalhes por situar-se o assunto fora do escopo desta tese.

8.2 Multiplicação por escalar em característica 3

Em característica 3, a operação de *triplicação de ponto* para as curvas supersingulares $E_{3,b}$ pode ser efetuada em tempo $O(m)$ com representação de base polinomial, ou simplesmente $O(1)$ em hardware usando com representação de base normal. De fato, como o cálculo de um cubo é uma operação linear em característica 3, dado $P = (x, y)$ calcula-se $[3]P = (x_3, y_3)$ com as seguintes fórmulas, que se originam diretamente das fórmulas aritméticas básicas numa curva em característica 3 (seção 5.3):

$$\begin{aligned} x_3 &= (x^3)^3 - b \\ y_3 &= -(y^3)^3 \end{aligned}$$

A linearidade da triplicação de pontos corresponde à linearidade da duplicação de pontos para curvas supersingulares em característica 2, conforme descoberto por Menezes e Vanstone (MENEZES; VANSTONE, 1990), e leva a um algoritmo de mul-

tiplicação por escalar via decomposição ternária muito mais rápido que o método da decomposição binária.

Seja $k = (k_t \dots k_1 k_0)_3$ onde $k_i \in \{-1, 0, 1\}$ e $k_t \neq 0$ a representação ternária algébrica de $k > 0$. O cálculo de $V = [k]P$ procede da seguinte maneira.

Multiplicação por escalar via decomposição ternária:
$V \leftarrow P$ **se** $k_t = 1$, **ou** $V \leftarrow -P$ **se** $k_t = -1$
para $i \leftarrow t-1, t-2, \dots, 1, 0$ **faça** {
    $V \leftarrow [3]V$
    se $k_i = \ \ 1$ **então** $V \leftarrow V + P$
    se $k_i = -1$ **então** $V \leftarrow V - P$
}
devolva V

Obviamente, as mesmas técnicas avançadas de otimização aplicáveis ao método binário podem ser facilmente generalizadas para o método ternário.

8.3 Curvas MNT

Qualquer curva elíptica $E(\mathbb{F}_q)$ de ordem n satisfaz o teorema de Hasse (SILVERMAN, 1986, V.1.1), segundo o qual o traço t do endomorfismo de Frobenius em E, relacionado a q e n pela equação $n = q + 1 - t$, está limitado a $|t| \leqslant 2\sqrt{q}$.

Definição 18. *O* k-ésimo polinômio ciclotômico *(LIDL; NIEDERREITER, 1997, definição 2.44) é o polinômio*

$$\Phi_k(x) = \prod_{d|k} (x^d - 1)^{\mu(k/d)},$$

onde

$$\mu(n) = \begin{cases} 1, & \textit{se } n = 1; \\ (-1)^{\ell}, & \textit{se } n \textit{ é o produto de } \ell \textit{ primos distintos}; \\ 0, & \textit{se } n \textit{ é divisível pelo quadrado de um número primo}. \end{cases}$$

Definição 19. *Seja Φ_k o k-ésimo polinômio ciclotômico para algum $k > 1$. Uma* curva MNT generalizada *(MIYAJI; NAKABAYASHI; TAKANO, 2001) é uma curva elíptica não supersingular construída através do método de multiplicação complexa (cfr. seção 5.3.8), contendo um subgrupo de ordem prima r (i.e. $n = mr$ para algum inteiro m) tal que $r \mid \Phi_k(t-1)$ mas $r \nmid \Phi_i(t-1)$ para todo $0 < i < k$.*

O seguinte lema segue imediatamente da definição:

Lema 4. *Em curvas MNT, o subgrupo de ordem r tem grau de imersão k.*

Demonstração. Mostraremos que $r \mid q^k - 1$ mas $r \nmid q^i - 1$ para todo $0 < i < k$. Claramente, $q \equiv t - 1 \pmod{r}$ devido à relação $n = q + 1 - t$, logo $r \mid q^s - 1$ se, e somente se, $r \mid (t-1)^s - 1$ para todo $s > 0$. É fato conhecido (LIDL; NIEDERREITER, 1997, teorema 2.45(i)) que $x^s - 1 = \prod_{d|s} \Phi_d(x)$. Portanto, uma vez que r é primo, $r \mid x^s - 1$ se, e somente se, $r \mid \Phi_d(x)$ para algum $d \mid s$, e assim $r \mid (t-1)^s - 1$ se, e somente se, $r \mid \Phi_d(t-1)$. Destarte, $r \mid q^s - 1$ se, e somente se, $r \mid \Phi_d(t-1)$ para algum $d \mid s$. Como assumimos pela definição que $r \nmid \Phi_i(t-1)$ para $i < k$, segue que $r \nmid q^i - 1$. Por outro lado, assumimos também que $r \mid \Phi_k(t-1)$, e portanto $r \mid q^k - 1$. □

A equação CM tem a forma:

$$DV^2 = 4q - t^2 = 4mr - (t-2)^2. \tag{8.1}$$

A estratégia para construir curvas MNT parece imediata: escolher k, t e m tais que $\Phi_k(t-1)$ contenha um fator primo r de tamanho adequado e $q = mr + t - 1$

seja primo[2], e então fatorar $4q - t^2$ como DV^2 onde D é livre de quadrados, usando finalmente o método CM para calcular os coeficientes da equação da curva. É fácil ver por que o método CM é necessário: a escolha do grau de imersão k impõe uma relação específica entre r e q (a saber, $r \mid q^k - 1$) e, indiretamente, uma condição particular sobre a ordem da curva, exigindo um método de construção que admita essas restrições de antemão.

Infelizmente, esta estratégia não é prática, porque de maneira geral o discriminante D de multiplicação complexa é grande demais (comparável a q) para que a construção seja exeqüível (o tempo de geração depende polinomialmente de D), e parâmetros criptograficamente significativos teriam $q \approx 2^{160}$, no mínimo.

Miyaji *et al.* (MIYAJI; NAKABAYASHI; TAKANO, 2001) originalmente consideraram apenas $k \in \{3, 4, 6\}$ e $r = \Phi_k(t - 1)$, para os quais a equação 8.1 reduz-se a uma equação de Pell cuja solução é bem conhecida (SMART, 1998). O caso de k arbitrário é muito mais difícil, pois não se conhece método geral para resolver equações diofânticas de grau $\deg(\Phi_k) \geqslant 4$. Mesmo a restrição $k \in \{5, 8, 10, 12\}$, que leva a equações diofânticas quárticas potencialmente solúveis pelo método de Tzanakis (TZANAKIS, 1996), mostrou-se mal sucedida em produzir curvas criptograficamente interessantes para D de tamanho razoável. Contudo, exibiremos uma abordagem diferente que permite construir curvas adequadas para qualquer valor do grau de imersão k. Por simplicidade, assumiremos que q seja primo.

8.4 Construção de curvas MNT generalizadas

Assumamos que D e t sejam escolhidos na equação 8.1 de modo que $\gcd(D, t) = 1$ (caso contrário não há esperança de que q seja primo). Por conveniência, seja $A = 4r$

[2]Uma potência de um número primo seria igualmente aceitável, mas a probabilidade de obter $q = p^u$ para $u > 1$ é negligível.

e $B = (t-2)^2$, de maneira que a equação 8.1 toma a forma:

$$DV^2 \;=\; Am - B.$$

Queremos resolver esta equação diofântica quadrática em m e V, garantindo que $q = mr + t - 1$ seja primo e tentando minimizar o valor de m, que será o cofator da curva construída. A solução geral de equações desse tipo é devida a Gauss (GAUSS, 1965), mas para este caso específico a solução pode ser simplificada, reduzindo-se a três passos.

Primeiro, resolve-se em m e z a equação diofântica linear $Dz = Am - B$. Esta equação possui soluções (IRELAND; ROSEN, 1990) para a escolha feita de t e D se, e somente se, $\gcd(A, D) \mid B$, a saber,

$$\begin{aligned} m_i &= m_0 + i(D/g), \\ z_i &= z_0 + i(A/g), \end{aligned}$$

onde $g = \gcd(A, D)$, $m_0 = (B/g)(A/g)^{-1} \bmod (D/g)$, e $z_0 = (Am_0 - B)/D$ (notar que A/g é inversível módulo D/g, porque $\gcd(A/g, D/g) = 1$). Os valores g, m_0, e z_0 podem ser calculados simultaneamente com o algoritmo estendido de Euclides (MENEZES; OORSCHOT; VANSTONE, 1999, algoritmo 2.107).

A seguir, resolve-se em V e i a equação diofântica quadrática $V^2 = z_0 + i(A/g)$. Isto claramente requer que z_0 seja um resíduo quadrático módulo A/g. Se este for o caso, seja $\{s_u\}$ o conjunto de raízes quadráticas de z_0 módulo A/g, isto é, o conjunto de valores s_u no intervalo $0 \leqslant s_u < A/g$ tais que $s_u^2 \equiv z_0 \pmod{A/g}$. Assim podemos escrever $V = s_u + (A/g)\alpha$ onde α é um inteiro qualquer, implicando $i = (V^2 - z_0)/(A/g) = (A/g)\alpha^2 + 2s_u\alpha + (s_u^2 - z_0)/(A/g)$. Portanto a solução completa

é:

$$\begin{aligned} V_\alpha &= s_u + \alpha(A/g), \\ i_\alpha &= (A/g)\alpha^2 + 2s_u\alpha + (s_u^2 - z_0)/(A/g), \end{aligned}$$

para cada raiz quadrada s_u de z_0 módulo A/g.

Finalmente, tomamos qualquer solução i_α tal que $q = m_{i_\alpha} r + t - 1$ seja primo. Embora possamos variar α para obter essa solução, para tornar a razão entre $\log q$ e $\log r$ a menor possível pode-se preferir restringir a busca por valores de q ao caso $\alpha = 0$, isto é, considerar somente $i_0 = (s_u^2 - z_0)/(A/g)$ e variar apenas t. De posse dos parâmetros q, m, r e t, procede-se ao método CM para obter efetivamente a equação da curva desejada.

Na prática, m tende a ser um valor com tamanho próximo ao de r, isto é, $\log q / \log r \approx 2$. Essas soluções são perfeitamente adequadas para a maioria dos sistemas criptográficos baseados em emparelhamentos; uma exceção notável são precisamente as assinaturas BLS, que exigem $\log q / \log r \approx 1$.

Nosso algoritmo de construção de curvas com grau de imersão arbitrário resolve o problema em aberto mencionado em (BONEH; LYNN; SHACHAM, 2002, seção 3.5). Em contrapartida, ele abre um problema relacionado, qual seja o de construir curvas de ordem *prima* com grau de imersão arbitrário, ou pelo menos curvas onde $\log q / \log r \approx 1$.

Outros algoritmos para a construção de curvas foram recentemente propostos, baseados em princípios semelhantes: o algoritmo Dupont-Enge-Morain (DUPONT; ENGE; MORAIN, 2002) e o algoritmo de Galbraith, por ele chamado algoritmo “folclórico” a despeito da inexistência de um “folclore” sobre o assunto (GALBRAITH, 2003). Alguns indícios recentes atribuem a autoria do método “folclórico” a C. Cocks e R. G. E. Pinch (BREZING; WENG, 2003).

Para a comodidade do leitor interessado em implementar nosso método de construção de curvas, segue um resumo em formato algorítmico, restrito ao caso $\alpha = 0$ (é simples, mas provavelmente desnecessário, modificar o passo 6 de modo a considerar outros valores pequenos de α, digamos, $\alpha < 128$):

Sejam σ o tamanho desejado aproximado (em bits) da ordem prima r, k o grau de imersão selecionado e D o discriminante de multiplicação complexa escolhido.

1. Escolha $t \approx 2^{\sigma/\delta}$ aleatoriamente, onde $\delta \equiv \deg(\Phi_k)$.

2. Calcule $r \leftarrow \Phi_k(t-1)$, $A \leftarrow 4r$, $B \leftarrow (t-2)^2$, e $g \leftarrow \gcd(A, D)$.

3. Verifique se r é primo, se $r \nmid q^d - 1$ para qualquer $d > 0$ tal que $d \mid k$, e se $g \mid B$. Se qualquer uma dessas condições falhar, escolha outro t no passo 1.

4. Resolva em m e z a equação diofântica linear $Dz - Am + B = 0$, isto é, faça $m_0 \leftarrow (B/g)(A/g)^{-1} \bmod (D/g)$, e $z_0 \leftarrow (Am_0 - B)/D$.

5. Calcule todas as raízes quadradas s_u de z_0 módulo A/g. Se z_0 não for um resíduo quadrático módulo A/g (isto é, se não tiver raízes quadradas módulo A/g), escolha outro t no passo 1.

6. Para cada raiz quadrada s_u, faça $i_0 \leftarrow (s_u^2 - z_0)/(A/g)$, $m \leftarrow m_0 + i_0(D/g)$, $n \leftarrow mr$, e $q \leftarrow n + t - 1$. Se q não for primo, reinicie com outro t no passo 1. Caso contrário, construa uma curva sobre $\mathbb{F}_q$ de ordem n e traço de Frobenius t usando o método da multiplicação complexa (BLAKE; SEROUSSI; SMART, 1999, capítulo VIII).

8.5 Seleção de geradores de grupos

Suponhamos que a característica do corpo finito $\mathbb{F}_q$ seja $p > 3$ e que k seja par, com $d \equiv k/2$. Descreveremos agora um método para selecionar geradores de grupos

que torna o cálculo de emparelhamentos mais eficiente, e como bônus melhora o desempenho de operações que não envolvem emparelhamentos diretamente, tais como a geração de pares de chaves.

Seja E uma curva $y^2 = x^3 + ax + b$, e seja $E'(\mathbb{F}_{q^d}) : y^2 = x^3 + v^2ax + v^3b$ o *twist* de E sobre $\mathbb{F}_{q^d}$, onde $v \in \mathbb{F}_{q^d}$ é algum não-resíduo quadrático. Em $\mathbb{F}_{q^k}$, v torna-se um resíduo quadrático, significando que a função $\Psi : (X, Y) \mapsto (v^{-1}X, (v\sqrt{v})^{-1}Y)$ é um isomorfismo mapeando o grupo dos pontos de $E'(\mathbb{F}_{q^d})$ a um subgrupo de $E(\mathbb{F}_{q^k})$.

Seja $Q' = (X, Y) \in E'(\mathbb{F}_{q^d})$, e seja $Q = \Psi(Q') = (v^{-1}X, (v\sqrt{v})^{-1}Y) \in E(\mathbb{F}_{q^k})$. Por construção, a abscissa de Q é um elemento de $\mathbb{F}_{q^d}$, permitindo a técnica de eliminação de denominadores descrita adiante na seção 8.6.2. Esta observação sugere o seguinte algoritmo para a seleção de geradores de grupos:

Algoritmo de seleção de geradores:

1. Escolher aleatoriamente um ponto $P \in E(\mathbb{F}_q)$ de ordem r.
2. Escolher aleatoriamente um ponto $Q' \in E'(\mathbb{F}_{q^d})$ e fazer $Q = \Psi(Q')$.

O domínio do emparelhamento de Tate é então visto como $\langle P\rangle \times \langle Q\rangle$. Pode ser desejável verificar explicitamente $e(P, Q) \neq 1$, mas como isto ocorre com probabilidade esmagadora (a probabilidade de falha é apenas cerca de r^{-1}), em algumas situações pode ser aceitável omitir esta verificação. Note-se que apenas P precisa ser de ordem r.

Operações que não usam o emparelhamento tais como geração de pares de chaves e transmissão de pontos podem ser efetuadas usando apenas aritmética em $\mathbb{F}_{q^d}$. Pontos de $E'(\mathbb{F}_{q^d})$ são mapeados de volta em pontos de $E(\mathbb{F}_{q^k})$ apenas quando for necessário para o cálculo de um emparelhamento. Isto evita muitas operações em $\mathbb{F}_{q^k}$ e reduz pela metade a utilização de banda.

Por exemplo, se $k = 2$, protocolos baseados em emparelhamentos podem ser implementados usando apenas aritmética em $E(\mathbb{F}_q)$ e suporte simples a operações em $\mathbb{F}_{q^2}$. Para valores mais altos de k, sugerimos implementar $\mathbb{F}_{q^k}$ como $\mathbb{F}_q[x]/R_k(x)$, onde $R_k(x)$ é o polinômio irredutível mais esparso possível contendo apenas termos de grau par. Nesse caso, os elementos de $\mathbb{F}_{q^d}$ são polinômios apenas com termos de grau par.

8.5.1 Algumas observações sobre os grupos selecionados

Lema 5. *Seja $Q = (X, Y) \in E(\mathbb{F}_{q^k})$ um ponto finito (i.e. $Q \neq O$. Então $\Phi^d(Q) = -Q$ se, e somente se, $X^{q^d-1} = 1$ (i.e. $X \in \mathbb{F}_{q^d}$) e $Y^{q^d-1} = -1$.*

Demonstração. Uma vez que $-Q = (X, -Y)$ (cfr. (SILVERMAN, 1986, seção III.1)), concluímos que $\Phi^d(X, Y) = (X^{q^d}, Y^{q^d}) = (X, -Y)$ se, e somente se $X^{q^d-1} = 1$ (i.e. $X \in \mathbb{F}_{q^d}$) e $Y^{q^d-1} = -1$. □

Assim, $\Psi(E'(\mathbb{F}_{q^d}))$ é precisamente o grupo dos pontos de $E(\mathbb{F}_{q^k})$ satisfazendo $\Phi^d(Q) = -Q$, que é um subgrupo dos pontos de traço zero de $E(\mathbb{F}_{q^k})$.

Destarte um modo alternativo de selecionar Q em nosso algoritmo seria escolher um ponto aleatório $R \in E(\mathbb{F}_{q^k})$ e fazer $Q \leftarrow R - \Phi^d(R)$. Porém, esta possibilidade é mais lenta que encontrar pontos em $E'(\mathbb{F}_{q^d})$, e não se obtém o bônus no desempenho de operações que não envolvem emparelhamentos.

O lema 5 pode ser usado para mostrar que os pontos de r-torção e traço zero têm uma forma especial:

Corolário 1. *Seja $Q = (X, Y) \in E(\mathbb{F}_{q^k})[r]$ um ponto finito com $\mathrm{tr}(Q) = O$. Então $X \in \mathbb{F}_{q^d}$ e $Y^{q^d-1} = -1$.*

Demonstração. Como $\mathrm{tr}(Q) = O$, o ponto Q está no auto-espaço associado ao auto-valor q do endomorfismo de Frobenius Φ, isto é, $\Phi(Q) = [q]Q$. Segue daí que $q^d \equiv -1$

$(\mathrm{mod}\ r)$, porque $q^{2d} \equiv 1\ (\mathrm{mod}\ r)$ e $2d = k$ é o menor inteiro para o qual isto vale (pela definição de grau de imersão). Logo, $\Phi^d(Q) = -Q$. Pelo lema 5, $X^{q^d-1} = 1$ e $Y^{q^d-1} = -1$. □

8.6 Cálculo eficiente do emparelhamento de Tate

Nesta seção propomos diversos aperfeiçoamentos a um algoritmo não publicado de Miller (MILLER, 1986) para calcular o emparelhamento de Tate.

Sejam $P \in E(\mathbb{F}_q)[r]$ e $Q \in E(\mathbb{F}_{q^k})$ pontos linearmente independentes, com $k > 1$. Seja f_r uma função racional sobre $\mathbb{F}_{q^k}$ com divisor $(f_r) = r(P) - r(O)$. Recordando a definição 17, o emparelhamento de Tate de ordem r é dado por $e_r(P, Q) = f_r(\mathcal{D})^{(q^k-1)/r}$, onde $\mathcal{D} \sim (Q) - (O)$ e o suporte de $\mathcal{D}$ não contém P nem O. Esta definição não sugere um método efetivo de cálculo, pois a função f_r não está explicitamente definida. A idéia de Miller é decompor f_r em funções de grau 1, a saber, as linhas retas definidas pela construção geométrica ("secantes e tangentes") da lei de grupo encontradas durante o cálculo de $[r]P$.

Seja $g_{U,V}$ a (equação da) reta que passa pelos pontos $U, V \in \langle P \rangle$, e seja $g_{U,V}(Q)$ o valor da equação dessa reta no ponto $Q \in E(\mathbb{F}_{q^k})$. A abreviação g_V denota $g_{V,-V}$. As funções $g_{U,V}$ são denominadas *funções de linha*. Em coordenadas afins, se a curva for definida pela equação $E(\mathbb{F}_q) : y^2 = x^3 + ax + b$, para $U = (x_U, y_U)$, $V = (x_V, y_V)$ e $Q = (x, y)$ temos:

$$
\begin{aligned}
g_{U,V}(Q) &= 1,\ Q \in \langle P \rangle. \\
g_{U,U}(Q) &= \lambda_1(x - x_U) + y_U - y,\ Q \notin \langle P \rangle. \\
g_{U,V}(Q) &= \lambda_2(x - x_U) + y_U - y,\ Q \notin \langle P \rangle,\ U \neq V. \\
g_U(Q) &= x - x_U,\ Q \notin \langle P \rangle.
\end{aligned}
$$

onde

$$\lambda_1 = \frac{3x_U^2 + a}{2y_U}, \quad \lambda_2 = \frac{y_V - y_U}{x_V - x_U}.$$

Teorema 4 (Fórmula de Miller). *Seja P um ponto de $E(\mathbb{F}_q)$ e f_c uma função com divisor $(f_c) = c(P) - ([c]P) - (c-1)(O)$, $c \in \mathbb{Z}$. Para todo $a, b \in \mathbb{Z}$, $f_{a+b}(\mathcal{D}) = f_a(\mathcal{D}) \cdot f_b(\mathcal{D}) \cdot g_{[a]P,[b]P}(\mathcal{D}) / g_{[a+b]P}(\mathcal{D})$ salvo um fator não nulo constante.*

Demonstração. Os divisores das funções de linha satisfazem:

$$\begin{aligned}(g_{[a]P,[b]P}) &= ([a]P) + ([b]P) + (-[a+b]P) - 3(O), \\ (g_{[a+b]P}) &= ([a+b]P) + (-[a+b]P) - 2(O).\end{aligned}$$

Logo, $(g_{[a]P,[b]P}) - (g_{[a+b]P}) = ([a]P) + ([b]P) - ([a+b]P) - (O)$. Da definição de f_c vemos que:

$$\begin{aligned}(f_{a+b}) &= (a+b)(P) - ([a+b]P) - (a+b-1)(O) \\ &= a(P) - ([a]P) - (a-1)(O) \\ &+ b(P) - ([b]P) - (b-1)(O) \\ &+ ([a]P) + ([b]P) - ([a+b]P) - (O) \\ &= (f_a) + (f_b) + (g_{[a]P,[b]P}) - (g_{[a+b]P}).\end{aligned}$$

Portanto, $f_{a+b}(\mathcal{D}) = f_a(\mathcal{D}) \cdot f_b(\mathcal{D}) \cdot g_{[a]P,[b]P}(\mathcal{D}) \,/\, g_{[a+b]P}(\mathcal{D})$. □

Pela definição de f_c, vemos que $(f_0) = (f_1) = 0$, significando que f_0 e f_1 são constantes sobre os pontos da curva, e portanto $f_0(\mathcal{D}) = f_1(\mathcal{D}) = 1$ para qualquer divisor $\mathcal{D}$ de grau zero (cfr. seção 5.4). Assim, $f_{a+1}(\mathcal{D}) = f_a(\mathcal{D}) \cdot g_{[a]P,P}(\mathcal{D}) / g_{[a+1]P}(\mathcal{D})$ e $f_{2a}(\mathcal{D}) = f_a(\mathcal{D})^2 \cdot g_{[a]P,[a]P}(\mathcal{D}) / g_{[2a]P}(\mathcal{D})$ para $a > 0$. Essas observações permitem escrever um algoritmo iterativo que calcula $f_r(\mathcal{D})$ combinando as fórmulas acima com o método da decomposição binária para calcular $[r]P$. Infelizmente, não é possível usar diretamente $\mathcal{D} = (Q) - (O)$ para calcular $g_{U,V}(\mathcal{D})$, pois $g_{U,V}$ tem um pólo (de ordem 2 ou 3) em O. Podemos, porém, usar um divisor equivalente $\mathcal{D} = (Q+R) - (R)$,

onde o ponto arbitrário $R \in E(\mathbb{F}_{q^k})$ não é um pólo nem um zero de $g_{U,V}$.

Seja então $(r_t, r_{t-1}, \ldots, r_1, r_0)$ a representação binária da ordem $r \geqslant 0$ de P, onde $r_i \in \{0, 1\}$ e $r_t \neq 0$. Assume-se $r \mid q^k - 1$. O cálculo do emparelhamento de Tate de ordem r, $e_r(P, Q)$, procede da seguinte maneira:

Algoritmo de Miller:

$R \leftarrow$ **aleat**$(E(\mathbb{F}_{q^k}))$, $\mathcal{D} \leftarrow (Q + R) - (R)$

$f \leftarrow 1$, $V \leftarrow P$

para $i \leftarrow t-1, t-2, \ldots, 1, 0$ **faça** $\{$

$\quad f \leftarrow f^2 \cdot g_{V,V}(\mathcal{D})/g_{[2]V}(\mathcal{D})$, $V \leftarrow [2]V$

$\quad$ **se** $r_i = 1$ **então** $f \leftarrow f \cdot g_{V,P}(\mathcal{D})/g_{V+P}(\mathcal{D})$, $V \leftarrow V + P$

$\}$

devolva $e_r(P, Q) \leftarrow f^{(q^k-1)/r}$

Este algoritmo falha para a escolha feita do ponto R se esse ponto for um pólo ou um zero de alguma das ocorrências de $g_{U,V}$. Nesse caso, um novo ponto R teria que ser escolhido e o processo reiniciado, mas a probabilidade de uma tal ocorrência na prática é negligível.

O algoritmo de Miller, sendo concomitante ao cálculo de uma multiplicação por escalar, beneficia-se das mesmas técnicas de otimização disponíveis para este cálculo (MENEZES; OORSCHOT; VANSTONE, 1999, seção 14.6). Uma possibilidade conceitualmente interessante é a decomposição de f_r em polinômios mais gerais que as retas utilizadas originalmente; esta idéia encontra-se desenvolvida em (EISENTRAEGER; LAUTER; MONTGOMERY, 2003) para o caso de decomposição em parábolas, mas os resultados pouco expressivos não parecem justificar a maior complexidade dessas variantes.

8.6.1 Simplificação de divisores

É possível simplificar consideravelmente o algoritmo de Miller para calcular o emparelhamento de Tate. Em particular, o cálculo completo de $f_r(\mathcal{D})$ é desnecessário, e pode ser substituído por uma expressão mais eficiente.

Teorema 5. *Se $P \in E(\mathbb{F}_q)[r]$ e $Q \in E(\mathbb{F}_{q^k})$ são pontos linearmente independentes, então $e_r(P,Q) = f_r(Q)^{(q^k-1)/r}$.*

Demonstração. Seja $R \notin \{O, -P, Q, Q-P\}$ um ponto da curva $E(\mathbb{F}_{q^k})$, e seja f'_r uma função racional com divisor $(f'_r) = r(P+R) - r(R) \sim (f_r)$, de modo que $e_r(P,Q) = f'_r((Q)-(O))^{(q^k-1)/r}$. Uma vez que f'_r não tem um zero nem um pólo em O, temos $f'_r((Q)-(O)) = f'_r(Q)/f'_r(O)$. Ora, a função f'_r é definida a menos de um fator constante, e pode ser escolhida[3] de modo que $f'_r(O) = 1$. Portanto, $e_r(P,Q) = f'_r(Q)^{(q^k-1)/r}$.

Por outro lado, $(f'_r) = r((P+R)-(R)) = r((P)-(O)+(g)) = (f_r)+r(g)$ para alguma função racional g, já que $(P+R)-(R) \sim (P)-(O)$. Assim, $f'_r = f_r g^r$, e como Q não é nem um zero nem um pólo de f_r ou f'_r (de modo que $g(Q) \in \mathbb{F}^*_{q^k}$ é bem definida), segue que $f'_r(Q)^{(q^k-1)/r} = f_r(Q)^{(q^k-1)/r} g(Q)^{q^k-1} = f_r(Q)^{(q^k-1)/r}$. □

Note-se que o caso especial em que P e Q são linearmente *dependentes*, onde o teorema não se aplica, pode ser tratado trivialmente à parte, já que nesses casos $e_r(P,Q) = 1$. Um exemplo é o caso $Q \in \{P, O\}$, que caraterizaria um divisor cujo suporte não é disjunto do suporte de (f_r).

8.6.2 Eliminação de denominadores

Sob certas circunstâncias de grande relevância prática, os denominadores $g_{[2]V}(Q)$ e $g_{V+P}(Q)$ no algoritmo de Miller podem ser completamente descartados. Mostra-

[3] Por exemplo, se f''_r é qualquer função com divisor $(f''_r) = r(P+R) - r(R)$, basta fazer $f'_r \equiv f''_r / f''_r(O)$.

remos que, para curvas supersingulares, isto ocorre no cálculo de $e_r(P, \phi(Q))$, com $Q \in E(\mathbb{F}_q)[r]$ e um mapa de distorção $\phi : E(\mathbb{F}_q) \to E(\mathbb{F}_{q^k})$ propriamente escolhido; para curvas ordinárias em característica $p > 3$ com grau de imersão par, basta escolher Q segundo o algoritmo de seleção de geradores delineado na seção 8.5. No que segue, seja d um fator próprio de k, isto é, $d \mid k$ e $d < k$.

Lema 6. *$q^d - 1$ é um fator de $(q^k - 1)/r$.*

Demonstração. Começamos com a fatoração $q^k - 1 = (q^d - 1) \sum_{i=0}^{k/d-1} q^{id}$. Uma vez que o grau de imersão é $k > 1$, temos $r \mid q^k - 1$ e $r \nmid q^d - 1$. Logo, $r \mid \sum_{i=0}^{k/d-1} q^{id}$, e portanto $q^d - 1$ sobrevive como fator de $(q^k - 1)/r$. □

Corolário 2 (Fatores irrelevantes). *É possível multiplicar ou dividir livremente $f_r(\mathcal{D})$ por qualquer fator não nulo $x \in \mathbb{F}_{q^d}$ sem afetar o valor do emparelhamento de Tate.*

Demonstração. Para completar o cálculo do emparelhamento, $f_r(\mathcal{D})$ é elevado ao expoente $(q^k - 1)/r$. Pelo lema 6, este expoente contém um fator $q^d - 1$, e portanto pelo Pequeno Teorema de Fermat (cfr. teorema 3), $x^{(q^k-1)/r} = 1$. □

Teorema 6. *Para curvas ordinárias em característica $p > 3$ com grau de imersão par e Q escolhido segundo o algoritmo de seleção de geradores, ou para curvas supersingulares com os parâmetros listados na tabela 3, os denominadores $g_{[2]V}$ e g_{V+P} no algoritmo de Miller podem ser completamente descartados sem alterar os valor de $e_r(P, Q)$.*

Demonstração. Mostraremos que, nos casos previstos pelo teorema, os denominadores no algoritmo de Miller estão definidos em $\mathbb{F}_{q^d}$, onde $d = k/2$.

Para curvas ordinárias e Q escolhido segundo o algoritmo de seleção de geradores, os denominadores na fórmula de Miller têm a forma $g_U(Q) \equiv x - u$, onde $x \in \mathbb{F}_{q^d}$ é a abscissa de Q e $u \in \mathbb{F}_q$ é a abscissa de U. Logo, $g_U(Q) \in \mathbb{F}_{q^d}$.

Para as curvas supersingulares com os parâmetros listados na tabela 3, podemos formular argumentos dedicados:

- (Característica 2) Seja $q \equiv 2^m$. Da condição $s^4 = s$ segue por indução que $s^{4^t} = s$ para todo $t \geqslant 0$; em particular, $s^{q^2} = s^{2^{2m}} = s$, isto é, $s \in \mathbb{F}_{q^2}$. Os denominadores na fórmula de Miller têm a forma $g_U(\phi(Q)) \equiv x + s^2 + u$, onde $x \in \mathbb{F}_q$ é a abscissa de Q e $u \in \mathbb{F}_q$ é a abscissa de U. Logo, $g_U(\phi(Q)) \in \mathbb{F}_{q^2}$.

- (Característica 3) Seja $q \equiv 3^m$. Da condição $r_b^3 - r_b - b = 0$ segue por indução que $r_b^{3^t} = r_b + b(t \bmod 3)$ para todo $t \geqslant 0$; em particular, $r_b^{q^3} = r_b^{3^{3m}} = r_b$, isto é, $r_b \in \mathbb{F}_{q^3}$. Os denominadores na fórmula de Miller têm a forma $g_U(\phi(Q)) \equiv r_b - x - u$, onde $x \in \mathbb{F}_q$ é a abscissa de Q e $u \in \mathbb{F}_q$ é a abscissa de U. Logo, $g_U(\phi(Q)) \in \mathbb{F}_{q^3}$.

- (Característica $p > 3$) Os denominadores na fórmula de Miller têm a forma $g_U(\phi(Q)) \equiv -x - u$, onde $x \in \mathbb{F}_p$ é a abscissa de Q e $u \in \mathbb{F}_p$ é a abscissa de U. Logo, $g_U(\phi(Q)) \in \mathbb{F}_p$.

Em todos os casos, os denominadores estão definidos em $\mathbb{F}_{q^d}$. Pelo corolário 2, esses denominadores podem ser descartados sem alterar o valor do emparelhamento de Tate. □

Essas observações sugerem a seguinte variante[4] (determinística) do algoritmo de Miller:

Algoritmo BKLS:

$f \leftarrow 1, \;\; V \leftarrow P$

para $i \leftarrow t-1, t-2, \ldots, 1, 0$ **faça** $\{$

[4]Nosso algoritmo tem sido chamado BKLS (DUURSMA; LEE, 2003) devido às iniciais dos autores da referência (BARRETO et al., 2002) em que foi originalmente publicado.

$f \leftarrow f^2 \cdot g_{V,V}(Q), \;\; V \leftarrow [2]V$

se $r_i = 1$ **então** $f \leftarrow f \cdot g_{V,P}(Q), \;\; V \leftarrow V + P$

}

devolva $e_r(P,Q) \leftarrow f^{(q^k-1)/r}$

8.6.3 Acoplando o emparelhamento com a característica

Pode-se acoplar o cálculo de f_n com o método mais eficiente da decomposição ternária em característica 3. Para este fim necessita-se de uma fórmula recursiva para $f_{3a}(Q)$, que é fácil de obter a partir da fórmula de Miller: o divisor de f_{3a} é $(f_{3a}) = 3(f_a) + (g_{[a]P,[a]P}) + (g_{[2a]P,[a]P}) - (g_{[2a]P}) - (g_{[3a]P})$, portanto descartando os denominadores irrelevantes obtém-se:

$$f_{3b}(Q) = f_b^3(Q) \cdot g_{[a]P,[a]P}(Q) \cdot g_{[2a]P,[a]P}(Q).$$

Note-se que não é necessário calcular efetivamente $[2a]P$, porque os coeficientes de $g_{[2a]P,[a]P}$ podem ser obtidos de $[a]P$ e $[3a]P$.

Em característica 3, a fórmula da triplicação de ponto é por si só mais eficiente que a fórmula de duplicação, já que a operação de elevação ao quadrado, que toma tempo $O(m^2)$, é substituída pela operação de elevação ao cubo, que tem complexidade no máximo linear; além disso, a triplicação é executada apenas uma fração $\log_3 2$ das vezes que a duplicação seria executada.

Adicionalmente, para o emparelhamento de Tate de ordem $n = (3^{(m-1)/2} \pm 1) \cdot 3^{(m+1)/2} + 1$ a contribuição da multiplicação escalar subjacente à complexidade do algoritmo de Miller é apenas $O(m^2)$ em vez de $O(m^3)$, já que ela envolve somente duas adições ou uma adição e uma subtração. A mesma observação vale para curvas elípticas supersingulares em característica 2.

8.6.4 Escolha da ordem do subgrupo

O cálculo de emparelhamentos pode beneficiar-se de uma escolha cuidadosa de parâmetros, particularmente a ordem r do subgrupo onde se efetuam os cálculos. De modo geral, um valor r com representação binária esparsa (isto é, com baixo peso de Hamming) reduz apreciavelmente o esforço computacional do cálculo de um emparelhamento, pois elimina muitas operações de adição ou subtração no algoritmo subjacente de multiplicação por escalar. Por exemplo, em vez de escolher uma ordem prima aleatória, é conveniente usar sempre que possível um *primo de Solinas* (SOLINAS, 1999) da forma $r = 2^\alpha \pm 2^\beta \pm 1$, já que $[r]P = [2^\beta(2^{\alpha-\beta} \pm 1) \pm 1]P$ envolve apenas duas adições ou subtrações mais α duplicações.

Pode-se obter uma ordem prima r razoavelmente esparsa com o método de construção descrito na seção 8.4. Para tanto, é preciso que o polinômio $\Phi_k(t-1)$ seja esparso (como acontece se todos os fatores primos de k forem bastante pequenos), que se reforce a condição $r \mid \Phi_k(t-1)$ para $r = \Phi_k(t-1)$, e que se restrinja $t-1$ a valores esparsos. Alguns testes simples (mas de modo nenhum exaustivos) mostram ser exeqüível obter ordens primas satisfazendo $\omega(r) \sim 0.1 \log r$ para $k = 6$; por exemplo, ordens de 160 bits podem ser escolhidas com peso de Hamming ao redor de 16. Infelizmente, há pouca esperança de obter dessa forma um primo de Solinas para $k > 2$.

8.6.5 Otimizando a exponenciação final

O cálculo do emparelhamento de Tate $e_r(P, Q)$ envolve uma exponenciação final, a saber, $f_r(Q)^z$ onde $z = (q^k - 1)/r$. Normalmente essa exponenciação demanda $O(m^3)$ passos. Todavia, o expoente z exibe uma estrutura bastante simples, que pode ser explorada para reduzir o esforço computacional, no caso de curvas supersingulares, para apenas $O(m^2)$ passos:

1. (Característica 2) Seja $q = 2^m$. Como vimos na demonstração do teorema 6, o

expoente de Tate é da forma $z = (q + 1 \pm \sqrt{2q})(q^2 - 1)$. Portanto, para obter $s = w^z$ calculam-se $t = w^q \cdot w \cdot w^{\pm\sqrt{2q}}$ e $s = t^{q^2}/t$. Elevar aos expoentes q, $\sqrt{2q}$ e q^2 (todos potências de 2) pode ser efetuado em tempo $O(m)$, de modo que a operação completa é dominada pelo número pequeno e constante de multiplicações e inversões, que necessitam tempo $O(m^2)$.

2. (Característica 3) Seja $q = 3^m$. Como vimos na demonstração do teorema 6, o expoente de Tate é da forma $z = (q + 1 \pm \sqrt{3q})(q^3 - 1)(q + 1)$. Portanto, para obter $s = w^z$ calculam-se $u = w^q \cdot w \cdot w^{\pm\sqrt{3q}}$, $t = u^{q^3}/u$, e $s = t^q \cdot t$. Elevar aos expoentes q, $\sqrt{3q}$ e q^3 (todos potências de 3) pode ser efetuado em tempo $O(m)$, de modo que a operação completa é dominada pelo número pequeno e constante de multiplicações e inversões, que necessitam tempo $O(m^2)$.

3. (Característica $p > 3$) Suponhamos que $p \equiv 2 \pmod 3$ e $p \equiv 3 \pmod 4$. A ordem da curva $E_{1,b}$ é $n = p + 1$. Seja r a ordem do subgrupo de interesse; note-se que $r \mid p+1$. Consideremos o cenário onde a representação de um ponto $t \in \mathbb{F}_{p^2}$ é $t = u + iv$ com $u, v \in \mathbb{F}_p$ e i satisfazendo $i^2 + 1 = 0$. O expoente de Tate é $z = (p^2 - 1)/r = ((p + 1)/r) \cdot (p - 1)$. Para obter $s = w^z \bmod p$, calculam-se $t = w^{(p+1)/r} \equiv u + iv$ e $s = (u + iv)^{p-1} = (u + v)^p/(u + iv) = (u - v)/(u + iv)$, usando a linearidade do mapa de Frobenius em característica p e o fato que $i^p = -i$ para $p \equiv 3 \pmod 4$. Simplificando, obtemos $s = (u^2 - v^2)/(u^2 + v^2) - 2uvi/(u^2 + v^2)$.

Técnicas semelhantes podem ser usadas para curvas MNT com grau de imersão k arbitrário usando propriedades dos polinômios ciclotômicos (def. 18), particularmente o fato (LIDL; NIEDERREITER, 1997, teorema 2.45) de que $q^k - 1 = \prod_{d|k} \Phi_d(q)$. Contudo, neste caso a redução de complexidade é apenas de um fator constante (a saber, cerca de $1/8$ se k for par), permanecendo como $O(m^3)$.

8.7 Técnicas adicionais

8.7.1 Pré-computação com base fixa

Sistemas criptográficos reais baseados em emparelhamentos freqüentemente precisam calcular emparelhamentos $e_n(P, Q)$ onde P é fixo (por exemplo, se P for o ponto base da curva) ou usado repetidamente (por exemplo, uma chave pública). Nesses casos, a multiplicação escalar subjacente ao algoritmo de Miller pode ser executado apenas uma vez para pré-calcular os coeficientes das equações de linha $g_{U,V}(Q)$. O aumento de desempenho resultante desta técnica é mais proeminente em característica $p > 3$, e pode atingir cerca de 40%.

8.7.2 Multiplicação de Karatsuba

O método de multiplicação de Karatsuba permite que um produto em $\mathbb{F}_{q^k}$ seja implementado com cerca de $k^2/2$ produtos em $\mathbb{F}_q$. Por comparação, o algoritmo convencional de multiplicação exigiria k^2 desses produtos.

Consideremos, por exemplo, $k = 6$. Para multiplicar $a = a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$ e $b = b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0$, obtendo o produto $c = c_{10}x^{10} + c_9x^9 + c_8x^8 + c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$, calculam-se:

$$\begin{aligned} d_{00} &\leftarrow a_0 \cdot b_0, \quad d_{11} \leftarrow a_1 \cdot b_1, \quad d_{22} \leftarrow a_2 \cdot b_2, \\ d_{33} &\leftarrow a_3 \cdot b_3, \quad d_{44} \leftarrow a_4 \cdot b_4, \quad d_{55} \leftarrow a_5 \cdot b_5, \end{aligned}$$

$$
\begin{aligned}
d_{01} &\leftarrow (a_0 + a_1) \cdot (b_0 + b_1) - d_{00} - d_{11}, \\
d_{02} &\leftarrow (a_0 + a_2) \cdot (b_0 + b_2) - d_{00} - d_{22}, \\
d_{04} &\leftarrow (a_0 + a_4) \cdot (b_0 + b_4) - d_{00} - d_{44}, \\
d_{13} &\leftarrow (a_1 + a_3) \cdot (b_1 + b_3) - d_{11} - d_{33}, \\
d_{15} &\leftarrow (a_1 + a_5) \cdot (b_1 + b_5) - d_{11} - d_{55}, \\
d_{23} &\leftarrow (a_2 + a_3) \cdot (b_2 + b_3) - d_{22} - d_{33}, \\
d_{24} &\leftarrow (a_2 + a_4) \cdot (b_2 + b_4) - d_{22} - d_{44}, \\
d_{35} &\leftarrow (a_3 + a_5) \cdot (b_3 + b_5) - d_{33} - d_{55}, \\
d_{45} &\leftarrow (a_4 + a_5) \cdot (b_4 + b_5) - d_{44} - d_{55},
\end{aligned}
$$

$$
\begin{aligned}
d_{03} &\leftarrow (a_0 + a_1 + a_2 + a_3) \cdot (b_0 + b_1 + b_2 + b_3) \\
&\quad -(d_{00} + d_{11} + d_{22} + d_{33} + d_{01} + d_{02} + d_{13} + d_{23}), \\
d_{05} &\leftarrow (a_0 + a_1 + a_4 + a_5) \cdot (b_0 + b_1 + b_4 + b_5) \\
&\quad -(d_{00} + d_{11} + d_{44} + d_{55} + d_{01} + d_{04} + d_{15} + d_{45}), \\
d_{25} &\leftarrow (a_2 + a_3 + a_4 + a_5) \cdot (b_2 + b_3 + b_4 + b_5) \\
&\quad -(d_{22} + d_{33} + d_{44} + d_{55} + d_{23} + d_{24} + d_{35} + d_{45}),
\end{aligned}
$$

$$
\begin{aligned}
c_0 &\leftarrow d_{00}, \\
c_1 &\leftarrow d_{01}, \\
c_2 &\leftarrow d_{02} + d_{11}, \\
c_3 &\leftarrow d_{03}, \\
c_4 &\leftarrow d_{04} + d_{13} + d_{22}, \\
c_5 &\leftarrow d_{05} + d_{23}, \\
c_6 &\leftarrow d_{15} + d_{24} + d_{33}, \\
c_7 &\leftarrow d_{25}, \\
c_8 &\leftarrow d_{35} + d_{44}, \\
c_9 &\leftarrow d_{45}, \\
c_{10} &\leftarrow d_{55}.
\end{aligned}
$$

Assim, apenas 18 multiplicações em $\mathbb{F}_q$ são necessárias para obter um produto em $\mathbb{F}_{q^6}$; o algoritmo convencional chegaria ao mesmo resultado com 36 multiplicações em $\mathbb{F}_q$.

8.7.3 Estrutura do mapa de distorção

Surpreendentemente, contudo, o método de Karatsuba *não* é a técnica mais rápida de multiplicação em todas as circunstâncias. Conforme visto na seção 8.6.2, ocorre com freqüência que o emparelhamento efetivo a ser calculado é $e_n(P, \phi(Q))$ onde P e Q estão ambos na curva $E(\mathbb{F}_q)$ em vez de $E(\mathbb{F}_{q^k})$, e o algoritmo de emparelhamento pode fazer uso explícito da forma do mapa de distorção ϕ para reduzir o número de produtos em $\mathbb{F}_q$ envolvidos na fórmula de Miller para apenas 2 por equação de linha. Por exemplo, se $Q = (u, v)$ o mapa de distorção indicado na tabela 2 para $E_{3,-1}(\mathbb{F}_{3^{97}})$ e o polinômio irredutível $x^6 + x - 1$ resultam em $\phi(Q) = [r(x), s(x)] = [-x^4 + x^3 - x^2 - u, vx^4 + vx^3 - vx^2 + vx]$, de maneira que as equações de linhas na fórmula de Miller tomam a forma $ar(x) + bs(x) + c = (-a + bv)x^4 + (a + bv)x^3 - (a + bv)x^2 +$

$bvx + (-au + c)$, que contém apenas os produtos au e bv, ambos em $\mathbb{F}_q$ (note-se que, freqüentemente, um ou ambos dentre a e b serão 1 ou 0).

8.7.4 Inversão no corpo finito

Existe uma alternativa simples aos algoritmos de inversão comumente usados (como o algoritmo estendido de Euclides, o algoritmo quase-inverso e o método de Itoh-Teechai-Tsujii) para calcular inversos multiplicativos em $\mathbb{F}_{q^k}$, a saber, resolvendo *formalmente* o sistema linear $u^{-1} \cdot u = 1$, de dimensão $k \times k$, usando determinantes. Desta forma, uma única inversão em $\mathbb{F}_q$ (a saber, a do determinante completo do sistema) é necessária para calcular um inverso em $\mathbb{F}_{q^k}$. Contudo, esta otimização não tem efeitos notáveis em sistemas criptográficos baseados em emparelhamentos, já que, geralmente, apenas algumas poucas inversões em $\mathbb{F}_{q^k}$ são necessárias para cada emparelhamento calculado.

8.7.5 Coordenadas projetivas

Koblitz (KOBLITZ, 1998) descreve um método de adição elíptica em característica 3 usando coordenadas projetivas ao custo de 10 multiplicações no corpo finito subjacente. Na verdade, a adição de pontos pode ser realizada com 9 multiplicações. Sejam $P_1 = (x_1, y_1, z_1)$, $P_2 = (x_2, y_2, 1)$; calcula-se $P_3 = P_1 + P_2 = (x_3, y_3, z_3)$ como:

$$A \leftarrow x_2 z_1 - x_1,\ B \leftarrow y_2 z_1 - y_1,\ C \leftarrow A^3,\ D \leftarrow C - z_1 B^2,$$

$$x_3 \leftarrow x_1 C - AD,\ y_3 \leftarrow BD - y_1 C,\ z_3 \leftarrow z_1 C.$$

Para recompor P_3 em coordenadas afins basta calcular $P_3 = (x_3/z_3,\ y_3/z_3)$. Isto envolve uma única inversão no corpo finito, que normalmente é postergada para o fim da operação completa de multiplicação por escalar.

Tabela 5: Tempos de cálculo do emparelhamento de Tate

corpo finito	tempo
$\mathbb{F}_{2^{241}}$	6.18 ms
$\mathbb{F}_{2^{271}}$	9.92 ms
$\mathbb{F}_{2^{283}}$	10.51 ms
$\mathbb{F}_{2^{353}}$	17.95 ms
$\mathbb{F}_{3^{97}}$	11.13 ms
$\mathbb{F}_{3^{163}}$	37.49 ms
$\mathbb{F}_p$, $\lvert p\rvert = 512$ bits	20.0 ms
$\mathbb{F}_p$ com pré-processamento	8.6 ms

8.8 Alguns resultados experimentais

Embora a eficiência dos algoritmos por nós propostos seja teoricamente quantificável em termos de número de operações elementares necessárias para a sua execução, é interessante ilustrar o desempenho efetivo desses algoritmos na prática por meio de implementações em software.

As operações mais pesadas em qualquer sistema criptográfico baseado em emparelhamentos são os cálculos dos próprios emparelhamentos. Apresentamos na tabela 5 os tempos observados para essas operações em plataforma Pentium IV 1 GHz, usando implementações elaboradas por Keith Harrison (HP Laboratories, United Kingdom), a quem manifestamos nossos sinceros agradecimentos.

O desempenho da geração de assinaturas BLS é comparável ao de assinaturas RSA ou DSA com o mesmo nível de segurança, como se observa a partir da tabela 6. As operações foram efetuadas em plataforma Pentium III 1 GHz. Todas as implementações exceto as do algoritmo BLS foram gentilmente elaboradas por Michael Scott (Dublin City University, Ireland), a quem manifestamos nossos agradecimentos.

Tempos de verificação de assinaturas BLS são listados na tabela 7, mostrando um aumento de um fator 36 para $\mathbb{F}_p$ e de 125 para $\mathbb{F}_{3^{97}}$ em relação a valores previamente publicados.

Tabela 6: Tempos de geração de assinaturas BLS

algoritmo	tempo de assinatura
RSA, $\|n\| = 1024$ bits, $\|d\| = 1007$ bits	7,9 ms
ECDSA em $\mathbb{F}_{2^{160}}$	5,7 ms
DSA, $\|p\| = 1024$ bits, $\|q\| = 160$ bits	4,1 ms
ECDSA em $\mathbb{F}_p$, $\|p\| = 160$ bits	4,0 ms
BLS em $\mathbb{F}_{3^{97}}$	3,5 ms
BLS em $\mathbb{F}_p$, $\|p\| = 157$ bits	3,0 ms

Tabela 7: Tempos de verificação de assinaturas BLS

original (BONEH; LYNN; SHACHAM, 2002)	nosso em $\mathbb{F}_p$	nosso em $\mathbb{F}_{3^{97}}$
2900 ms	80	23 ms

Numa situação mais realística, a tabela 8 traz a resolução na localização de alterações (porcentagem da área total relatada como alterada se qualquer pixel em seu interior for modificado) e os tempos de inserção e detecção de marcas d'água HBC2. As imagens têm dimensões 640×480 pixels, e as assinaturas digitais empregadas são baseadas no algoritmo BLS sobre $\mathbb{F}_{3^{97}}$ e no algoritmo RSA com módulo de 1024 bits. Para o tratamento das imagens, utilizamos a biblioteca IMG, de autoria de Hae Yong Kim.

8.9 Abscissas vs. ordenadas em característica 3

A complexidade computacional da construção iterativa probabilística da seção 7.3 é cúbica no grau de extensão do corpo finito subjacente, isto é $O(m^3)$, proveniente da

Tabela 8: Inserção e detecção de marcas d'água HBC2

algoritmo	granularidade	# blocos	resolução	inserção	detecção
BLS	8×8 pixels	4800	$0,04\%$	23 s	124,7 s
BLS	20×20 pixels	768	$0,26\%$	3,7 s	20,0 s
RSA-1024	20×20 pixels	768	$0,26\%$	6,1 s	0,31 s

necessidade de calcular raízes quadradas nesse corpo finito.

Embora a técnica descrita na seção 8.1 reduza a complexidade para $O(m^2 \log m)$, proporemos aqui para característica 3 uma abordagem diferente, mas conceitualmente simples, que evita inteiramente o cálculo de raízes quadradas e executa em apenas $O(m^2)$ passos. Esta complexidade deriva do esforço computacional de resolver um sistema de m equações lineares com coeficientes fixos em $\mathbb{F}_3$, e do cálculo do quadrado de um elemento de $\mathbb{F}_{3^m}$.

Comparado com a construção original, nosso esquema precisa de um número ligeiramente maior de passos aleatórios para produzir um valor de *hash*, a saber, cerca de três em vez de duas consultas a um oráculo de *hash*, mas a eficiência muito maior de cada consulta compensa este comportamento. O tamanho da assinatura também aumenta em cerca de 2 bits, ou mais precisamente o espaço necessário para representar um elemento de $\mathbb{F}_3$.

Por outro lado, ao contrário das otimizações que propusemos até agora, esta nova construção implica uma alteração estrutural no algoritmo BLS que, mesmo pequena, exige uma análise de segurança à parte, desenvolvida adiante.

8.9.1 *Hash* eficiente em pontos da curva

Seja $\mathcal{C} : \mathbb{F}_{3^m} \to \mathbb{F}_{3^m}$ a função $\mathcal{C}(x) = x^3 - x$. O núcleo de $\mathcal{C}$ é $\mathbb{F}_3$ (LIDL; NIEDERREITER, 1997, capítulo 2, seção 1), portanto o posto de $\mathcal{C}$ is $m-1$ (HOFFMAN; KUNZE, 1971, seção 3.1, teorema 2).

Definição 20 ((LIDL; NIEDERREITER, 1997)). *O* traço absoluto *de um elemento* $a \in \mathbb{F}_{3^m}$ *é*

$$\mathrm{tr}(a) = a + a^3 + a^9 + \cdots + a^{3^{m-1}}.$$

O traço absoluto está sempre em $\mathbb{F}_3$ como se pode verificar facilmente notando que, pela definição acima, $\mathcal{C} \circ \mathrm{tr} \equiv 0$, isto é, $\mathrm{tr}(a)^3 = \mathrm{tr}(a)$ para todo $a \in \mathbb{F}_{3^m}$. O

traço absoluto também é uma função sobrejetora e linear, e portanto sempre pode ser representado como uma matriz em alguma base.

Obter um ponto completo a partir de uma abscissa especificada (calculando a partir dela uma ordenada adequada) é tarefa comum em criptografia baseada em curvas elípticas. Tal técnica é empregada em todos os algoritmos adotados em padrões existentes (AMERICAN NATIONAL STANDARDS INSTITUTE – ANSI, 1999; IEEE P1363 Working Group, 2000; NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY – NIST, 2000).

Em corpos de característica 3, o cálculo de um cubo é uma operação linear (o cubo é calculado pelo mapa de Frobenius). Portanto, é mais vantajoso mapear um mensagem M a uma *ordenada* em vez de uma abscissa. Esta propriedade é explorada pela função *Map3Group* abaixo, que é probabilística no mesmo sentido da função original *Map2Group* 7.3.

Seja $y^2 = x^3 - x + b \equiv \mathcal{C}(x) + b$ a equação da curva elíptica utilizada sobre $\mathbb{F}_{3^m}$, onde $b = \pm 1$ é um polinômio cúbico, e seja J um parâmetro inteiro pequeno. Calcula-se *Map3Group* como segue:

Algoritmo *Map3Group*$_h(M, t)$

1. Inicializar um contador $j = 0$.
2. Mapear o par (j, M) ao par $h(j, M) = (y, \tau) \in \mathbb{F}_{3^m} \times \mathbb{F}_3$.
3. Calcular $u = y^2 - b$.
4. Resolver a equação cúbica $\mathcal{C}(x) = u$.
5. Se nenhuma solução for encontrada:
   (a) Se $j < 2^J$, incrementar j e tentar de novo a partir do passo 2.

(b) Caso contrário, declarar que M não pode ser mapeada num ponto da curva.

6. Caso contrário, usar τ para escolher entre as soluções x_0, x_1 e x_2, e retornar (x_τ, y).

Mais uma vez, a escolha entre as raízes no último passo acima requer uma convenção determinística arbitrária de rotular as soluções da equação cúbica. Uma convenção simples deriva da observação que, se x é uma solução da equação $\mathcal{C}(x) = u$, então as outras duas soluções são $x + 1$ e $x + 2$. Logo, as três soluções diferem entre si no termo independente de sua representação polinomial (ou, equivalentemente, no coeficiente de θ em base normal), e o seletor τ pode ser usado para escolher esse termo ou coeficiente.

A equação no passo 4 admite uma solução se, e somente se, $\mathrm{tr}(u) = 0$ (LIDL; NIEDERREITER, 1997, teorema 2.25). Isto acontece para 1/3 do número de elementos de $\mathbb{F}_{3^m}$, uma vez que o traço absoluto é uma função linear e sobrejetora.

Para cada valor de j, a probabilidade de que $h'(j||M)$ leve a um ponto G^* é a mesma de encontrar uma solução da equação cúbica no passo 4, isto é, 1/3 para valores de u uniformemente distribuídos. Logo, o número esperado de chamadas de h' é aproximadamente 3 (em vez de 2, como é o caso na função *Map2Group*), e a probabilidade de que uma dada mensagem M não seja mapeável a um ponto da curva é $(2/3)^{2^J} \leqslant \delta$.

O contador incrementado no passo 5 é limitado por 2^J. Se esse limite for atingido, a mensagem é declarada não mapeável a um ponto da curva. A probabilidade de falha δ é feita arbitrariamente pequena escolhendo J suficientemente grande, a saber, $J = \lceil \lg \lg(1/\delta) - \lg(\lg(3) - 1) \rceil \approx \lceil \lg \lg(1/\delta) \rceil + 1 = I + 1$, onde I é o tamanho do contador na função *Map2Group*.

A título de ilustração, comparamos o desempenho experimental das funções *Map2Group* e *Map3Group* para a curva $E : y^2 = x^3 - x + 1$ sobre $\mathbb{F}_{3^{97}}$. Nosso

Tabela 9: Tempos de execução de *Map2Group* e *Map3Group* em $\mathbb{F}_{3^{97}}$.

Map2Group	*Map3Group*
5.53 ms	0.054 ms

esquema aperfeiçoado executa notavelmente mais rápido (cerca de cem vezes) que a construção original, como se vê na tabela 8.9.1. Os resultados foram obtidos através de uma implementação em linguagem C++, executando numa plataforma Pentium III 1 GHz.

8.9.2 Resolução da equação cúbica

A complexidade computacional da função *Map3Group* deriva do cálculo do quadrado no passo 3 e da resolução da equação cúbica no passo 4. Calcular um quadrado obviamente não é mais complexo que $O(m^2)$. Mostraremos a seguir como resolver eficientemente em $\mathbb{F}_{3^m}$ a equação cúbica $\mathcal{C}(x) = u$.

Em base polinomial:

- **Cálculo do traço absoluto**

  Sabendo que o traço absoluto é uma forma linear $\mathrm{tr} : \mathbb{F}_{3^m} \to \mathbb{F}_3$, pode-se pré-calcular sua representação T (uma m-tupla de elementos de $\mathbb{F}_3$) na base escolhida, e a partir daí obter $\mathrm{tr}(u)$ como o produto interno $T \cdot u$ em tempo $O(m)$.

- **Resolução da equação $\mathcal{C}(x) = u$**

  A equação cúbica reduz-se a um sistema de equações lineares com coeficientes em $\mathbb{F}_3$, que pode ser resolvido em não mais que $O(m^2)$ passos. Atinge-se esta marca verificando inicialmente se o sistema tem solução, isto é, se $\mathrm{tr}(u) = 0$. Em caso afirmativo, como o posto de $\mathcal{C}$ é $m - 1$ obtém-se uma matriz inversível (fixa) A de dimensão $(m-1) \times (m-1)$ removendo uma linha e uma coluna da representação matricial de $\mathcal{C}$ na base dada, por exemplo, a linha e a coluna

correspondentes ao termo independente. Uma solução da equação cúbica é então dada por um elemento arbitrário $x_0 \in \mathbb{F}_3$ (o seletor de raiz) e pela solução (única) $x' = (x_1, \ldots, x_{m-1})$ do sistema $Ax = u'$, onde $u' = (u_1, \ldots, u_{m-1})$. Como A é fixa, sua inversa A^{-1} pode ser pré-calculada, e assim $x' = A^{-1}u'$ obtém-se em tempo $O(m^2)$.

Em base normal:

- **Cálculo do traço absoluto**

  O traço absoluto pode ser calculado facilmente numa base normal. Da definição de traço absoluto, calcular $\mathrm{tr}(u)$ resume-se a somar todos os coeficientes de u na base normal e multiplicar o resultado por $\mathrm{tr}(\theta)$. Obviamente, é vantajoso escolher θ de modo que $\mathrm{tr}(\theta) = 1$, evitando a multiplicação final.

- **Resolução da equação $\mathcal{C}(x) = u$**

  Usando base normal para representar elementos do corpo finito, o sistema linear acarretado pela equação cúbica é muito esparso (bidiagonal), e pode ser resolvido em tempo $O(m)$ com o seguinte algoritmo:

  Resolução da equação $\mathcal{C}(x) = u$:
  $x_0 \leftarrow$ seletor de raiz (um elemento arbitrário de $\mathbb{F}_3$)
  para $i \leftarrow 1, \ldots, m-1$ **faça** $\{$
  $\quad x_i \leftarrow x_{i-1} - u_i$
  $\}$
  se $x_{m-1} = x_0 + u_0$ **então devolva** $\{x\}$, **senão devolva** $\varnothing$

8.9.3 Prova de segurança

Mostraremos agora que nossa construção de *hash* é segura, no modelo de oráculo aleatório (BELLARE; ROGAWAY, 1993), contra fraude existencial sob ataques de

mensagem escolhida. O teorema abaixo e sua demonstração seguem de perto as idéias expostas em (BONEH; LYNN; SHACHAM, 2002, lema 4). Por simplicidade, discutimos aqui apenas o *hash* no grupo completo da curva elíptica (por oposição a um subgrupo próprio).

Definição 21. *Um algoritmo $\mathcal{F}$ é chamado* fraudador existencial *para um par de chaves (s, V) se for probabilisticamente capaz de produzir uma mensagem M e uma assinatura válida σ de M sob a chave pública V, sem conhecer a chave privada s.*

Admite-se que um fraudador existencial $\mathcal{F}$ tenha acesso a um *oráculo de hash* e um *oráculo de assinaturas*, isto é, que $\mathcal{F}$ possa obter, adaptativamente, certo número de valores de *hash* e assinaturas válidas (sob o mesmo par de chaves) de *outras* mensagens à sua escolha.

Definição 22. *Um fraudador existencial $\mathcal{F}$ $(t, q_H, q_S, \varepsilon)$-*quebra *um esquema de assinatura se $\mathcal{F}$ executa no máximo t passos, faz no máximo q_H consultas adaptativas a um oráculo de* hash *e no máximo q_S consultas adaptativas a um oráculo de assinatura, e produz com probabilidade não inferior a ε uma mensagem M e uma assinatura válida σ para M sob um dado par de chaves (s, V) aleatoriamente gerado.*

Definição 23. *Um esquema de assinatura é $(t, q_H, q_S, \varepsilon)$-*seguro *contra fraude existencial em ataques adaptativos de mensagem escolhida se nenhum fraudador puder $(t, q_H, q_S, \varepsilon)$-quebrá-lo.*

Teorema 7. *Admitindo que o esquema de assinatura BLS seja $(t, q_H, q_S, \varepsilon)$-seguro no grupo $\mathbb{G}$ ao usar uma função de* hash *aleatoriamente escolhida $h : \{0,1\}^* \to \mathbb{G}^*$, esse esquema será $(t - 2^J q_H, q_H, q_S, \varepsilon)$-seguro se a função de* hash *h for substituída por Map3Group$_{h'}$ onde h' é uma função de* hash *aleatoriamente escolhida $h' : \{0,1\}^* \to \mathbb{F}_{3^m} \times \mathbb{F}_3$.*

Demonstração. Suponhamos, por absurdo, que um algoritmo $\mathcal{F}'$ possa $(t, q_H, q_S, \varepsilon)$-quebrar o algoritmo BLS no grupo $\mathbb{G}$ caso a função de *hash* seja *Map3Group$_{h'}$*. Cons-

truiremos um algoritmo $\mathcal{F}$ que $(t + 2^J q_H, q_H, q_S, \varepsilon)$-quebra o algoritmo BLS quando h é um oráculo aleatório $h : \{0,1\}^* \rightarrow \mathbb{G}^*$.

O fraudador $\mathcal{F}$ executa $\mathcal{F}'$ como uma caixa preta. $\mathcal{F}$ usa seu próprio oráculo aleatório $h : \{0,1\}^* \rightarrow \mathbb{G}^*$ para simular (e camuflar) o comportamento *Map3Group*$_{h'}$ para $\mathcal{F}'$. $\mathcal{F}$ também mantém uma tabela s_{ij} com q_H linhas e 2^J colunas de elementos de $\mathbb{F}_{3^m} \times \mathbb{F}_3$.

Na inicialização, $\mathcal{F}$ preenche s_{ij} com elementos uniformemente selecionados de $\mathbb{F}_{3^m} \times \mathbb{F}_3$, e então executa $\mathcal{F}'$, mantendo um registro de todas as mensagens únicas M_i cujo *hash* com h' $\mathcal{F}'$ solicita. Quando $\mathcal{F}'$ pede o *hash* h' de uma mensagem (w, M_i) cuja parte M_i não foi previamente observada por $\mathcal{F}$ (e cuja parte w é uma seqüência arbitrária de J bits), $\mathcal{F}$ calcula $(x_i, y_i) = h(M_i) \in G^*$ e varre a linha s_{ij}, $0 \leqslant j < 2^J$. Note-se que a extensão média da varredura é de 3 passos. Para cada $(x, \tau) = \mathsf{s}_{ij}$, $\mathcal{F}$ resolve a equação cúbica no passo 4 de *Map3Group*, buscando pontos em G^*. Para o menor valor j tal que s_{ij} conduz a uma solução da equação cúbica, F substitui s_{ij} por um ponto diferente (x_i, τ_i) onde $\tau_i \in \mathbb{F}_3$ é escolhido de modo que (x_i, τ_i) corresponda a (x_i, y_i) no passo 6 de *Map3Group*$_{h'}$. Desta maneira, *Map3Group*$_{h'}(M_i) = h(M_i)$ como queríamos. Como o valor inicial aleatório de s_{ij} produziu espontaneamente uma solução da equação cúbica, a distribuição estatística de soluções não é afetada, e $\mathcal{F}'$ não tem como distinguir entre o valor original e seu substituto.

Uma vez que esta alteração preliminar de s_{ij} esteja completa, $\mathcal{F}$ será capaz de responder a consultas de *hash* h' feitas por $\mathcal{F}'$ para pares (w', M_i) retornando simplesmente $\mathsf{s}_{iw'}$. A função simulada h' vista por $\mathcal{F}'$ é estatisticamente indistinguível daquela que seria vista num ataque real. Portanto, se $\mathcal{F}'$ consegue quebrar o esquema de assinatura que usa *Map3Group*$_{h'}$, então $\mathcal{F}$, executando $\mathcal{F}'$ ao consultar h, tem sucesso com a mesma probabilidade, e sofre apenas um pequeno aumento em tempo de execução devido à manutenção de registros de operação. □

Observação: se o *hash* for feito num subgrupo próprio da curva elíptica, o sistema será $(t - 2^J q_H \lg n, q_H, q_S, \varepsilon)$-seguro (onde n é a ordem da curva), uma vez que cada passo de inicialização envolverá uma multiplicação pelo cofator apropriado.

8.9.4 Esquema BLS modificado

Descreveremos agora as modificações necessárias ao algoritmo BLS original descrito na seção 7.2.1 para usar a função *Map3Group*. A geração de pares de chaves não sofre alterações.

8.9.4.1 Assinatura

Para assinar uma mensagem $M \in \{0,1\}^*$, mapeia-se M a um ponto $P_M \in \langle P \rangle$ usando *Map3Group*. Seja $S_M = (x_S, y_S) = [s]P_M$. A assinatura σ é o par (t_S, y_S), onde $t_S \in \mathbb{F}_3$ é escolhido segundo a mesma convenção do seletor de raízes usado no passo 6 da função *Map3Group*, isto é, se $x_S = (x_0, \ldots x_{m-1})$ na representação utilizada, então $t_S = x_0$.

8.9.4.2 Verificação

Dada uma chave pública (m, P, V), uma mensagem M, e uma assinatura (t_S, y_S) executam-se os seguintes passos:

1. Encontrar um ponto $S = (x_S, y_S) \in E(\mathbb{F}_{3^m})$ de ordem r satisfazendo $y_S = y_\sigma$ e $x_S = (x_0, \ldots x_{m-1})$ onde $x_0 = t_S$. Se nenhum ponto assim existir, rejeitar a assinatura.

2. Calcular os emparelhamentos de Tate $u \leftarrow e(P, \phi(S))$ e $v \leftarrow e(V, \phi(h(M)))$.

3. Aceitar a assinatura se, e somente se, $u = v$.

Ao contrário do esquema original, aqui não é necessário testar se $u^{-1} = v$, pois o ponto S é univocamente determinado devido à presença de t_S na assinatura. A desvan-

tagem de um tamanho ligeiramente maior neste esquema é de certo modo compensada pela capacidade proporcionalmente maior (por um fator 2) de detectar tentativas de fraude, uma vez que agora apenas o ponto exato resultante do processo de assinatura é aceito como válido.

Por outro lado, em certas aplicações com sérias restrições de espaço pode ser necessário manter apenas a ordenada y_S na assinatura. Isto é possível, mas acarreta um preço em desempenho. Suponhamos que um emparelhamento $e(P, V)$ precise ser calculado num sistema BLS onde só se conhece a abscissa x_V de $V = (x_V, y_V)$, isto é, onde V e $-V$ são igualmente aceitáveis para uma verificação de assinatura. Evita-se nesse sistema o custo de calcular dois emparelhamentos usando a propriedade que $e(P, -V) = e(P, V)^{-1}$, conforme destacado em (BONEH; LYNN; SHACHAM, 2002, seção 5.1). Por outro lado, num sistema onde só se conhece a ordenada y_V parece inevitável calcular *três* emparelhamentos $e(P, V_i)$, $i \in \{0, 1, 2\}$, correspondentes aos três possíveis valores da abscissa, isto é, $x_i = x_V + i$, $i \in \{0, 1, 2\}$. No entanto, verifica-se facilmente que $V_0 + V_1 + V_2 = O$, e portanto $e(P, V_2) = [e(P, V_0) \cdot e(P, V_1)]^{-1}$. Além disso, como P é o mesmo, e os pontos V_0 e V_1 compartilham a mesma ordenada y_V, esses dois emparelhamentos podem ser calculados concomitantemente, com uma redução modesta de desempenho em comparação com o cálculo de um único emparelhamento.

9 CONCLUSÕES

Conforme vimos, é possível construir marcas d'água topológicas seguras capazes de localizar alterações em imagens e outros sinais digitais, mas a construção exige cuidados de natureza criptográfica além daqueles de índole específica de processamento de sinais. Ao mesmo tempo, a investigação de linhas de ataque pode levar naturalmente à definição de algoritmos sólidos.

Seguindo esse caminho, elaboramos diversos ataques contra esquemas existentes de marca d'água topológica (inclusive contra versões preliminares de nossos próprios esquemas), e como resultado positivo propusemos um algoritmo específico (HBC2) que se mostrou, segundo nossa análise, resistente a todos os ataques conhecidos. Um problema ainda aberto é a definição de esquemas topológicos aplicáveis em tempo real sem a necessidade de hardware dedicado, mas nosso teorema sobre a otimalidade de HBC2 parece indicar que técnicas radicalmente diferentes das usadas até aqui serão necessárias em tentativas de solucionar esse problema.

Fomos bem sucedidos em resolver, com nosso algoritmo para construção de curvas contendo subgrupos com grau de imersão arbitrário, o problema em aberto mencionado em (BONEH; LYNN; SHACHAM, 2002, seção 3.5). Contudo, nossa solução reconhecidamente abre um problema relacionado, qual seja o de construir curvas de ordem *prima* com grau de imersão arbitrário. A solução deste problema é de fundamental importância para a utilidade futura do algoritmo BLS, uma vez que pequenos aumentos no tamanho do grupo básico acarretam aumentos substancialmente maiores

no grau de imersão necessário para manter um nível proporcional de segurança.

Nossa abordagem acerca da implementação do emparelhamento de Tate demonstrou a viabilidade prática não só das marcas d'água topológicas com alta resolução (ou pequenos volumes de dados hospedados), mas de uma classe enorme e ainda em expansão de algoritmos criptográficos, a saber, a dos sistemas baseados em emparelhamentos. A relevância dessa classe de algoritmos, e portanto das nossas contribuições, torna-se patente quando se observa que inúmeros problemas até então abertos (como sistemas criptográficos baseados em identidade) foram resolvidos elegantemente através de emparelhamentos.

Uma linha interessante de pesquisa futura é a aplicação das diversas técnicas aqui apresentadas a curvas algébricas mais gerais; por exemplo um algoritmo rápido para extrair raízes n-ésimas seria útil para curvas superelípticas. Investigações sobre os motivos algébricos que levam a operações lineares na lei de grupo de variedades abelianas gerais também seria de grande interesse, bem como as possibilidades de otimizar o cálculo do emparelhamento de Tate nessas variedades. A existência ou não de fatores irrelevantes até mesmo em curvas hiperelípticas (de gênero $g \geqslant 2$) constitui um problema aberto.

Elencamos agora sistematicamente nossas contribuições originais, e para cada uma, sugerimos alguns tópicos de futuras pesquisas relacionadas a essas contribuições. A relevância de nossos resultados é atestada pelo número apreciável de referências a eles na literatura existente (cfr. seção 1.2).

Contribuição original	**Sugestão de pesquisa**
Conceituação de ataque de transplante e de ataque de aniversário avançado, aplicáveis a quase todos os esquemas topológicos de marca d'água.	Elaborar novos ataques, particularmente contra o modo HBC2 (ou mostrar que não se aplicam).

Definição de encadeamento de blocos de *hash* (HBC2), o primeiro esquema topológico de marca d'água resistente aos ataque de transplante e de aniversário avançado.	Formular uma demonstração formal de segurança para o modo HBC2. Um obstáculo conhecido é quantificar a segurança de HBC2 em termos da resolução na localização de alterações, tornando a prova de segurança dependente de uma métrica de resolução. Outra linha de pesquisa é detalhar a adaptação do modo HBC2 a sinais N-dimensionais, com particular atenção à escolha da métrica de resolução mais adequada.
Apresentação de um algoritmo com complexidade $O(n^2 \log n)$, onde $n = \log q$, para a extração de raízes quadradas em determinados corpos finitos $\mathbb{F}_q$.	Estender o algoritmo para raízes superiores e/ou para outros corpos finitos, e diminuir a complexidade para $O(n^2)$ mantendo o algoritmo prático.
Definição de algoritmos com complexidade $O(m)$ para triplicação de ponto e $O(m^2)$ para multiplicação por escalar em curvas supersingulares sobre corpos ternários $\mathbb{F}_{3^m}$.	Determinar que outras curvas ou variedades abelianas admitem operações eficientes e propor algoritmos para essas operações.
Descrição de um método para construir curvas elípticas ordinárias (não-supersingulares) contendo subgrupos com grau de imersão arbitrário.	Propor um algoritmo análogo que produza curvas de ordem prima. Embora essencial para certos sistemas baseados em emparelhamentos, a solução deste problema parece especialmente difícil, dada a raridade de curvas adequadas mesmo com ordem composta.

Elaboração de um algoritmo para selecionar geradores de grupos amigáveis a emparelhamentos, tornando mais eficientes (em tempo e espaço) várias operações independentes de emparelhamentos.	Definir emparelhamentos com valores intrinsecamente mais compactos. Os métodos atualmente conhecidos permitem comprimir apenas os pontos das curvas elípticas subjacentes, e recorrem a valores extensos pelo menos em cálculos intermediários.
Aperfeiçoamento do algoritmo de Miller para calcular o emparelhamento de Tate, na forma de uma variante eficiente e determinística que evita muitas operações irrelevantes do algoritmo convencional. O resultado, chamado algoritmo BKLS, torna viável uma família inteira de algoritmos baseados em emparelhamentos.	Formular otimizações análogas para curvas hiperelípticas e outras variedades abelianas, e propor novas classes de emparelhamentos (por exemplo, formas multilineares com número arbitrário de argumentos).

REFERÊNCIAS

AJTAI, M.; DWORK, C. A public-key cryptosystem with worst case/average-case equivalence. In: ACM SYMPOSIUM ON THEORY OF COMPUTING (STOC). *Proceedings*. El Paso, USA: ACM Press, 1997. p. 284–293.

AL-RIYAMI, S. S.; PATERSON, K. G. *Authenticated Three Party Key Agreement Protocols from Pairings*. 2002. Cryptology ePrint Archive, Report 2002/035. Disponível em: http://eprint.iacr.org/2002/035. Acesso em: 29 de outubro de 2003.

____. *Certificateless Public Key Cryptography*. 2003. Cryptology ePrint Archive, Report 2003/126. Disponível em: http://eprint.iacr.org/2003/126. Acesso em: 29 de outubro de 2003.

AMERICAN NATIONAL STANDARDS INSTITUTE – ANSI. *Public Key Cryptography for the Financial Services Industry: the Elliptic Curve Digital Signature Algorithm (ECDSA) – ANSI X9.62*. [S.l.], 1999. Also published in FIPS 186-2.

BALASUBRAMANIAN, R.; KOBLITZ, N. The improbability that an elliptic curve has subexponential discrete log problem under the Menezes-Okamoto-Vanstone algorithm. *Journal of Cryptology*, Springer-Verlag, New York, USA, v. 11, n. 2, p. 141–145, 1998.

BALFANZ, D. et al. Secret handshakes from pairing-based key agreements. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY – S&P'2003. *Proceedings*. Berkeley, USA: IEEE Computer Society, 2003. p. 180–196.

BARAK, B. et al. On the (im)possibility of obfuscating programs. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2001. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 2139), p. 1–18.

BARRETO, P. S. L. M.; KIM, H. Y. Pitfalls in public key watermarking. In: BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING – SIBGRAPI. *Proceedings*. Campinas, Brasil, 1999. p. 241–242.

____. *Fast hashing onto elliptic curves over fields of characteristic 3*. 2001. Cryptology ePrint Archive, Report 2001/098. Disponível em: http://eprint.iacr.org/2001/098. Acesso em: 29 de outubro de 2003.

BARRETO, P. S. L. M. et al. Efficient algorithms for pairing-based cryptosystems. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2002. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2442), p. 377–387.

BARRETO, P. S. L. M.; KIM, H. Y.; RIJMEN, V. Um modo de operação de funções de hashing para localizar alterações em dados digitalmente assinados. In: SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES (SBrT). *Anais*. Gramado, Brasil, 2000. #5150124. 1 CD-ROM.

____. Toward a secure public-key blockwise fragile authentication watermarking. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Thessaloniki, Greece, 2001. v. 2, p. 494–497.

____. Toward a secure public-key blockwise fragile authentication watermarking. *IEE Proceedings on Vision, Image and Signal Processing*, v. 149, n. 2, p. 57–62, 2002. Este artigo é a versão completa de (BARRETO; KIM; RIJMEN, 2001).

BARRETO, P. S. L. M.; LYNN, B.; SCOTT, M. Constructing elliptic curves with prescribed embedding degrees. In: SECURITY IN COMMUNICATION NETWORKS – SCN'2002. *Proceedings*. Amalfi, Italy: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2576), p. 263–273.

____. Efficient implementation of pairing-based cryptosystems. *Journal of Cryptology*, Springer-Verlag, New York, USA, 2003. No prelo.

____. On the selection of pairing-friendly groups. In: SELECTED AREAS IN CRYPTOGRAPHY – SAC'2003. *Proceedings*. Ottawa, Canada: Springer-Verlag, 2003. (Lecture Notes in Computer Science). No prelo.

BARRETO, P. S. L. M.; RIJMEN, V. The ANUBIS block cipher. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Belgium: NESSIE Consortium, 2000.

____. The KHAZAD legacy-level block cipher. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Belgium: NESSIE Consortium, 2000.

____. The WHIRLPOOL hashing function. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Belgium: NESSIE Consortium, 2000.

BARRETO, P. S. L. M. et al. Improved SQUARE attacks against reduced-round Hierocrypt. In: FAST SOFTWARE ENCRYPTION – FSE'2001. *Proceedings*. Yokohama, Japan: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2355), p. 165–173.

BARRETO, P. S. L. M.; VOLOCH, J. F. *Efficient Computation of Roots in Finite Fields*. 2003. Preprint.

BELLARE, M.; ROGAWAY, P. Random oracles are practical: A paradigm for designing efficient protocols. In: ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY. *Proceedings*. Fairfax, USA: ACM Press, 1993. p. 62–73.

BERTONI, G. et al. Efficient GF(p^m) arithmetic architectures for cryptographic applications. In: TOPICS IN CRYPTOLOGY – CT-RSA'2003. *Proceedings*. San Francisco, USA: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2612), p. 158–175.

BHATTACHARJEE, S.; KUTTER, M. Compression tolerant image authentication. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Chicago, USA, 1998. v. 1, p. 435–439.

BINNEY, J.; MERRIFIELD, M. *Galactic Astronomy*. New Jersey, USA: Princeton University Press, 1998.

BLAKE, I.; SEROUSSI, G.; SMART, N. P. *Elliptic Curves in Cryptography*. London, UK: Cambridge University Press, 1999.

BLEICHENBACHER, D. *Attack against the pseudo-random number generator prescribed for DSA digital signatures*. 2001. Não publicado. Disponível em: http://www.lucent.com/press/0201/010205.bla.html. Acesso em: 29 de outubro de 2003.

BONEH, D.; FRANKLIN, M. Identity-based encryption from the Weil pairing. *SIAM Journal of Computing*, v. 32, n. 3, p. 586–615, 2003.

BONEH, D.; LYNN, B.; SHACHAM, H. Short signatures from the Weil pairing. In: ADVANCES IN CRYPTOLOGY – ASIACRYPT'2001. *Proceedings*. Gold Coast, Australia: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2248), p. 514–532.

BONEH, D.; MIRONOV, I.; SHOUP, V. A secure signature scheme from bilinear maps. In: TOPICS IN CRYPTOLOGY – CT-RSA'2003. *Proceedings*. San Francisco, USA: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2612), p. 98–110.

BOYD, C.; MAO, W.; PATERSON, K. G. Deniable authenticated key establishment for internet protocols. In: 11TH INTERNATIONAL WORKSHOP ON SECURITY PROTOCOLS. *Proceedings*. Cambridge, UK: Springer-Verlag, 2003. (Lecture Notes in Computer Science). No prelo.

BOYEN, X. Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2003. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 2003. (Lecture Notes in Computer Science). No prelo.

BREZING, F.; WENG, A. *Elliptic curves suitable for pairing based cryptography*. 2003. Cryptology ePrint Archive, Report 2003/143. Disponível em: http://eprint.iacr.org/2003/143. Acesso em: 29 de outubro de 2003.

CHAE, J.; MUKHERJEE, D.; MANJUNATH, B. Color image embedding using multidimensional lattice structures. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Chicago, USA, 1998. v. 1, p. 460–464.

CHOI, Y.; AIZAWA, K. Watermarking using inter-block correlation: Extension to JPEG coded domain. *IEICE Transactions on Fundamentals*, E84-A, n. 5, p. 893–897, 2001.

CHOR, B.; RIVEST, R. A knapsack-type cryptosystem based on arithmetic in finite fields. *IEEE Transactions on Information Theory*, v. 34, n. 5, p. 901–909, 1988.

COHEN, H. *A Course in Computational Algebraic Number Theory*. Berlin, Germany: Springer-Verlag, 1993.

COOK, S. A. The complexity of theorem proving procedures. In: ACM SYMPOSIUM ON THEORY OF COMPUTING (STOC). *Proceedings*. Shaker Heights, USA: ACM Press, 1971. p. 151–158.

COPPERSMITH, D. Fast evaluation of logarithms in fields of characteristics two. *IEEE Transactions on Information Theory*, v. 30, p. 587–594, 1984.

COURTOIS, N.; FINIASZ, M.; SENDRIER, N. How to achieve a McEliece-based digital signature scheme. In: ADVANCES IN CRYPTOLOGY – ASIACRYPT'2001. *Proceedings*. Gold Coast, Australia: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2248), p. 157–174.

DEGUILLAUME, F.; VOLOSHYNOVSKIY, S.; PUN, T. Hybrid robust watermarking resistant against copy attack. In: EUROPEAN SIGNAL PROCESSING CONFERENCE (EUSIPCO'2002). *Proceedings*. Toulouse, France, 2002. No prelo.

DUPONT, R.; ENGE, A.; MORAIN, F. *Building curves with arbitrary small MOV degree over finite prime fields*. 2002. Cryptology ePrint Archive, Report 2002/094. Disponível em: http://eprint.iacr.org/2002/094. Acesso em: 29 de outubro de 2003.

DUURSMA, I.; LEE, H.-S. *Tate-pairing implementations for tripartite key agreement*. 2003. Cryptology ePrint Archive, Report 2003/053. Disponível em: http://eprint.iacr.org/2003/053. Acesso em: 29 de outubro de 2003.

EISENTRAEGER, K.; LAUTER, K.; MONTGOMERY, P. Fast elliptic curve arithmetic and improved Weil pairing evaluation. In: TOPICS IN CRYPTOLOGY – CT-RSA'2003. *Proceedings*. San Francisco, USA: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2612), p. 343–354.

FREY, G.; MÜLLER, M.; RÜCK, H. The Tate pairing and the discrete logarithm applied to elliptic curve cryptosystems. *IEEE Transactions on Information Theory*, v. 45, n. 5, p. 1717–1719, 1999.

FREY, G.; RÜCK, H.-G. A remark concerning m-divisibility and the discrete logarithm problem in the divisor class group of curves. *Mathematics of Computation*, v. 62, p. 865–874, 1994.

FRIDRICH, J.; GOLJAN, M.; BALDOZA, A. C. New fragile authentication watermark for images. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Vancouver, Canada, 2000. v. 1, p. 446–449.

FRIDRICH, J.; GOLJAN, M.; MEMON, N. D. Further attacks on the Yeung-Mintzer fragile watermark. In: SPIE INTERNATIONAL CONFERENCE ON SECURITY AND WATERMARKING OF MULTIMEDIA CONTENTS. *Proceedings*. San Jose, USA, 2000. v. 3971, p. 428–437.

FRIEDMAN, G. The trustworthy digital camera – restoring credibility to the photographic image. *IEEE Transactions on Consumer Electronics*, v. 39, p. 905–910, 1993.

GAGNÉ, M. *Applications of Bilinear Maps in Cryptography*. 126 p. Dissertação (Mestrado) — University of Waterloo, Waterloo, Ontario, Canada, 2002.

GALBRAITH, S. Supersingular curves in cryptography. In: ADVANCES IN CRYPTOLOGY – ASIACRYPT'2001. *Proceedings*. Gold Coast, Australia: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2248), p. 495–513.

____. *Pairings in elliptic curve cryptography*. 2003. Mensagem pessoal (correio eletrônico).

GALBRAITH, S.; HARRISON, K.; SOLDERA, D. Implementing the Tate pairing. In: ALGORITHM NUMBER THEORY SYMPOSIUM – ANTS V. *Proceedings*. Sydney, Australia: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2369), p. 324–337.

GALBRAITH, S.; HESS, F.; SMART, N. P. Extending the GHS Weil descent attack. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'2002. *Proceedings*. Amsterdam, The Netherlands: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2332), p. 29–44.

GALBRAITH, S.; SMART, N. P. A cryptographic application of Weil descent. In: 7TH IMA INTERNATIONAL CONFERENCE ON CODES AND CRYPTOGRAPHY. *Proceedings*. Cirencester, UK: Springer-Verlag, 1999. (Lecture Notes in Computer Science, v. 1746), p. 191–200.

GALLANT, R.; LAMBERT, R.; VANSTONE, S. A. Improving the parallelized Pollard lambda search on binary anomalous curves. *Mathematics of Computation*, v. 69, p. 1699–1705, 2000.

GAUDRY, P.; HESS, F.; SMART, N. P. Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Cryptology*, Springer-Verlag, New York, USA, v. 15, p. 19–46, 2002.

GAUSS, C. F. *Disquisitiones Arithmeticae*. London, UK: Yale University Press, 1965.

GENTRY, C. et al. Cryptanalysis of the NTRU signature scheme NSS. In: ADVANCES IN CRYPTOLOGY – ASIACRYPT'2001. *Proceedings*. Gold Coast, Australia: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2248), p. 1–20.

GENTRY, C.; SZYDLO, M. Cryptanalysis of the revised NTRU signature scheme. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'2002. *Proceedings*. Amsterdam, The Netherlands: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2332), p. 299–320.

GORDON, D. M. Discrete logarithms in $\mathrm{GF}(p)$ using the number field sieve. *SIAM Journal on Discrete Mathematics*, v. 6, p. 124–138, 1993.

HARRISON, K.; PAGE, D.; SMART, N. P. Software implementation of finite fields of characteristic three. *LMS Journal Computation and Mathematics*, London Mathematical Society, London, UK, v. 5, p. 181–193, 2002.

HARTUNG, F.; GIROD, B. Fast public-key watermarking of compressed video. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Santa Barbara, USA, 1997. v. 1, p. 528–531.

HOFFMAN, K.; KUNZE, R. *Linear Algebra*. 2nd. ed. New Jersey, USA: Prentice Hall, 1971.

HOFFSTEIN, J. et al. *NTRUSign: Digital Signatures Using the NTRU Lattice*. 2001. Advances in Cryptology – Asiacrypt'2001, rump session. Disponível em: http://www.ntru.com/NTRUFTPDocsFolder/NTRUSign.pdf. Acesso em: 29 de outubro de 2003.

HOFFSTEIN, J.; PIPHER, J.; SILVERMAN, J. H. NSS: The NTRU signature scheme. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'2001. *Proceedings*. Innsbruck, Austria: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 2045), p. 211–228.

HOLLIMAN, M.; MEMON, N. Counterfeiting attacks on oblivious block-wise independent invisible watermarking schemes. *IEEE Transactions on Image Processing*, v. 9, n. 3, p. 432–441, 2000.

HU, F.; WU, C.-H.; IRWIN, J. D. *A New Forward Secure Signature Scheme using Bilinear Maps*. 2003. Cryptology ePrint Archive, Report 2003/188. Disponível em: http://eprint.iacr.org/2003/188. Acesso em: 29 de outubro de 2003.

IEEE P1363 Working Group. *Standard Specifications for Public-Key Cryptography – IEEE Std 1363-2000*. [S.l.], 2000.

IRELAND, K.; ROSEN, M. *A Classical Introduction to Modern Number Theory*. 2nd. ed. Berlin, Germany: Springer-Verlag, 1990.

ITOH, T.; TEECHAI, O.; TSUJII, S. A fast algorithm for computing multiplicative inverses in $\mathrm{GF}(2^m)$ using normal bases. *Information and Computation*, v. 78, p. 171–177, 1988.

IZU, T.; TAKAGI, T. Efficient computations of the Tate pairing for the large MOV degrees. In: 5TH INTERNATIONAL CONFERENCE ON INFORMATION SECURITY AND CRYPTOLOGY (ICISC 2002). *Proceedings*. Seoul, Korea: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2587), p. 283–297.

JACOBSON, M.; MENEZES, A. J.; STEIN, A. Solving elliptic curve discrete logarithm problems using Weil descent. *Journal of the Ramanujan Mathematical Society*, v. 16, p. 231–260, 2001.

JOUX, A.; NGUYEN, K. *Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups*. 2001. Cryptology ePrint Archive, Report 2001/003. Disponível em: http://eprint.iacr.org/2001/003. Acesso em: 29 de outubro de 2003.

KALKER, T.; LINNARTZ, J.-P.; DIJK, M. van. Watermark estimation through detector analysis. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Chicago, USA, 1998. v. 1, p. 425–429.

KIM, H. Y.; BARRETO, P. S. L. M. Fast binary image resolution increasing by k-nearest neighbor learning. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Vancouver, Canada, 2000. v. 2, p. 327–330.

KNUDSEN, L. R.; WAGNER, D. A. Integral cryptanalysis. In: FAST SOFTWARE ENCRYPTION – FSE'2002. *Proceedings*. Leuven, Belgium: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2365), p. 112–127.

KOBLITZ, N. An elliptic curve implementation of the finite field digital signature algorithm. In: ADVANCES IN CRYPTOLOGY – CRYPTO'98. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 1998. (Lecture Notes in Computer Science, v. 1462), p. 327–337.

LAY, G. J.; ZIMMER, H. Constructing elliptic curves with given group order over large finite fields. In: ALGORITHM NUMBER THEORY SYMPOSIUM – ANTS V. *Proceedings*. Sydney, Australia: Springer-Verlag, 1994. (Lecture Notes in Computer Science, v. 877), p. 250–263.

LENSTRA, A. K.; VERHEUL, E. R. The XTR public key system. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2000. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 2000. (Lecture Notes in Computer Science, v. 1880), p. 1–19.

LI, C. T.; LOU, D. C.; CHEN, T. H. Image authentication and integrity verification via content-based watermarks and a public key cryptosystem. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Vancouver, Canada, 2000. v. 3, p. 694–697.

LIDL, R.; NIEDERREITER, H. *Finite Fields*. 2nd. ed. Cambridge, UK: Cambridge University Press, 1997. (Encyclopedia of Mathematics and its Applications, 20).

LIN, C.-Y.; WU, T.-C. *An identity-based ring signature scheme from bilinear pairings*. 2003. Cryptology ePrint Archive, Report 2003/117. Disponível em: http://eprint.iacr.org/2003/117. Acesso em: 29 de outubro de 2003.

MACLANE, S.; BIRKHOFF, G. *Algebra*. 3rd. ed. New York, USA: Chelsea Publishing Company, 1993.

MACQ, B. M.; QUISQUATER, J.-J. Cryptology for digital TV broadcasting. *Procedings of the IEEE*, v. 83, n. 6, p. 944–957, 1995.

MALONE-LEE, J.; SMART, N. P. Modifications of ECDSA. In: SELECTED AREAS IN CRYPTOGRAPHY – SAC'2002. *Proceedings*. Newfoundland, Canada: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2595), p. 1–12.

MARVEL, L. M.; RETTER, C. T.; BONCELET-JR., C. G. Hiding information in images. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Chicago, USA, 1998. v. 2, p. 396–398.

MENEZES, A. J. *Elliptic Curve Public Key Cryptosystems*. Boston, USA: Kluwer Academic Publishers, 1993.

MENEZES, A. J.; OKAMOTO, T.; VANSTONE, S. A. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, v. 39, p. 1639–1646, 1993.

MENEZES, A. J.; OORSCHOT, P. C. van; VANSTONE, S. A. *Handbook of Applied Cryptography*. Boca Raton, USA: CRC Press, 1999.

MENEZES, A. J.; QU, M. Analysis of the Weil descent attack of Gaudry, Hess and Smart. In: TOPICS IN CRYPTOLOGY – CT-RSA'2001. *Proceedings*. San Francisco, USA: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 2020), p. 308–318.

MENEZES, A. J.; VANSTONE, S. A. The implementation of elliptic curve cryptosystems. In: ADVANCES IN CRYPTOLOGY – AUSCRYPT'90. *Proceedings*. Sydney, Australia: Springer-Verlag, 1990. (Lecture Notes in Computer Science, v. 453), p. 2–13.

MERKLE, R. C.; E.HELLMAN, M. Hiding information and signatures in trapdoor knapsacks. *IEEE Transactions on Information Theory*, v. 24, p. 525–530, 1978.

MILLER, V. *Short programs for functions on curves*. 1986. Não publicado. Disponível em: http://crypto.stanford.edu/miller/miller.pdf. Acesso em: 29 de outubro de 2003.

MIYAJI, A.; NAKABAYASHI, M.; TAKANO, S. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Transactions on Fundamentals*, E84-A, n. 5, p. 1234–1243, 2001.

MORAIN, F. Building cyclic elliptic curves modulo large primes. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'91. *Proceedings*. Brighton, UK: Springer-Verlag, 1991. (Lecture Notes in Computer Science, v. 547), p. 328–336.

NAKAHARA-JR., J. et al. Square attacks on reduced-round PES and IDEA block ciphers. In: 23RD SYMPOSIUM ON INFORMATION THEORY IN THE BENELUX. *Proceedings*. Louvain-la-Neuve, Belgium, 2002.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY – NIST. *Federal Information Processing Standard (FIPS 186-2) – Digital Signature Standard (DSS)*. [S.l.], January 2000.

____. *Federal Information Processing Standard (FIPS 180-2) – Secure Hash Standard (SHS)*. [S.l.], August 2002.

NGUYEN, P.; STERN, J. Cryptanalysis of the Ajtai-Dwork cryptosystem. In: ADVANCES IN CRYPTOLOGY – CRYPTO'98. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 1998. (Lecture Notes in Computer Science, v. 1462), p. 223–242.

NISHIMURA, K.; SIBUYA, M. Probability to meet in the middle. *Journal of Cryptology*, Springer-Verlag, New York, USA, v. 2, n. 1, p. 13–22, 1990.

NYBERG, K.; RUEPPEL, R. A new signature scheme based on the DSA giving message recovery. In: ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY. *Proceedings*. Fairfax, USA: ACM Press, 1993. p. 58–61.

OKAMOTO, T.; POINTCHEVAL, D. The gap-problems: a new class of problems for the security of cryptographic schemes. In: PRACTICE AND THEORY IN PUBLIC KEY CRYPTOGRAPHY – PKC'2001. *Proceedings*. Cheju Island, Korea: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 1992), p. 104–118.

OORSCHOT, P. C. van; WIENER, M. Parallel collision search with applications to hash functions and discrete logarithms. In: ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY. *Proceedings*. Fairfax, USA: ACM Press, 1994. p. 210–218.

PAGE, D.; SMART, N. Hardware implementation of finite fields of characteristic three. In: WORKSHOP ON CRYPTOGRAPHIC HARDWARE AND EMBEDDED SYSTEMS – CHES'2002. *Proceedings*. Redwood Shores, USA: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2523), p. 529–539.

PATARIN, J. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'96. *Proceedings*. Zaragoza, Spain: Springer-Verlag, 1996. (Lecture Notes in Computer Science, v. 1070), p. 33–48.

PATARIN, J.; COURTOIS, N.; GOUBIN, L. Quartz, 128-bit long digital signatures. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Belgium: NESSIE Consortium, 2000.

PINTSOV, L. A.; VANSTONE, S. A. Postal revenue collection in the digital age. In: FINANCIAL CRYPTOGRAPHY'2000. *Proceedings*. Anguilla, British West Indies: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 1962), p. 105–120.

PIVA, A. et al. DCT-based watermark recovering without resorting to the uncorrupted original image. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Santa Barbara, USA, 1997. v. 1, p. 520–523.

POHLIG, S.; HELLMAN, M. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, v. 24, p. 106–110, 1978.

POLLARD, J. M. Monte Carlo methods for index computation (mod p). *Mathematics of Computation*, v. 32, p. 918–924, 1978.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. M. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, v. 21, p. 120–126, 1978.

RUBIN, K.; SILVERBERG, A. Supersingular abelian varieties in cryptology. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2002. *Proceedings*. Santa Barbara,

USA: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2442), p. 336–353.

SATOH, T.; ARAKI, K. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Commentarii Mathematici Universitatis Sancti Pauli*, v. 47, p. 81–92, 1998.

SCHNORR, C. Efficient signature generation for smart cards. *Journal of Cryptology*, Springer-Verlag, New York, USA, v. 4, n. 3, p. 161–174, 1991.

SCHNORR, C.; HÖRNER, H. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'95. *Proceedings*. Saint Malo, France: Springer-Verlag, 1995. (Lecture Notes in Computer Science, v. 921), p. 1–12.

SHOUP, V. OAEP reconsidered. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2001. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 2139), p. 239–259.

SILVERMAN, J. H. *The Arithmetic of Elliptic Curves*. Berlin, Germany: Springer-Verlag, 1986. (Graduate Texts in Mathematics, 106).

SIMMONS, G. J. Subliminal communication is easy using the DSA. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'93. *Proceedings*. Lofthus, Norway: Springer-Verlag, 1994. (Lecture Notes in Computer Science, v. 765), p. 218–232.

SMART, N. P. *The Algorithmic Resolution of Diophantine Equations*. London, UK: Cambridge University Press, 1998. (London Mathematical Society Student Texts, 41).

____. An identity based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters*, v. 38, p. 630–632, 2002.

SMART, N. P.; WESTWOOD, J. Point multiplication on ordinary elliptic curves over fields of characteristic three. *Applicable Algebra in Engineering, Communication and Computing*, v. 13, p. 485–497, 2003.

SOLINAS, J. *Generalized Mersenne numbers*. 1999. Technical Report CORR-39, Department of C&O, University of Waterloo.

STINSON, D. R. *Cryptography: Theory and Practice*. 2nd. ed. Boca Raton, USA: Chapman & Hall/CRC Press, 2002.

SUN, H.-M.; HSIEH, B.-T. *Security Analysis of Shim's Authenticated Key Agreement Protocols from Pairings*. 2003. Cryptology ePrint Archive, Report 2003/113. Disponível em: http://eprint.iacr.org/2003/113. Acesso em: 29 de outubro de 2003.

TZANAKIS, N. Solving elliptic diophantine equations by estimating linear forms in elliptic logarithms: the case of quartic equations. *Acta Arithmetica*, v. 75, p. 165–190, 1996.

VAUDENAY, S. Hidden collisions on DSS. In: ADVANCES IN CRYPTOLOGY – CRYPTO'96. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 1996. (Lecture Notes in Computer Science, v. 1109), p. 83–88.

____. Cryptanalysis of the Chor-Rivest cryptosystem. In: ADVANCES IN CRYPTOLOGY – CRYPTO'98. *Proceedings*. Santa Barbara, USA: Springer-Verlag, 1998. (Lecture Notes in Computer Science, v. 1462), p. 243–256.

VERHEUL, E. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. In: ADVANCES IN CRYPTOLOGY – EUROCRYPT'2001. *Proceedings*. Innsbruck, Austria: Springer-Verlag, 2001. (Lecture Notes in Computer Science, v. 2045), p. 195–210.

WALLACE, G. K. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics*, v. 38, n. 1, p. 18–34, 1992.

WIENER, M. J.; ZUCCHERATO, R. Faster attacks on elliptic curve cryptosystems. In: SELECTED AREAS IN CRYPTOGRAPHY – SAC'98. *Proceedings*. Kingston, Canada: Springer-Verlag, 1999. (Lecture Notes in Computer Science, v. 1556), p. 190–200.

WILLIAMS, H. C. A modification of the RSA public-key encryption procedure. *IEEE Transactions on Information Theory*, v. 26, n. 6, p. 726–729, 1980.

WONG, P. W. A public key watermark for image verification and authentication. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Chicago, USA, 1998. v. 1, p. 455–459.

WONG, P. W.; MEMON, N. Secret and public key image watermarking schemes for image authentication and ownership verification. *IEEE Transactions on Image Processing*, v. 10, n. 10, p. 1593–1601, 2001.

WU, M.; LIU, B. Watermarking for image authentication. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Chicago, USA, 1998. v. 2, p. 437–441.

YEUNG, M. M.; MINTZER, F. An invisible watermarking technique for image verification. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Santa Barbara, USA, 1997. v. 1, p. 680–683.

ZHANG, F.; KIM, K. ID-based blind signature and ring signature from pairings. In: ADVANCES IN CRYPTOLOGY – ASIACRYPT'2002. *Proceedings*. Queenstown, New Zealand: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2501), p. 533–547.

____. Efficient ID-based blind signature and proxy signature from bilinear pairings. In: AUSTRALASIAN CONFERENCE ON INFORMATION SECURITY AND PRIVACY – ACISP'03. *Proceedings*. Wollongong, Australia: Springer-Verlag, 2003. (Lecture Notes in Computer Science, v. 2727), p. 312–323.

APÊNDICE A – PUBLICAÇÕES DO AUTOR

1. BARRETO, P. S. L. M.; KIM, H. Y. Pitfalls in public key watermarking. In: BRAZILIAN SYMPOSIUM ON COMPUTER GRAPHICS AND IMAGE PROCESSING – SIBGRAPI. *Proceedings*. Campinas, Brasil, 1999. p. 241–242.

2. KIM, H. Y.; BARRETO, P. S. L. M. Fast binary image resolution increasing by k-nearest neighbor learning. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Vancouver, Canadá, 2000. v. 2, p. 327–330.

3. BARRETO, P. S. L. M.; KIM, H. Y.; RIJMEN, V. Um modo de operação de funções de hashing para localizar alterações em dados digitalmente assinados. In: SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES (SBrT). *Anais*. Gramado, Brasil, 2000. #5150124. 1 CD-ROM.

4. BARRETO, P. S. L. M.; RIJMEN, V. The ANUBIS block cipher. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Bélgica: NESSIE Consortium, 2000.

5. ____. The KHAZAD legacy-level block cipher. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Bélgica: NESSIE Consortium, 2000.

6. ____. The WHIRLPOOL hashing function. In: FIRST OPEN NESSIE WORKSHOP. *Proceedings*. Leuven, Bélgica: NESSIE Consortium, 2000.

7. BARRETO, P. S. L. M.; RIJMEN, V.; NAKAHARA JR., J.; PRENEEL, B.; VANDEWALLE, J.; KIM, H. Y. Improved SQUARE attacks against reduced-round Hierocrypt. In: FAST SOFTWARE ENCRYPTION – FSE'2001. *Proceedings*. Yokohama, Japão: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2355), p. 165–173.

8. BARRETO, P. S. L. M.; KIM, H. Y.; RIJMEN, V. Toward a secure public-key blockwise fragile authentication watermarking. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING. *Proceedings*. Thessaloniki, Grécia, 2001. v. 2, p. 494–497.

9. BARRETO, P. S. L. M.; KIM, H. Y. *Fast hashing onto elliptic curves over fields of characteristic 3*. 2001. Cryptology ePrint Archive, Report 2001/098.

10. NAKAHARA JR., J.; BARRETO, P. S. L. M.; PRENEEL, B.; VANDEWALLE, J.; KIM, H. Y. Square attacks on reduced-round PES and IDEA block ciphers. In: 23RD SYMPOSIUM ON INFORMATION THEORY IN THE BENELUX. *Proceedings*. Louvain-la-Neuve, Bélgica, 2002.

11. BARRETO, P. S. L. M.; KIM, H. Y.; RIJMEN, V. Toward a secure public-key blockwise fragile authentication watermarking. *IEE Proceedings on Vision, Image and Signal Processing*, v. 149, n. 2, p. 57–62, 2002. Observação: este artigo é um desenvolvimento do item 8 acima.

12. BARRETO, P. S. L. M.; KIM, H. Y.; LYNN, B.; SCOTT, M. Efficient algorithms for pairing-based cryptosystems. In: ADVANCES IN CRYPTOLOGY – CRYPTO'2002. *Proceedings*. Santa Barbara, EUA: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2442), p. 377–387.

13. BARRETO, P. S. L. M.; LYNN, B.; SCOTT, M. Constructing elliptic curves with prescribed embedding degrees. In: SECURITY IN COMMUNICATION

NETWORKS – SCN'2002. *Proceedings*. Amalfi, Itália: Springer-Verlag, 2002. (Lecture Notes in Computer Science, v. 2576), p. 263–273.

14. DAEMEN, J.; RIJMEN, V.; BARRETO, P. S. L. M. Rijndael – Beyond the AES. In: MIKULÁŠSKÁ KRYPTOBESÍDKA 2002 – 3RD CZECH AND SLOVAK CRYPTOGRAPHY WORKSHOP. *Proceedings*. Praga, República Tcheca, 2002.

15. BARRETO, P. S. L. M.; LYNN, B.; SCOTT, M. On the selection of pairing-friendly groups. In: SELECTED AREAS IN CRYPTOGRAPHY – SAC'2003. *Proceedings*. Ottawa, Canadá: Springer-Verlag, 2003. (Lecture Notes in Computer Science). No prelo.

16. ____. Efficient implementation of pairing-based cryptosystems. *Journal of Cryptology*, Springer-Verlag, New York, EUA, 2003. No prelo.

17. BARRETO, P. S. L. M. A new algorithm for efficient construction of $LALR(1)$ and $LR(1)$ parsers. 2003. Preprint.

18. BARRETO, P. S. L. M.; VOLOCH, J. F. Efficient Computation of Roots in Finite Fields. 2003. Preprint.