UNIVERSIDADE DE ÉVORA
Departamento de Matemática

Mestrado em Matemática e Aplicações

Criptosistemas baseados em curvas elípticas:
âmbito e limitações

Dissertação de mestrado realizada sob orientação dos Professores:

Doutor Fernando Paulo Estrela de Pinho e Almeida
(Prof. Associado no Dep. de Mat. do Ist. Sup. Téc. de Lisboa)
Doutor Augusto José Franco de Oliveira
(Prof. Emérito da Universidade de Évora)

João Carlos Lopes Horta

Março 2009

UNIVERSIDADE DE ÉVORA
Departamento de Matemática

Mestrado em Matemática e Aplicações

Criptosistemas baseados em curvas elípticas:
âmbito e limitações

Dissertação de mestrado realizada sob orientação dos Professores:

Doutor Fernando Paulo Estrela de Pinho e Almeida
(Prof. Associado no Dep. de Mat. do Ist. Sup. Téc. de Lisboa)
Doutor Augusto José Franco de Oliveira
(Prof. Emérito da Universidade de Évora)

Dissertação apresentada ao Departamento de Matemática da Universidade de Évora como um dos requisitos para a obtenção do título de Mestre em Matemática e Aplicações.

João Carlos Lopes Horta

Março 2009

Resumo

As curvas elípticas têm um papel de relevo na criptografia actual, estando na origem de um dos métodos para estudo da factorização e da primalidade. O problema do logaritmo discreto no grupo de uma curva elíptica é utilizado como fonte para uma função de uma via num dos mais eficientes sistemas criptográficos. Não se obteve ainda um algoritmo com um tempo de execução subexponencial que permitisse resolver esse problema relativamente ao grupo de uma curva elíptica e que pudesse pôr em causa a segurança de um sistema criptográfico fundamentado nas curvas elípticas.

Os critérios de primalidade baseados em curvas elípticas são ainda entre os melhores métodos utilizados para passar certificado de primalidade a um número com mais de mil dígitos decimais.

A eficiência dos métodos para factorizar um número inteiro N, baseados em curvas elípticas, é tanto maior quanto maior for a diferença entre $\sqrt{N}$ e um dos divisores primos de N, o que impõe critérios preferenciais no uso do *software* mais adequado para as aplicações informáticas.

Abstract

Cryptosystems based on elliptic curves: scope and limits

Elliptic curves are mostly relevant in today's cryptography, allowing methods for factorization and primality. The discrete logarithm problem in the context of elliptic curves is used as a source for a one-way function in one of the most efficient cryptographic systems. No algorithm with a sub-exponential execution time, with respect to that problem, has been obtained, putting in risk the security of a cryptographic system based on elliptic curves.

Primality tests based on elliptic curves are still among the best ones allowing primality certificates for numbers with more than a thousand digits.

The factorization methods based on elliptic curves, currently used to factorize an integer N, become more and more efficient as the difference between one of the prime factors of N and $\sqrt{N}$ grows; this circumstance forces preferential criteria for the use of *software.*

Agradecimentos

As minhas palavras de apreço a todos aqueles que de uma forma ou de outra me apoiaram ao longo da realização deste trabalho.

Meu especial agradecimento:

A Deus;

À minha família;

Ao orientador e co-orientador da presente dissertação,
Professor Doutor Paulo Almeida
Professor Doutor Augusto Franco de Oliveira.

Sublinho com elevação o papel que o **Instituto Português de Apoio ao Desenvolvimento - IPAD** tem tido na formação de quadros caboverdianos, de que eu sou um exemplo.

A todos, um bem haja.

Conteúdo

Introdução

O envio e a recepção de informações sigilosas são necessidades que acompanharam a humanidade há milhares de anos. Essas necessidades vieram a dar origem ao termo *criptografia*, que, no sentido lato, significa conjunto de técnicas para cifrar e decifrar mensagens pelos seus interlocutores, tornando difícil o conhecimento dos conteúdos das mensagens por pessoas estranhas a elas.

Com o aparecimento da internet, as informações circulam de uma forma rápida e precisa, mas podem ser interceptadas por terceiros que não sejam o destinatário. Sendo assim, houve necessidade de criação de sistemas que permitissem a circulação de informações de uma forma segura, isto é, longe dos olhos dos "indesejados"; afinal a criptografia está a ser utilizada a todos os níveis: militar, económico, diplomático, etc.

A segurança desses sistemas baseia-se na hipotética dificuldade em resolver determinados tipos de problemas matemáticos, como o problema da factorização e o problema do logaritmo discreto.

A ideia de utilizar conhecimentos matemáticos na criptografia remonta a muitos anos atrás. Júlio César (100-44 a.C.), cifrava as suas mensagens trocando as posições das letras numa certa ordem, o que actualmente se traduz utilizando a operação de adição módulo n (ver [11]).

A grande revolução no campo da criptografia deu-se em 1976 com a denominada "*criptografia de chave pública*", quando Whitfield Diffie e Martin E. Hellman publicaram o artigo "*New directions in cryptography*" (ver [8]).

O sistema criptográfico baseado em curvas elípticas — naturalmente munidas de uma estrutura de grupo abeliano — foi proposto em 1985 indepentemente por Neal Koblitz e Victor Miller (ver [15, p. 131]) como uma forma de implementação da criptografia de chave pública, sendo até então os sistemas criptográficos baseados no grupo multiplicativo de um grupo finito $\mathbb{F}_q^*$.

Acerca do uso das curvas elípticas em criptografia cabe perguntar:

Como se relacionam elas com o estudo da primalidade e da factorização?

Que vantagens trazem elas em relação aos sistemas baseados no grupo $\mathbb{F}_q^*$?

Tendo em conta estas questões, o presente trabalho foi desenvolvido da seguinte forma:

No **Capítulo 1** aborda-se os conceitos e as propriedades importantes nos estudos de criptografia e criptosistemas.

No **Capítulo 2** apresenta-se algumas propriedades relevantes das curvas elípticas no campo da criptografia, onde se destaca o estudo do problema do logaritmo discreto no grupo de uma curva elíptica.

No **Capítulo 3** apresenta-se algoritmos de factorização e de primalidade aplicando as curvas elípticas. Fez-se uso de dois programas informáticos:

GP/PARI CALCULATOR Version 2.3.2

e

SAGE Version3.1.4, Release Date: 2008-10-20

para, principalmente, testar a capacidade dos algoritmos neles implementados para o estudo da factorização e da primalidade.

No **Capítulo 4** apresenta-se a situação das curvas elípticas na criptografia realçando as sua vantagens e desvantagens em relação aos outros métodos utilizados.

Os resultados obtidos foram fruto de pesquisas e análises bibliográficas, mas também de testes feitos aos programas PARI E SAGE no que concerne aos estudos de factorização e de primalidade.

Capítulo 1

Criptografia e criptosistemas

Criptografia (do grego kryptós, "escondido", graphé, "escrita") é o estudo dos princípios e técnicas pelas quais a informação pode ser transformada da sua forma original para outra ilegível, de forma que possa ser conhecida apenas pelo seu destinatário, o que a torna difícil de ser lida por alguém não autorizado.

A criptografia tem quatro objetivos principais:

1. *Confidencialidade da mensagem:* só o destinatário autorizado deve ser capaz de extrair o conteúdo da mensagem da sua forma cifrada.
2. *Integridade da mensagem*: o destinatário deverá ser capaz de determinar se a mensagem foi alterada durante a transmissão.
3. *Autenticação do remetente*: o destinatário deverá ser capaz de identificar o remetente e verificar que foi mesmo ele quem enviou a mensagem.
4. *Não-repúdio ou irretratabilidade do destinatário*: não deverá ser possível ao destinatário negar o envio da mensagem.

Apresenta-se, de seguida, alguns conceitos utilizados no estudo da criptografia.

- *Texto claro* — ou *mensagem* — é uma informação inteligível por qualquer um.
- *Criptograma* — ou *texto cifrado* — é uma informação ininteligível para qualquer um, excepto para o seu destinatário "legítimo".
- *Cifração* — é o processo de tranformação de um texto claro em um criptograma.
- *Decifração* — é o processo de recuperação de um texto claro a partir de um criptograma.

Chama-se *criptosistema* a um sêxtuplo (A, M, C, K, E, D), onde:

- A representa um conjunto finito de *alfabetos*, que segundo certas regras sintáticas e semânticas, permite escrever um texto claro bem como o seu respectivo criptograma.
- M — denominado *espaço de mensagens* — representa um conjunto de mensagens ou textos claros.
- C — denominado *espaço de criptograma* — representa um conjunto de criptogramas ou textos cifrados.

- K — denominado *espaço de chaves* — representa um conjunto finito formado por dois tipos de elementos: chaves de cifração, representadas por e, e chaves de decifração, representadas por d. Cada elemento $e \in K$ determina uma única função bijectiva E_e de M em C, denominada *função de cifração*. Para cada chave

$$e \in K,$$

existe uma única chave

$$d \in K,$$

que determina uma única função E_d de C em M, denominada *função de decifração*, tal que, se

$$E_e(m) = c \qquad \text{então} \qquad E_d(c) = m,$$

onde $m \in M$ e $c \in C$.

- E representa um conjunto de funções de cifração.
- D representa um conjunto de funções de decifração.

Nota 1.1 *Geralmente apresenta-se um criptosistema especificando apenas os conjuntos*

$$K, E \text{ e } D.$$

Fundamentalmente, existem dois tipos de criptosistemas, atendendo ao uso das chaves:

1. *Criptosistemas simétricos* — ou *de chave secreta*: a chave de cifração é relacionada de uma forma directa à chave de decifração — que podem ser idênticas ou admitem uma simples transformação entre as duas chaves. Às vezes usa-se uma única chave — usada por ambos interlocutores — na premissa de que esta é conhecida apenas por eles — por isso a denominação *criptosistema simétrico*. As chaves, na prática, representam um segredo compartilhado entre dois ou mais interlocutores, que pode ser usado para manter uma ligação confidencial de informação. Neste tipo de criptosistema é necessário *um sistema seguro para a combinação das chaves*. Uma vez que cada par de interlocutores necessita de uma chave secreta, uma rede de comunicação com n interlocutores necessita de $\frac{n(n-1)}{2}$ — isso constitui uma grande desvantagem de um criptosistema simétrico.

2. *Criptosistemas assimétricos* — ou de *chave pública* (ver [9, p. 42]), sistema proposto inicialmente por Diffie e Hellman em 1976): cada entidade é possuidora de um par de chaves, uma *pública* — para cifrar mensagens — e uma *privada* — para decifrar criptogramas e para autenticar uma mensagem. A chave pública deve ser distribuída, livremente, para todos os correspondentes enquanto que a chave privada deve ser conhecida apenas pelo seu dono. Uma mensagem cifrada pela chave pública deve ser decifrada apenas pela chave privada correspondente. *Os sistemas assimétricos devem obedecer à condição de que não se pode determinar a chave privada a partir da chave pública.*

As chaves num criptosistema assimétrico estão interligadas através de uma função f que se denomina *função de uma via.*

Definição 1.1 *Dados dois conjuntos A e B e a função*

$$f : A \to B,$$

diz-se que f é uma função de uma via *se dado um $a \in A$ for* computacionalmente fácil *determinar $f(a)$ e dado um $b \in \mathrm{Im}\,(f)$ for* computacionalmente difícil *obter um $a \in A$ tal que $f(a) = b$.*

Nota 1.2 *Diz-se que a resolução de um problema é* computacionalmente fácil *se utilizando os recursos computacionais existentes essa puder ser executada num tempo desejado para quem a executa; caso contrário, diz-se que a resolução desse problema é* computacionalmente difícil.

Dada uma função de uma via f, escolhe-se $a \in A$ para a chave privada, e determina-se $f(a)$ para a chave pública. Tendo em conta as características de uma função de uma via, será computacionalmente difícil determinar a a partir de $f(a)$.

Para algumas aplicações utiliza-se uma função de uma via f invertível cuja inversa f^{-1} possa ser obtida se se estiver na posse de determinadas informações. Tal função é uma *"falsa função de uma via"*.

A escolha de função de uma via f baseia-se, muitas vezes, em determinados tipos de problemas matemáticos que se consideram de difícil resolução, como os apresentados abaixo:

1. O problema de factorização de um número natural n.
2. O problema do logaritmo discreto em determinados tipos de grupo.
3. "SVP - shorter vector problem".

Conceitos como *algoritmo* e *complexidade computacional de um algoritmo* estão bem associados a um criptosistema.

Definição 1.2 *Chama-se* algoritmo *a todo o processo bem definido, constituído por um conjunto de instruções que a partir de um conjunto de valores de entrada produz um conjunto de valores de saída. Diz-se que um algoritmo é* determinístico *se o conjunto de valores de entrada determinar completamente o conjunto de valores de saída. Se o mesmo conjunto de valores de entrada produzir conjuntos de valores de saída diferentes, o algoritmo dir-se-á* probabilístico ou aleatório.

Nota 1.3 *Dado o uso de computadores, em geral, os valores de entrada e de saída de um algoritmo estão em código binário — cujos dígitos são* 0 *e* 1.

Definição 1.3 *Dado um valor n, chama-se* comprimento de n *ao número de dígitos, dado por $\log_2 n$, que tem a composição de n em código binário, utilizando um esquema de codificação apropriado.*

Definição 1.4 *O* tempo de execução de um algoritmo *é o número máximo de operações elementares (operações bit a bit) que se efectua ao se executar o algoritmo a partir de um certo conjunto de valores de entrada. A* memória consumida *é o número de símbolos escritos na memória para levar a cabo um algoritmo.*

A *eficiência de um algoritmo* medir-se-á pelo seu *tempo de execução* e pela *memória consumida*, quando ele for implementado num computador.

Dado um um algoritmo associa-se a ele uma função f, que limita superiormente o recurso computacional necessário para a sua execução, que se denomina *parâmetro de complexidade*. Se o recurso considerado for o tempo de execução ou a memória consumida, f mede, respectivamente, a *complexidade de tempo* e a *complexidade de espaço* (ver *[6, p. 3]*).

Nota 1.4 *Muitas vezes é difícil obter a partir de um algoritmo a sua complexidade de tempo e a sua complexidade de espaço (ver [18, p. 58]). Nessa situação, há que estabelecer uma aproximação. Para isso, utiliza-se, respectivamente, a complexidade de tempo e a complexidade de espaço* assimptótico, *isto é, estuda-se o aumento do tempo de execução e do consumo da memória, quando o comprimento n do valor de entrada aumenta ilimitadamente.*

Definição 1.5 *Sejam $f, g : \mathbb{N} \to \mathbb{R}$, duas funções estritamente positivas.*
Escreve-se:

- $f(n) = O(g(n))$ *se existir um número real estritamente positivo c e um número natural n_0 tal que*
$$\forall n \geq n_0 \qquad 0 \leq f(n) \leq c\, g(n).$$
- $f(n) = o(g(n))$ *se*
$$\lim \frac{f(n)}{g(n)} = 0.$$
A expressão $o(1)$ é portanto usada para representar uma função $f(n)$ cujo limite é 0 quando n tende para ∞.

Definição 1.6 *Define-se para n natural a função*

$$L_n(\alpha, c) = \exp\left((c + o(1))(\log n)^{\alpha}(\log\log n)^{1-\alpha}\right),$$

onde $0 \leq \alpha \leq 1$, $c > 0$. Se se omitir o segundo parâmetro c, considerar-se-á $c = \frac{1}{2}$. Sendo n o comprimento do valor da entrada de um algoritmo, a sua complexidade — relativamente a um determinado recurso computacional — pode ser medida em função do valor de α (ver [6, p. 4]):

- Se $\alpha = 0$, uma função de complexidade $O(L_n(0, c))$ será *polinomial* em $\log n$. Neste caso considera-se que o algoritmo é muito eficiente, tendo em conta o propósito para o qual o algoritmo foi concebido.
- Se $\alpha = 1$, uma função de complexidade $O(L_n(1, c))$ será *exponencial* em $\log n$. Neste caso, considera-se que o algoritmo não é eficiente, tendo em conta o propósito para o qual foi concebido.
- Se $0 < \alpha < 1$, dir-se-á que uma função de complexidade $O(L_n(\alpha, c))$ será *sub-exponencial*. Neste caso, considera-se que o algoritmo é também eficiente, embora o desejável — para quem o concebe — seja ter uma complexidade polinomial.

Diz-se que um *criptosistema é seguro* se ele garantir todos os objectivos para os quais foi concebido.

Nem todos os criptosistemas são utilizados para atingir todos os objetivos principais da criptografia. Mesmo em criptosistemas bem concebidos, bem implementados e usados adequadamente, alguns dos objectivos não são práticos — ou mesmo desejáveis — em algumas circunstâncias. Por exemplo, o remetente de uma mensagem pode querer permanecer anónimo ou pode não interessar a confidencialidade.

Se um criptosistema garantir a confidencialidade da informação circulada, os criptogramas não poderão revelar as informações da respectiva mensagem. Esta ideia remete-nos para o conceito de *segurança semântica* (ver [22, p. 22]).

Uma vez que a chave pública é do conhecimento de todos, o receptor do criptograma não tem informação acerca do seu emissor nem da sua integridade. A garantia de autenticidade, integridade e a irretratabilidade num criptosistema são dadas através de uma *assinatura* — uma mensagem — que acompanha o respectivo criptograma.

Um *ataque* a um criptosistema, consiste num algoritmo que resolva um problema do qual decorre o algoritmo de encriptação.

A *eficiência* de um ataque é medida pela sua *complexidade de tempo e de espaço* que se pretende neste caso, que sejam os menores possíveis.

1.1 Criptografia baseada em grupos finitos

Os dois principais sistemas utilizados num criptosistema de chave pública são o sistema *RSA* e os *protocolos baseados no problema do logaritmo discreto* num grupo cíclico (ver [9]).

Para este trabalho, vai-se realçar o estudo de criptosistemas baseados no *problema do logaritmo discreto* (PLD) pois um *criptosistema baseado em curvas elípticas* — tema deste trabalho — baseia-se nesse problema.

Tendo em conta que um PLD num grupo finito G pode ser reduzido a um PLD num subgrupo de G cuja ordem é um número primo, segundo a *simplificação de Pohlig e Hellman* (que se verá mais adiante), define-se o PLD num grupo finito de ordem prima.

Definição 1.7 *Seja $(G, \oplus)$ um grupo de ordem l, onde l é um número primo, e seja $P, Q \in G$. O* logaritmo discreto *em G de Q na base P é um número natural*

$$n = \mathrm{dlog}_P(Q)$$

tal que

$$Q = nP = \underbrace{P \oplus P \oplus \ldots \oplus P}_{n \text{ vezes}}.$$

O número n é determinado modulo l. A determinação de n designa-se por problema de logaritmo discreto *(PLD) em G.*

A *complexidade da resolução de* PLD depende muito da escolha do grupo. Por exemplo, para o grupo aditivo do corpo finito $\mathbb{F}_q$, onde $q = p^m$ para um número primo p e um número natural m, o PLD é fácil de ser resolvido (ver [6, p. 8]). Para fins criptográficos, a escolha do grupo G deve ser cuidadosa e deve obedecer a algumas condições, como as que se seguem:

1. A ordem de G deve ser um número primo l *muito grande*, de forma a tornar a resolução de certos problemas computacionalmente difíceis.
2. A complexidade de espaço dos elementos de G deve ser da ordem $O(\log l)$;
3. A operação $\oplus$ de G deve ser eficientemente determinada — com complexidade de tempo polinomial.
4. O PLD deve ser de difícil resolução, isto é, a complexidade de tempo deve ser exponenecial.

Uma das opções para grupo o G muito utilizada é o grupo multiplicativo $\mathbb{F}_q^*$ de um corpo finito $\mathbb{F}_q$, para um valor de q muito grande.

Uma outra opção para grupo o G é o *grupo de pontos racionais* $E_{/\mathbb{F}_q}$ *de uma curva elíptica* (que se definirá mais adiante) definida sobre um corpo finito.

A soma de pontos num grupo $E_{/\mathbb{F}_q}$ é fácil de determinar, enquanto que o PLD aplicado a esse grupo é muito difícil de resolver, tendo em conta os recursos computacionais existentes. Ainda mais, o PLD aplicado ao grupo $E_{/\mathbb{F}_q}$ tem grau de dificuldade superior ao PLD num grupo multiplicativo $\mathbb{F}_q^*$, onde $\mathbb{F}_q$ e $E_{/\mathbb{F}_q}$ têm a mesma ordem (ver [11, p. 396]).

Os *grupos de classes de ordens de um corpo numérico* serão uma boa opção, pois são considerados seguros e práticos, embora apresentem algumas limitações — como pode-se ver em [9].

1.2 Primalidade e factorização

A factorização e a primalidade conduzem a dois problemas matemáticos de extrema importância, incidindo no estudo da implementação de criptosistemas assimétricos. Por exemplo, precisa-se da factorização em números primos para constituir criptosistemas cujos esquemas se baseiam em grupos cujas ordens são números primos muito grandes — *como é o caso de criptosistemas baseados em curvas elípticas.* A factorização é um dos problemas em que se baseiam as seguranças de alguns criptosistemas, como é o caso do sistema *RSA* (ver [11, p. 113]).

Primalidade

Sabe-se que no século 300 a.C. Euclides provou a existência de infinitos números primos. Sabe-se ainda que sendo $\pi(N)$ o número de números primos não excedendo N; o teorema dos números primos afirma que

$$\pi(N) \sim \frac{N}{\log N} \qquad (N \to +\infty).$$

Uma das formas de saber se um número natural N é primo ou composto, é verificar se para todo o número inteiro $n \leq \sqrt{N}$ se tem $N \equiv 0 \pmod{n}$.(Nota-se que se n for um divisor de N, então $\frac{N}{n} \leq \sqrt{N}$) contudo, este método tem uma complexidade de ordem $O\left(N\sqrt{N}\right)$.

Tem-se desenvolvido estudos muito intensos no sentido de descobrir números primos cada vez maiores. Existem *algoritmos probabilísticos do estudo de primalidade* — para um valor de entrada N, em que a resposta pode ser: N é composto ou N é *provavelmente*

primo. Se for a primeira opção é porque N será realmente um número composto, mas se for a segunda opção, terá que se passar um *certificado de primalidade* ao número N — um teste de primalidade realizado por um algoritmo que garanta a 100% que N é um número primo — pois não há argumentos matemáticos suficientes que garantam que N é realmente um número primo quando se utiliza um algoritmo probabilístico para o estudo da primalidade.

O *método das curvas elípticas é um dos métodos mais utilizados* para passar certificado de primalidade a um número natural n supostamente primo (ver [6, p. 597]).

Factorização

É possível encontrar um factor de n em $\sqrt{n}$ passos — no máximo — mas a técnica utilizada para tal é muito lenta para valores de n muito grandes. Por exemplo, se n for composto por 100 dígitos, supondo que em cada segundo verificam-se 1 000 000 de possíveis divisores, então encontrar-se-á um divisor de n em cerca de $3,2 \times 10^{37}$ anos (ver [21, pág. 126]).

O *teorema fundamental da aritmética* garante que todo o número natural $n > 1$ pode ser decomposto de uma forma única — a menos da permutação — num produto,

$$n = p_1^{a_1} p_2^{a_2} ... p_k^{a_k},$$

onde a_i, $i = 1...k$, são números inteiros positivos e $p_1 < p_2 < ... < p_k$ são números primos. Um grande problema da aritmética, fundamentado neste teorema, é a f*actorização de um número natural* $n > 1$.

O estudo da factorização, pela sua natureza, é mais difícil do que o estudo da primalidade (ver [18, p. 89]). Segundo [23, p. 189], o maior número factorizado até ao ano 2007 foi um inteiro de 200 dígitos, enquanto que nesse mesmo ano provava-se a primalidade de um inteiro com muitos milhares de dígitos.

Muitos métodos foram desenvolvidos para resolver o problema de factorização. Alguns são concebidos para factorizar um número natural n mediante determinadas condições, são denominados *métodos específicos de factorização.* São alguns deles (ver [11]):

- O método de *Pollard* - ρ ;
- O método de Pollard - $p - 1$;
- *O método das curvas elípticas* (ver [18, p. 90 - 94]).
- "General number field sieve".
- "Shortest vector".

O método das curvas elípticas é inspirado no método de Pollard - $p - 1$, mas oferece algumas vantagens em relação a este (ver [21, p. 125-138]).

A complexidade de tempo do método das curvas elípticas para obter o menor factor primo de um número natural n é $L\left(\frac{1}{2}, \sqrt{2}\right)$ (ver [6, p. 7, 604-606]).

Capítulo 2

Aritmética de uma curva elíptica

Seja $\mathbb{K}$ um corpo comutativo e $\overline{\mathbb{K}}$ o seu fecho algébrico.

Definição 2.1 *Uma* curva elíptica E *sobre o corpo* $\mathbb{K}$, *representada por* $E(\mathbb{K})$, *é uma curva não singular definida pelo conjunto de soluções no plano projectivo* $P^2(\overline{\mathbb{K}})$ *da* equação homogénea de Weierstrass

$$Y^2Z + uXYZ + vYZ^2 = X^3 + aX^2Z + bXZ^2 + cZ^3,$$

onde $u, v, a, b, c \in \mathbb{K}$, *ou seja,*

$$E(\mathbb{K}) = \left\{[X, Y, Z] \in P^2(\overline{\mathbb{K}}) : Y^2Z + uXYZ + vYZ^2 = X^3 + aX^2Z + bXZ^2 + cZ^3\right\}.$$

A não singularidade da curva elíptica significa que as derivadas parciais,

$$\frac{\partial F}{\partial X}, \frac{\partial F}{\partial Y} \text{ e } \frac{\partial F}{\partial Z},$$

não se anulam simultaneamente para nenhum ponto da curva, onde

$$F(X, Y, Z) = Y^2Z + uXYZ + vYZ^2 - X^3 - aX^2Z - bXZ^2 - cZ^3.$$

Definição 2.2 *Seja* $E(\mathbb{K})$ *uma curva elíptica e* $P \in E(\mathbb{K})$. *O ponto* P *diz-se* $\mathbb{K}$-ponto racional *se as suas coordenadas pertencerem ao corpo* $\mathbb{K}$.

Nota 2.1 *Por vezes diz-se, simplesmente,* ponto racional *quando não se tem dúvida acerca do corpo sobre o qual a curva é definida.*

O conjunto dos $\mathbb{K}$-pontos racionais *representa-se por* $E_{/\mathbb{K}}$.

Uma curva elíptica tem um único ponto racional para $Z = 0$, o ponto $[0, 1, 0]$, denomina-se *ponto no infinito* e representa-se por $\mathcal{O}$.

Quando $Z \neq 0$, um ponto

$$[X, Y, Z] = [a, b, c]$$

da curva $E(\mathbb{K})$ corresponde no plano afim ao ponto

$$(x, y) = \left(\frac{a}{c}, \frac{b}{c}\right) \in \overline{\mathbb{K}}^2$$

dado pela *equação não homogénea de Weierstrass*

$$y^2 + uxy + vy = x^3 + ax^2 + bx + c, \tag{2.1}$$

onde $u, v, a, b, c \in \mathbb{K}$.

O conjunto de pontos da curva elíptica $E(\mathbb{K})$ dados no plano afim por

$$\left\{(x, y) \in \overline{\mathbb{K}}^2 : y^2 + uxy + vy = x^3 + ax^2 + bx + c\right\}$$

diz-se a *parte afim* dessa curva elíptica.

Assim sendo, uma curva elíptica $E(\mathbb{K})$ pode ser dada, também, por

$$\left\{(x, y) \in \overline{\mathbb{K}}^2 : y^2 + uxy + vy = x^3 + ax^2 + bx + c\right\} \cup \{\mathcal{O}\}$$

e o conjunto $E_{/\mathbb{K}}$ dos $\mathbb{K}$-*pontos racionais* por

$$\left\{(x, y) \in \mathbb{K}^2 : y^2 + uxy + vy = x^3 + ax^2 + bx + c\right\} \cup \{\mathcal{O}\},$$

onde $u, v, a, b, c \in \mathbb{K}$.

A partir da equação de uma curva elíptica, algumas constantes são definidas e utilizadas nas fórmulas usadas adiante. São elas (ver [4, p. 30]):

$$\left.\begin{array}{l} b_2 = u^2 + 4a, \\ b_4 = uv + 2b, \\ b_6 = v^2 + 4c, \\ b_8 = u^2c + 4ac - uvb + av^2 - b^2, \\ c_4 = b_2^2 - 24b_4 \text{ e} \\ c_6 = -b_2^3 + 36b_2b_4 - 216b_6. \end{array}\right\} \tag{2.2}$$

Definição 2.3 *Sejam $E_{/\mathbb{K}}$ e b_2, b_4, b_6 e b_8 definidos como acima. O* discriminante *de $E(\mathbb{K})$ representado por Δ, define-se por*

$$\Delta = -b_2^2b_8 - 8b_4^3 - 27b_6^2 + 9b_2b_4b_6.$$

Uma curva elíptica é não singular se e só se $\Delta \neq 0$ (ver [4, p. 31]). Sendo assim, uma curva elíptica $E(\mathbb{K})$ tem discriminante Δ sempre diferente de zero.

Sempre que $\Delta \neq 0$, define-se o *j-invariante* $j(E)$ da curva elíptica E como sendo,

$$j(E) = \frac{c_4^3}{\Delta}.$$

Definição 2.4 *Duas curvas elípticas E_1 e E_2 sobre um corpo $\mathbb{K}$, definidas pelas equações não-homogéneas de Weierstrass*

$$y^2 + u_1xy + v_1y = x^3 + a_1x^2 + b_1x + c_1 \tag{2.3}$$

e

$$y^2 + u_2xy + v_2y = x^3 + a_2x^2 + b_2x + c_2 \tag{2.4}$$

são isomorfas sobre sobre $\mathbb{K}$, se existirem constantes $r, s, t \in \mathbb{K}$' e $d \in \mathbb{K}^*$ (grupo multiplicativo do corpo $\mathbb{K}$), tais que a mudança de variável

$$(x, y) \mapsto (d^2x + r, d^3y + d^2sx + t) \tag{2.5}$$

transforma a equação 2.3 na equação 2.4.

Nota 2.2 *Nota-se que o isomorfismo é definido sobre o corpo* $\mathbb{K}$. *Curvas que não são isomorfas sobre* $\mathbb{K}$, *podem ser isomorfas sobre uma extensão* $\mathbb{K}'$ *de* $\mathbb{K}$ *(ver [4, p. 31]).*

Lema 2.1 *Duas curvas elípticas* E_1 *e* E_2 *sobre um corpo* $\mathbb{K}$ *são isomorfas se tiverem o mesmo* j-invariante. *Duas curvas* E_1 *e* E_2 *com a mesma* j-invariante *são isomorfas sobre* $\overline{K}$ *(ver [4, p. 31]).*

A equação não-homogénea de Weierstrass 2.1 pode ser simplicada, conforme for a característica de $\mathbb{K}$, aplicando a mudança de variável 2.5. Assim sendo, tem-se:

1. Se a característica de $\mathbb{K}$ for diferente de 2 e de 3, a equação 2.1 poderá ser transformada numa equação do tipo

$$y^2 = x^3 + b'x + c',$$

onde b', $c' \in \mathbb{K}$, e por conseguinte $\Delta = -16\,(4b'^3 + 27c'^2)$.

2. Se a característica de $\mathbb{K}$ for igual a 2 e $u \neq 0$, a equação 2.1 poderá ser transformada numa equação do tipo

$$y^2 + xy = x^3 + a'x^2 + c',$$

onde a', $c' \in \mathbb{K}$, e por conseguinte $\Delta = c'$. Se $u = 0$, a equação 2.1 poderá ser transformada numa equação do tipo

$$y^2 + v'y = x^3 + b'x + c',$$

onde v', b', $c' \in \mathbb{K}$, e por conseguinte $\Delta = v'^4$.

3. Se a característica de $\mathbb{K}$ for igual a 3 e $u^2 \neq -a$, a equação 2.1 poderá ser transformada numa equação do tipo

$$y^2 = x^3 + a'x^2 + c',$$

onde a', $c' \in \mathbb{K}$, e por conseguinte $\Delta = -a'^3c'$. Se $u^2 = -a$, a equação 2.1 poderá ser transformada numa equação do tipo

$$y^2 = x^3 + b'x + c',$$

onde b', $c' \in \mathbb{K}$, e por conseguinte $\Delta = -b'^3$.

Observação 2.1 *Sempre que a característica de* $\mathbb{K}$ *for diferente de dois o conjunto dos* $\mathbb{K}$-pontos racionais *é representado por* $E(a,b,c)_{/\mathbb{K}}$, *onde* a, b, c *são os coeficientes dos termos de grau* 2, 1 *e* 0, *respectivamente, do polinómio* $f(x) = x^3 + ax^2 + bx + c$, *que figura no* 2^o *membro da equação da parte afim da curva elíptica* $E(\mathbb{K})$.

2.1 Lei de grupo numa curva elíptica

É possível definir uma "adição" de pontos no conjunto $E_{/\mathbb{K}}$ dos $\mathbb{K}$-pontos racionais de uma curva elíptica E sobre o corpo $\mathbb{K}$. A adição dos pontos baseia-se no facto de que no plano $P^2\left(\overline{\mathbb{K}}\right)$, uma recta intersectando uma curva elíptica em pelo menos dois $\mathbb{K}$-pontos racionais, intersecta a curva num terceiro ponto que é também um $\mathbb{K}$-ponto racional.

Então para adicionar dois $\mathbb{K}$-pontos racionais de uma curva elíptica, digamos P e Q, primeiro une-se esses dois pontos por uma linha recta obtendo o terceiro ponto $P * Q$ de intersecção com a curva. Depois une-se o ponto $P * Q$ com o ponto $\mathcal{O}$ — ponto no infinito do plano $P^2(\mathbb{K})$ — por uma linha recta obtendo-se o terceiro ponto de intersecção que é $P + Q$, adição de P com Q.

Quando se quer calcular o dobro de um ponto (diga-se P), a mesma técnica poder ser aplicada, substituindo apenas a recta que une os dois pontos pela tangente ao único ponto P. Nestas condições o ponto $\mathcal{O}$ é o elemento neutro para a operação assim definida e além disso todo o ponto P de $E_{/\mathbb{K}}$ tem um simétrico $-P$ relativamente a esta operação. Como a operação é associativa — e comutativa — pode dizer-se que $E_{/\mathbb{K}}$ constitui um grupo comutativo para a adição acabada de definir.

Teorema 2.2 (Estrutura do grupo) *Seja $E(\mathbb{F}_q)$ uma curva elíptica. O grupo $E_{/\mathbb{F}_q}$ é* isomorfo ao grupo

$$\mathbb{Z}_{/n_1} \oplus \mathbb{Z}_{/n_2}$$

tal que $n_1 \mid n_2$ e $n_1 \mid q-1$, onde n_1 e n_2 são inteiros positivos unicamente determinados (ver [6, p. 272)]).

Estude-se de modo mais analítico a adição para pontos de $E_{/\mathbb{K}}$ situados na parte afim de $P_2(\mathbb{K})$.

Sejam $P = (x_1, y_1)$, $Q = (x_2, y_2)$ com $x_1 \neq x_2$ pontos de $E_{/\mathbb{K}}$ distintos de $\mathcal{O}$. Seja $R = (x_3, y_3)$ a soma de P e Q, isto é,

$$R = P + Q.$$

A recta que passa por P e Q é dada pela equação

$$y = \lambda x + \mu$$

onde

$$\lambda = \frac{y_1 - y_2}{x_1 - x_2}$$

e

$$\mu = y_1 - \lambda x_1 = y_2 - \lambda x_2.$$

Os pontos de intersecção dessa recta com a curva são obtidos pela equação

$$(\lambda x + \mu)^2 + (ux + v)(\lambda x + \mu) = x^3 + ax^2 + bx + c\ ,$$

o que é equivalente à equação $r(x) = 0$ onde

$$r(x) = x^3 + \left(a - \lambda^2 - u\lambda\right) x^2 + (b - 2\lambda\mu - v\lambda - u\mu)\, x + c - \mu^2 - v\mu\ .$$

Por outro lado já se conhece duas raízes de $r(x)$, x_1 e x_2, e portanto

$$r(x) = (x - x_1)(x - x_2)(x - x_3)\ .$$

Comparando-se os coeficientes dos termos do 2^o grau das duas expressões, obtém-se

$$\lambda^2 + u\lambda - a = x_1 + x_2 + x_3\ ;$$

como x_1 e x_2 pertencem ao corpo $\mathbb{K}$, o elemento x_3 também pertence bem como $\lambda x_3 + \mu$.

Note-se que se $P = (x_1, y_1)$ pertencer à curva também lhe pertencerá o ponto

$$(x_1, -y_1 - ux_1 - v),$$

o que corresponde a $-P$ uma vez que o ponto $\mathcal{O}$ é o elemento neutro do grupo.

A soma R tem abcissa x_3 e deve pertencer à curva elíptica e a sua ordenada terá então de ser dada por

$$y_3 = -\lambda x_3 - \mu - ux_3 - v.$$

No cáculo do dobro de $P = (x_1, y_1)$ o declive da recta tangente é a derivada implícita y' no ponto P obtida a partir de

$$y^2 + uxy + vy - x^3 - ax^2 - bx - c = 0.$$

Tem-se então com relação à adição em $E_{/\mathbb{K}}$:

1. $\mathcal{O}$ é o elemento neutro;
2. Simétrico de $P = (x_1, y_1)$ é

$$-P = (x_1, -y_1 - ux_1 - v);$$

3. $P + Q = (x_3, y_3)$ onde

$$x_3 = \lambda^2 + u\lambda - a - x_1 - x_2,$$
$$y_3 = \lambda(x_1 - x_3) - y_1 - ux_3 - v$$

e

$$\lambda = \begin{cases} \dfrac{y_1 - y_2}{x_1 - x_2} & \text{se } P \neq \pm Q \\ \dfrac{3x_1^2 + 2ax_1 + b - uy_1}{2y_1 + ux_1 + v} & \text{se } P = Q. \end{cases} \tag{2.6}$$

O estudo das curvas elípticas em criptografia assume grande importância quando elas são consideradas sobre um corpo finito $\mathbb{F}_q$, pois sobre $\mathbb{Q}$, $\mathbb{R}$ ou $\mathbb{C}$ os cálculos envolvem aproximações tornando o sistema criptográfico pouco prático e preciso (ver [2, p. 16]).

Lei de grupo de uma curva elíptica sobre o corpo $\mathbb{F}_p$

O corpo $\mathbb{F}_p$ é constituído por p elementos, isto é,

$$\mathbb{F}_p = \{0, 1, 2, 3, ..., p-1\},$$

onde 0, 1, 2, 3,... ,$p-1$ são classes residuais mod p.

A equação da curva elíptica $E(\mathbb{F}_p)$ pode reduzir-se à forma

$$y^2 = x^3 + ax^2 + bx + c \,(\text{mod } p)$$

quando $p \geq 3$, com $\Delta \neq 0$. Então:

1. O simétrico de $P = (x_1, y_1) \neq \mathcal{O} \in E_{/\mathbb{F}_q}$ é

$$-P = (x_1, -y_1);$$

2. Seja $P = (x_1, y_1) \in E_{/\mathbb{F}_q}$, $Q = (x_2, y_2) \in E_{/\mathbb{F}_q}$ e $x_1 \neq x_2$. Então

$$P + Q = (x_3, y_3) \in E_{/\mathbb{F}_q},$$

onde

$$\begin{cases} x_3 \equiv \lambda^2 - a - x_1 - x_2 \pmod{p} \\ y_3 \equiv \lambda.(x_1 - x_3) - y_1 \pmod{p} \\ \lambda \equiv \dfrac{y_2 - y_1}{x_2 - x_1} \pmod{p}. \end{cases}$$

3. Seja $P = (x_1, y_1) \neq \mathcal{O} \in E_{/\mathbb{F}_q}$.e $y_1 \neq 0$. Então

$$P + P = 2P = (x_3, y_3),$$

onde

$$\begin{cases} x_3 \equiv \lambda^2 - a - 2x_1 \pmod{p} \\ y_3 \equiv \lambda.(x_1 - x_3) - y_1 \pmod{p} \\ \lambda \equiv \dfrac{3x_1^2 + 2ax_1 + b}{2y_1} \pmod{p} \end{cases} \quad \text{e} \quad y_1 \neq 0.$$

4. Se $y_1 = 0$ então $2P = \mathcal{O}$.

Exemplo 2.1 *Considera-se o grupo* $E(0,1,3)_{/\mathbb{F}_{11}}$ *cujos elementos são os apresentados na tabela seguinte:*

$\mathcal{O}$	$(4,4)$	$(7,1)$
$(0,5)$	$(4,7)$	$(7,10)$
$(0,6)$	$(5,1)$	$(9,2)$
$(1,4)$	$(5,10)$	$(9,9)$
$(1,7)$	$(6,4)$	$(10,1)$
$(3,0)$	$(6,7)$	$(10,10)$.

Seja $P = (4,4)$, $Q = (0,6) \in E(0,1,3)_{/\mathbb{F}_{11}}$. *Tem-se:*

- *O simétrico de* P *é*

$$\begin{aligned} -P &= (4,-4) \\ &= (4,7) \end{aligned}$$

- *A soma de* P *e* Q *é*

$$\begin{aligned} P + Q &= (x_3, y_3) \\ &= (10,10), \end{aligned}$$

onde

$$\lambda = \frac{6-4}{0-4} (\mathrm{mod}\, 11) = 2.7^{-1} (\mathrm{mod}\, 11) = 2.8 (\mathrm{mod}\, 11) = 5$$
$$x_3 = 5^2 - 4 (\mathrm{mod}\, 11) = 21 (\mathrm{mod}\, 11) = 10$$
$$y_3 = 5.(4-10) - 4 (\mathrm{mod}\, 11) = 5.5 + 7 (\mathrm{mod}\, 11) = 10$$

- *O dobro de* P *é*

$$\begin{aligned} 2P &= (x_3, y_3) \\ &= (7,1), \end{aligned}$$

onde

$\lambda = \frac{3.4^2+1}{2.4} (\text{mod } 11) = 49.8^{-1} (\text{mod } 11) = 49.7 (\text{mod } 11) = 2$
$x_3 = 2^2 - 2.4 (\text{mod } 11) = 7$
$y_3 = 2.(4-7) - 4 (\text{mod } 11) = 2.8 + 7 (\text{mod } 11) = 1$

Lei de grupo de uma curva elíptica sobre o corpo F_{2^m}

O corpo $\mathbb{F}_{2^m} = \mathbb{F}_2[x] / \langle f(x) \rangle$, onde

$$f(x) = a_m x^m + a_{m-1} x^{m-1} + a_{m-2} x^{m-2} + ... + a_1 x + a_0,$$

$a_i \in \mathbb{F}_2$, $i = 0, ..., m$, é irredutível em $\mathbb{F}_2[x]$, isto é, $\mathbb{F}_{2^m}$ é composto pelos polinómios da forma

$$a_{m-1} x^{m-1} + a_{m-2} x^{m-2} + ... + a_1 x + a_0,$$

onde $a_i \in \mathbb{F}_2$,$i = 0, ..., m-1$ que são restos da divisão de $g(x) \in \mathbb{F}_2[x]$ por $f(x)$.

Por ser $\mathbb{F}_{2^m}$ um corpo de característica dois, a equação da curva elíptica $E(\mathbb{F}_{2^m})$ é dada pela equação

$$y^2 + xy = x^3 + ax^2 + c, \tag{2.7}$$

onde a, $c \in \mathbb{F}_{2^m}$ e $c \neq 0$, ou pela equação

$$y^2 + vy = x^3 + bx + c, \tag{2.8}$$

onde v, b e $c \in \mathbb{F}_{2^m}$ e $v \neq 0$.

Nota 2.3 *Note-se que a primeira equação tem $j(E)$ sempre diferente de zero enquanto que a segunda tem $j(E) = 0$.*

Para o grupo da curva elíptica $E(\mathbb{F}_{2^m})$ dada pela equação

$$y^2 + xy = x^3 + ax^2 + c,$$

onde a, $c \in \mathbb{F}_{2^m}$ e $c \neq 0$, tem-se:

1. O simétrico de $P = (x_1, y_1) \neq \mathcal{O} \in E_{/\mathbb{F}_{2^m}}$ é

$$-P = (x_1, x_1 + y_1);$$

2. Seja $P = (x_1, y_1) \in E_{/\mathbb{F}_{2^m}}$, $Q = (x_2, y_2) \in E_{/\mathbb{F}_{2^m}}$, e $x_1 \neq x_2$. Então

$$P + Q = (x_3, y_3) \in E_{/\mathbb{F}_{2^m}},$$

onde

$$\begin{cases} x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \\ y_3 = \lambda(x_1 + x_3) + x_3 + y_1 \\ \lambda = \dfrac{y_2 + y_1}{x_2 + x_1} \end{cases}$$

em $E_{/\mathbb{F}_{2^m}}$.

3. Seja $P = (x_1, y_1) \neq \mathcal{O} \in E_{/\mathbb{F}_{2^m}}$.e $x_1 \neq 0$. Então

$$P + P = 2P = (x_3, y_3),$$

onde

$$\begin{cases} x_3 = \lambda^2 + \lambda + a \\ y_3 = x_1^2 + (\lambda + 1).x_3 \\ \lambda = x_1 + \dfrac{y_1}{x_1} \end{cases}$$

em $E_{/\mathbb{F}_{2^m}}$.

4. Se $x_1 = 0$ então $2P = \mathcal{O}$, pois $-P = P$.

Para o grupo da curva elíptica $E(\mathbb{F}_{2^m})$ dada pela equação

$$y^2 + vy = x^3 + bx + c,$$

onde v, b e $c \in \mathbb{F}_{2^m}$ e $v \neq 0$, tem-se:

1. O inverso de $P = (x_1, y_1) \neq \mathcal{O} \in E_{/\mathbb{F}_{2^m}}$ é

$$-P = (x_1, y_1 + v).$$

2. Seja $P = (x_1, y_1) \in E_{/\mathbb{F}_{2^m}}$, $Q = (x_2, y_2) \in E_{/\mathbb{F}_{2^m}}$, e $x_1 \neq x_2$. Então

$$P + Q = (x_3, y_3) \in E_{/\mathbb{F}_{2^m}},$$

onde

$$\begin{cases} x_3 = \lambda^2 + x_1 + x_2 \\ y_3 = \lambda(x_1 + x_3) + y_1 + v \\ \lambda = \dfrac{y_2 + y_1}{x_2 + x_1} \end{cases}$$

em $E_{/\mathbb{F}_{2^m}}$.

3. Seja $P = (x_1, y_1) \neq \mathcal{O} \in E_{/\mathbb{F}_{2^m}}$. Então

$$P + P = 2P = (x_3, y_3),$$

onde

$$\begin{cases} x_3 = \lambda^2 \\ y_3 = \lambda(x_1 + x_3) + y_1 + v \\ \lambda = \dfrac{x_1^2 + b}{v} \end{cases}$$

em $E_{/\mathbb{F}_{2^m}}$.

Multiplicação por um escalar

Definição 2.5 *Chama-se* multiplicação de um ponto $P \in E$ por um escalar $k \in \mathbb{N}$, *e representa-se por* kP, *à soma de* P *consigo mesmo* k *vezes.*

A multiplicação de $P \in E$ *pelo escalar* $-k$, *é igual a* $-(kP)$, *isto é,*

$$-kP = -(kP).$$

A multiplicação de $P \in E$ *pelo escalar* 0 *é igual ao ponto* $\mathcal{O}$, *isto é,*

$$0P = \mathcal{O}.$$

Para se calcular kP, em primeiro lugar escreve-se k na base 2, isto é,

$$k = k_0 + k_1.2 + k_2.2^2 + k_3.2^3 + ... + k_r.2^r,$$

onde cada k_i, $i = 0, ..., r$, ou é 0 ou é 1.

De seguida calcula-se

$P_0 = P$

$P_1 = 2P_0 = 2P$

$P_2 = 2P_1 = 2^2P$

$P_3 = 2P_2 = 2^3P$

.....

$P_r = 2P_{r-1} = 2^rP$

No final, calcula-se $kP =$ (a soma dos P_i para $k_i = 1$) (ver [21, p. 136]).

Exemplo 2.2

$$171P = \left(2^7 + 2^5 + 2^3 + 2 + 1\right) P = 2^7P + 2^5P + 2^3P + 2P + P.$$

Calcula-se P_1, P_3, P_5 *e* P_7 *e, finalmente,*

$$171P = P_1 + P_3 + P_5 + P_7 + P.$$

No total operou-se 11 *vezes,* 7 *para se obter os* P_i, $i = 1, ..., 7$ *mais* 4 *para se obter a soma,*

$$P_1 + P_3 + P_5 + P_7 + P = 171P.$$

Nota 2.4 *Note-se que* $r \leq \log_2 k$ *e, por conseguinte, calcula-se* kP *em menos do que* $2\log_2 k$ *passos.*

A *ordem de* $P \in E$ é o menor inteiro positivo n tal que $nP = \mathcal{O}$. Uma vez que $E_{/\mathbb{F}_q}$ é um grupo finito, o conjunto

$$\{\mathcal{O}, P, 2P, 3P, ..., (n-1)\,P\}$$

é um subgrupo cíclico de $E_{/\mathbb{F}_q}$ e, consequentemente, n divide a ordem $\#E_{/\mathbb{F}_q}$ do grupo $E_{/\mathbb{F}_q}$ (o cálculo da ordem será tratado mais adiante).

Nota 2.5 *Por ser* $\#E_{/\mathbb{F}_q} = n_1n_2$, *se* n_1 *for igual a* 1, *então* $E_{/\mathbb{F}_q}$ *será um grupo cíclico de ordem* n_2, *isto é, existe* $P \in E_{/\mathbb{F}_q}$ *tal que*

$$E_{/\mathbb{F}_q} = \{kP : 0 \leq k \leq n_2 - 1\};$$

tal ponto P *denomina-se um* gerador do grupo $E_{/\mathbb{F}_q}$. *Se* $n_1 > 1$ *dir-se-á que* $E_{/\mathbb{F}_q}$ *tem* rango 2 *(ver* [10]*).*

Exemplo 2.3 *Seja* $E(\mathbb{F}_{29})$ *a curva elíptica definida pela equação*

$$y^2 = x^3 + 4x + 20.$$

$\#E_{/\mathbb{F}_{29}} = 37$.

Por ser 37 *um número primo, conclui-se que* $E_{/\mathbb{F}_{29}}$ *é um grupo cíclico. Exceptuando o ponto* $\mathcal{O}$*, todo ponto em* $E_{/\mathbb{F}_{29}}$ *é um gerador.*

Definição 2.6 *Seja uma curva elíptica* $E(\mathbb{F}_q)$ *e* m *um inteiro positivo. Diz-se que um elemento*

$$P \in E(\mathbb{F}_q)$$

é um m-ponto de torsão *se e só se* $mP = \mathcal{O}$.

O conjunto dos m-pontos de torsão de $E(\mathbb{F}_q)$ é um subgrupo de $E(\mathbb{F}_q)$ e representa-se por $E[m]$, isto é,

$$E[m] = \{P \in E(\mathbb{F}_q) : mP = \mathcal{O}\}.$$

Lema 2.3 *Seja dada uma curva elíptica* $E(\mathbb{F}_q)$*, e seja* k *um número natural e* m *um número primo diferente da característica* p *de* $\mathbb{F}_q$ *tal que*

$$m \mid \#E_{/\mathbb{F}_q} \qquad e \qquad m \nmid q - 1.$$

Então $E_{/\mathbb{F}_{q^k}}$ *contém* m^2 *pontos de ordem* m *se e só se* m *divide* $q^k - 1$ *(ver [4, p. 43]).*

Teorema 2.4 *Seja* $E(\mathbb{F}_q)$ *uma curva elíptica. Se a característica* p *de* $\mathbb{F}_q$ *for um número que é primo com* $m \geqslant 2$ *então*

$$E[m] \simeq \mathbb{Z}_{/m} \oplus \mathbb{Z}_{/m}.$$

Por outro lado, quando $m = p^r$*, onde* p *é a característica de* $\mathbb{F}_q$*, então*

$$E[p^r] = \{\mathcal{O}\} \qquad ou \qquad E[p^r] = \mathbb{Z}_{/p^r},$$

para todo número natural $r \geq 1$ *(ver [6, p. 273]).*

Exemplo 2.4 *Seja* $E(\mathbb{F}_{2003})$ *definida pela equação*

$$y^2 + 2xy + 8y = x^3 + 5x^2 + 1136x + 531.$$

A ordem do grupo $E_{/\mathbb{F}_{2003}}$ *é* 1956 *e o ponto*

$$P = (1118, 529) \in E_{/\mathbb{F}_{2003}}$$

tem ordem igual a 1956*. Isso implica que o grupo* $E_{/\mathbb{F}_{2003}}$ *é cíclico e gerado por* P.

O "Weil pairing"

Com vista ao estudo de ataques aos criptosistemas baseados em curvas elípticas, vai-se fazer uma breve abordagem aos *pares de Weil*, conceito introduzido por *André Weil*, matemático bem conhecido principalmente pelos seus trabalhos em geometria algébrica e teoria dos números. Para um estudo mais detalhado, pode-se ver [23, p. 339-379].

Seja uma curva elíptica $E(\mathbb{K})$ sobre um corpo $\mathbb{K}$ de característica diferente de zero e seja $m \geqslant 2$ um número inteiro primo com a característica do corpo $\mathbb{K}$. Seja

$$\mu_m = \left\{x \in \overline{\mathbb{K}} : x^m = 1\right\}$$

o grupo dos n-ésimas raízes de unidade em $\overline{\mathbb{K}}$. Uma vez que a característica de $\mathbb{K}$ não divide m, a equação

$$x^m = 1$$

não tem raízes múltiplas, assim sendo, ela tem m raízes distintas em $\overline{\mathbb{K}}$. Então μ_m é um *grupo cíclico de ordem* m. Todo o gerador ζ de μ_m denomina-se uma *raíz primitiva de unidade.*

Teorema 2.5 *Seja uma curva elíptica $E(\mathbb{K})$ sobre um corpo $\mathbb{K}$ de característica diferente de zero e seja $m \geqslant 2$ um número inteiro primo com a característica de $\mathbb{K}$. Existe um par*

$$e_m : \quad E[m] \times E[m] \quad \rightarrow \quad \mu_m,$$

denominado par de Weil *que satisfaz as seguintes propriedades:*

1. *e_m é bilinear em cada variável, isto é,*

$$e_m(S_1 + S_2, T) = e_m(S_1, T)\, e_m(S_2, T)$$

e

$$e_m(S, T_1 + T_2) = e_m(S, T_1)\, e_m(S, T_2)$$

para todo $S, S_1, S_2, T, T_1, T_2 \in E[m]$.

2. *e_m é não-degenerada em cada variável, isto é, se*

$$e_m(S, T) = 1$$

para todo $T \in E[m]$, então

$$S = \mathcal{O}$$

e também se

$$e_m(S, T) = 1$$

para todo $S \in E[m]$, então

$$T = \mathcal{O}.$$

3. *$e_m(T, T) = 1$, para todo $T \in E[m]$.*

4. *$e_m(S, T) = e_m(T, S)^{-1}$, para todo $S, T \in E[m]$.*

5. *$e_m(\sigma(S), \sigma(T)) = \sigma(e_m(S, T))$, para todo $S, T \in E[m]$ e para todo automorfismo σ de $\overline{\mathbb{K}}$ cuja a restrição a $\mathbb{K}$ seja uma função identidade.*

6. *$e_m(\alpha(S), \alpha(T)) = e_m(S, T)^{\deg(\alpha)}$, para todo endomorfismo separável α da curva elíptica $E(\mathbb{K})$ e onde $\deg(\alpha)$ representa o grau de α. Se a curva elíptica E for definida sobre um corpo finito $\mathbb{F}_q$, então a propriedade será válida se α for o* endomorfismo de Frobenius *ϕ_q (ver [23]).*

Corolário 2.6 *Se S, T formar a base de $E[m]$, então $e_m(S,T)$ será uma raíz primitiva m-ésima de unidade.*

Corolário 2.7 *Se $E[m] \subseteq E_{/\mathbb{K}}$ então $\mu_m \subset \mathbb{K}$.*

Teorema 2.8 *Seja uma curva elíptica $E(\mathbb{F}_q)$, $P \in E[m]$, onde m é primo com q, e seja μ_m o grupo das m-ésimas raízes da unidade em $\overline{\mathbb{F}_q}$.*

A. *Existe $R \in E[m]$ tal que $e_m(P,R)$ é uma m-ésima raíz primitiva da unidade.*

B. *A função*

$$\begin{array}{rccc} \theta: & \langle P \rangle & \to & \mu_m \\ & Q & \mapsto & e_m(Q,R), \end{array}$$

onde $\mu_m \subseteq \mathbb{F}_{q^k}$ é um isomorfismo de grupo.

2.2 A ordem do grupo de uma curva elíptica sobre um corpo finito

A determinação da ordem do grupo $E_{/\mathbb{F}_q}$ de uma curva elíptica $E(\mathbb{F}_q)$ é uma tarefa muito difícil para valores de q muito grandes — por exemplo para q com 100 ou mais dígitos decimais.

O conhecimento da ordem $\#E_{/\mathbb{F}_q}$ do grupo $E_{/\mathbb{F}_q}$ é muito importante para o estudo da primalidade e factorização de um número natural, bem como para o estudo do problema do logaritmo discreto no grupo $E_{/\mathbb{F}_q}$ (que será tratado mais adiante)

Uma vez que a equação de Weierstrass 2.1 tem no máximo duas soluções para cada $x \in \mathbb{F}_q$ então

$$\#E_{/\mathbb{F}_q} \in [1, 2q+1].$$

Nota-se que se $q = p$, $p > 3$

$$\#E_{/\mathbb{F}_q} = 1 + \sum_{x \in \mathbb{F}_p} \left(\left(\frac{x^3 + bx + c}{p} \right) + 1 \right),$$

onde $\left(\frac{x^3 + bx + c}{p} \right)$ é o *símbolo de Legendre.*

Exemplo 2.5 $\#E(0,1,3)_{/\mathbb{F}_{11}} = 1 + \sum_{x \in \mathbb{F}_{11}} \left(\left(\frac{x^3 + x + 3}{11} \right) + 1 \right) = 1 + 11 + +6 = 18.$

Contudo, esta fórmula não é muito prática para valores de p muito grandes.

Mantendo-se o corpo $\mathbb{F}_q$, a ordem de uma curva elíptica $E(\mathbb{F}_q)$ varia muito se se mudar os coeficientes a, b e c do polinómio

$$f(x) = x^3 + ax^2 + bx + c,$$

que constitui o segundo membro da equação que representa $E(\mathbb{F}_q)$.

No entanto, o teorema seguinte dá uma boa estimativa para $\#E_{/\mathbb{F}_q}$.

Teorema 2.9 (Helmut Hasse) *A ordem* $\#E_{/\mathbb{F}_q}$ *de uma curva elíptica em* $\mathbb{F}_q$ *é tal que*

$$\#E_{/\mathbb{F}_q} = q + 1 - t,$$

onde $|t| \leq 2\sqrt{q}$. *Isto é,*

$$\#E_{/\mathbb{F}_q} \in [q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}]\,.$$

O intervalo

$$[q + 1 - 2\sqrt{q},\ q + 1 + 2\sqrt{q}]$$

é denominado intervalo de Hasse *e* t *é denominado* traço da curva elíptica $E(\mathbb{F}_q)$ *(ver [6, p. 278] ou [10, p. 82]).*

O teorema que se segue dá uma boa margem para determinação do traço t da curva elíptica $E(\mathbb{F}_q)$.

Teorema 2.10 *Seja* $\mathbb{F}_q$ *um corpo finito onde* $q = p^m$. *Existe uma curva elíptica* $E(\mathbb{F}_q)$ *tal que*

$$\#E_{/\mathbb{F}_q} = q + 1 - t$$

se e só se ocorrer uma das condições que se seguem:

i) $t \not\equiv 0 \pmod{p}$ *e* $t^2 \leq 4q$.

ii) m *é um* número ímpar *e ocorre uma das seguintes condições:*

a) $t = 0$ *ou*

b) $t^2 = 2q$ *e* $p = 2$ *ou*

c) $t^2 = 3q$ *e* $p = 3$.

iii) m *é um* número par *e ocorre uma das seguintes condições:*

a) $t^2 = 4q$ ou

b) $t^2 = p$ e $p \not\equiv 1 \pmod{3}$ ou

c) $t = 0$ e $p \not\equiv 1 \pmod{4}$ (ver [10, p. 82]).

Uma das consequências do teorema anterior é a seguinte.

Corolário 2.11 *Para todo o número primo* p *e traço* t *tal que* $|t| \leq 2\sqrt{p}$, *existe uma curva elíptica* $E(\mathbb{F}_p)$ *tal que*

$$\#E_{/\mathbb{F}_p} = p + 1 - t.$$

Isto é, qualquer inteiro n *pertencente ao intervalo de Hasse*

$$[p + 1 - 2\sqrt{p},\ p + 1 + 2\sqrt{p}]$$

é a ordem de um grupo $E_{/\mathbb{F}_p}$ *de uma curva elíptica* $E(\mathbb{F}_p)$.

Assim sendo, existe uma curva elíptica $E(\mathbb{F}_p)$ *tal que* $\#E_{/\mathbb{F}_p} = p + 1$.

Exemplo 2.6 *Seja $p = 37$ e o grupo $E_{/\mathbb{F}_{37}}$ da curva elíptica $E(\mathbb{F}_{37})$. A tabela seguinte apresenta para cada número inteiro n pertencente ao intervalo de Hasse*

$$\left[37 + 1 - 2\sqrt{37}, 37 + 1 + 2\sqrt{37}\right],$$

o correspondente valor de b e c tal que $\#E_{/\mathbb{F}_p}(0, b, c) = n$;

n	(b,c)	n	(b,c)	n	(b,c)	n	(b,c)	n	(b,c)
26	(5,0)	31	(2,8)	36	(1,0)	41	(1,16)	46	(1,11)
27	(0,9)	32	(3,6)	37	(0,5)	42	(1,9)	47	(3,15)
28	(0,6)	33	(1,13)	38	(1,5)	43	(2,9)	48	(0,1)
29	(1,12)	34	(1,18)	39	(0,3)	44	(1,7)	49	(0,2)
30	(2,2)	35	(1,8)	40	(1,2)	45	(2,14)	50	(2,0)

Se a curva elíptica E for definida sobre $\mathbb{F}_q$ também ela será definida sobre uma extensão $\mathbb{F}_{q^n}$ de $\mathbb{F}_q$. O grupo $E_{/\mathbb{F}_q}$ é um subgrupo do grupo $E_{/\mathbb{F}_{q^n}}$, logo $\#E_{/\mathbb{F}_q}$ divide $\#E_{/\mathbb{F}_{q^n}}$. Se $\#E_{/\mathbb{F}_q}$ for conhecido poder-se-á determinar $\#E_{/\mathbb{F}_{q^n}}$, conforme o teorema que se segue (ver [10]).

Teorema 2.12 *Seja uma curva elíptica $E(\mathbb{F}_q)$ e $\#E_{/\mathbb{F}_q} = q + 1 - t$. Então*

$$\#E_{/\mathbb{F}_{q^n}} = q^n + 1 - V_n,$$

para todo $n \geq 2$, onde $\{V_n\}$ é uma sequência definida recursivamente por

$$\begin{cases} V_0 = 2 \\ V_1 = t \\ V_n = V_1 V_{n-1} - qV_{n-2} \qquad \text{para todo } n \geq 2. \end{cases}$$

Definição 2.7 *Uma curva elíptica $E(\mathbb{F}_q)$ diz-se* supersingular *se a característica p do corpo $\mathbb{F}_q$ dividir o traço t da curva. Se p não dividir t, dir-se-á que a curva é* não-supersingular.

Proposição 2.13 *Equivalentemente, pode dizer-se que uma curva é supersingular se e só se:*

(i) *$p = 2$ ou $p = 3$ e $j(E) = 0$.*

(ii) *$p \geq 5$ e $t = 0$ (ver [4]).*

Nota 2.6 *Uma curva elíptica $E(\mathbb{F}_{2^m})$ definida pela equação 2.8, ela é uma curva supersingular, mas se ela for definida pela equação 2.7, ela será uma curva não-supersingular.*

Exemplo 2.7 *A curva elíptica $E(\mathbb{F}_p)$,para um primo p maior ou igual a 5 e $p \equiv 2 \pmod 3$, definida pela equação $y^2 = x^3 + 1$, é uma curva supersingular. O grupo $E_{/\mathbb{F}_p}$ é um grupo cíclico de ordem $p + 1$.*

Definição 2.8 *Uma curva elíptica $E(\mathbb{F}_q)$ diz-se* anómala *se*

$$\#E_{/\mathbb{F}q} = q,$$

isto é, se o traço da curva t for igual a 1.

2.3 O problema do logaritmo discreto em curvas elípticas

O problema oposto à multiplicação de um ponto $P \in E_{/\mathbb{F}_q}$por um escalar $k \in \mathbb{N}$ é o seguinte:

Dado $P \in E_{/\mathbb{F}_q}$ de ordem n e $Q = lP$, determinar o número l.

Definição 2.9 *Seja um grupo $E_{/\mathbb{F}_q}$ e seja $P \in E_{/\mathbb{F}_q}$ de ordem n e $Q = lP$. O* problema do logaritmo discreto — *PLD — sobre $E_{/\mathbb{F}_q}$ consiste na determinação do menor inteiro l,*

$$0 \leq l \leq n-1,$$

tal que $lP = Q$. Tal número l é denominado logaritmo discreto — LD — *de Q na base P e representa-se por*

$$l = \mathrm{dlog}_P(Q).$$

A segurança de um criptosistema baseado em curvas elípticas baseia-se na dificuldade em resolver este problema — de difícil resolução.

A forma mais "ingénua" de resolver o PLD $Q = lP$, é determinar $2P$, $3P$, $4P$, ..., até obter o ponto Q. Em média dá-se $\frac{n}{2}$ passos para se obter o ponto Q, o que será muito se n for muito grande — como é o caso real de um criptosistema baseado em curvas elípticas, onde $n \geq 2^{160}$.

Os métodos utilizados para resolver os PLD, de uma forma geral, requerem um grupo finito e comutativo G. Assim sendo, pode-se, também, aplicar esses métodos a um grupo $E_{/\mathbb{F}_q}$. Contudo, a complexidade da adição de pontos num grupo $E_{/\mathbb{F}_q}$ — bem como o cálculo do dobro de um ponto — torna esses métodos muito lentos quando aplicados à resolução de um PLD num grupo $E_{/\mathbb{F}_q}$.

Os algoritmos utilizados para resolver o PLD num grupo $E_{/\mathbb{F}_q}$ são agrupados em dois:

Grupo 1. Os *algoritmos de carácter específico* cujo tempo de execução e a própria aplicação depende de determinados tipos de parâmetros da curva elíptica $E(\mathbb{F}_q)$.

Grupo 2. Os *algortmos de carácter geral* cujo tempo de execução depende apenas to tamanho de cada parâmetro da curva elíptica.

Algoritmos de carácter geral.

A simplificação de Pohlig e Hellman.

Pohlig e Hellman chegaram à conclusão que para resolver um PLD num grupo comutativo finito G, basta resolver esse problema nalguns seus subgrupos, cujas ordens são potências de números primos, e aplicar o *Teorema Chinês dos Restos — TCR.* Além do mais o problema pode ser reduzido ao caso de subgrupos cujas ordens são números primos, como se pode ver mais abaixo.

Seja o grupo comutativo finito G de ordem n gerado pelo ponto P e seja $Q \in G$ tal que $Q = lP$ Para além disso deve-se conhecer a factorização prima de n,

$$n = \prod_i p_i^{e_i}.$$

Quer determinar-se $l = \mathrm{dlog}_P(Q)$.

Seja p um número primo e p^e a maior potência de p que divide n. Escreve-se l na base p como

$$l = l_0 + l_1 p + l_2 p^2 + ...$$

com $0 \leq l_i < p$. Determina-se o valor de $l \pmod{p^e}$ determinando sucessivamente l_0, l_1, l_2, ..., l_{e-1}.

O procedimento é o seguinte.

Passo 1 Determina-se $T = \left\{ k\left(\frac{n}{p}P\right) : 0 \leq k \leq p-1 \right\}$.

Passo 2 Determina-se $\frac{n}{p}Q$, que é igual a $l_0\left(\frac{n}{p}P\right)$ de T.

Passo 3 Se $e = 1$ pára-se, senão continua-se.

Passo 4 Seja $Q_1 = Q - l_0 P$.

Passo 5 Determina-se $\frac{n}{p^2}Q_1$, que é igual a $l_1\left(\frac{n}{p}P\right)$ de T.

Passo 6 Se $e = 2$ pára-se, senão continua-se.

Passo 7 Supõe-se que já se calculou l_0, l_1, ..., l_{r-1} e Q_1, Q_2, ..., Q_{r-1}.

Passo 8 Seja $Q_r = Q_{r-1} - l_{r-1}p^{r-1}P$.

Passo 9 Determina-se l_r tal que $\frac{n}{p^{r+1}}Q_r = l_r\left(\frac{n}{p}P\right)$.

Passo 10 Se $r = e - 1$, pára-se, senão volta-se ao *Passo 7*.

Então

$$l \equiv l_0 + l_1 p + ... + l_{e-1}p^{e-1} \pmod{p^e},$$

pois

$$l - \left(l_0 + l_1 p + ... + l_{e-1}p^{e-1}\right) = p^e\left(l_e + l_{e+1}p + l_{e+2}p^2 + ...\right).$$

Resolvendo problemas de logaritmo discreto em subgrupos de ordem p vai-se determinar $l \pmod{p^e}$. Depois de determinar $l \pmod{p^e}$ para todo número primo p divisor de n, a solução inicial, l, do PLD $Q = lP$ é obtida aplicando o TCR.

Nota 2.7 *Este método apesar de parecer muito prático, tem duas questões a considerar. A primeira tem a ver com o conhecimento da ordem do grupo G, pois a determinação da ordem do grupo de uma curva elíptica, como já se disse atrás, é uma tarefa muito difícil. A segunda tem a ver com a factorização dessa ordem, pois a factorização é, também, um problema de difícil resolução. Além do mais se a ordem n de G tiver um divisor primo muito grande, o grau da dificuldade da resolução do PLD dado pela simplificação de Pollig e Hellman é quase igual ao do PLD inicial.*

Exemplo 2.8 *Seja* $E_{/\mathbb{F}_{1009}}(0,71,602)$ *de ordem* 1060. *Considera-se o ponto*

$$P = (1,237) \in E_{/\mathbb{F}_{1009}}(0,71,602)$$

de ordem

$$n = 530 = 2 \times 5 \times 53$$

e

$$Q = (190,271) \in E_{/\mathbb{F}_{1009}}(0,71,602).$$

Seja $Q = lP$, *onde* $1 \leq l \leq 529$. *Procede-se da seguinte forma:*

Para o divisor primo $p = 2$ *tem-se*

$$\frac{n}{p}Q = 265Q = (50,0) = 265P,$$

então

$$l \equiv 1 \pmod 2.$$

Para o divisor primo $p = 5$ *tem-se*

$$\frac{n}{p}Q = 106Q = (639,849) = 4\left(\frac{n}{p}P\right),$$

então

$$l \equiv 4 \pmod 5.$$

Para o divisor primo $p = 53$ *tem-se*

$$\frac{n}{p}Q = 10Q = (592,97) = 48\left(\frac{n}{p}P\right),$$

então

$$l \equiv 48 \pmod{53}.$$

A solução PLD $Q = lP$ *é dada pela determinação de um inteiro positivo inferior a* 530 *que seja congruente com* 1, 4 *e* 48 *modulo* 2, 5 *e* 48, *respectivamente, isto é, pela resolução do sistema*

$$\begin{cases} l \equiv 1 \pmod 2 \\ l \equiv 4 \pmod 5 \\ l \equiv 48 \pmod{53} \end{cases}$$

Assim sendo, obtém-se $l = 419$.

Nota 2.8 *Note-se que em vez de determinar o conjunto T para cada p divisor de n — o que é pouco prático para valores de p muito grandes — no intuito de resolver o problema do logaritmo discreto no subgrupo de ordem p, pode-se utilizar outros métodos de resolução do PLD - como os que abaixo são tratados.*

O método de Shanks - Baby Step / Giant Step (BSGS).

Desenvolvido por *D. Shanks*, este método é aplicado para resolver o PLD num grupo comutativo finito G de ordem n, e requer, aproximadamente, $\sqrt{n}$ passos e armazenamento.

Seja G um grupo comutativo finito de ordem n gerado por P e seja $Q \in G$ tal que $Q = lP$.

Quer determinar-se $l = \mathrm{dlog}_P(Q)$.

Através da divisão euclidiana pode obter-se

$$l = \lceil \sqrt{n} \rceil a + b$$

onde $0 \leq a, b < \lceil \sqrt{n} \rceil$.

Então, a equação inicial, $Q = lP$, pode ser escrita na seguinte forma,

$$(Q - bP) = a\left(\lceil \sqrt{n} \rceil P\right).$$

Passo 1 Vai-se construir uma tabela com todos os valores

$$R_b = Q - bP \text{ - } os \text{ "baby steps"},$$

para $0 \leq b \leq \lceil \sqrt{n} \rceil - 1$.

Passo 2 Depois vão ser determinados os valores da forma

$$S_a = a\left(\lceil \sqrt{n} \rceil P\right) \text{ - } os \text{ "giant steps"},$$

com $0 \leq a \leq \lceil \sqrt{n} \rceil - 1$.

Passo 3 Cada vez que se determina um giant step, verifica-se se o referido valor não aparece na tabela dos baby steps. Quando isso acontecer os valores de a e de b serão determinados. Assim sendo, ter-se-á

$$l = \lceil \sqrt{n} \rceil a + b$$

com os valores concretos de a, b e $\lceil \sqrt{n} \rceil$.

Nota 2.9 *Ao contrário do método de simplificação de Pollig e Hellman, a aplicação deste método não requer necessariamente o conhecimento da ordem do grupo G, basta ter um valor $m \geq \sqrt{n}$, onde n é um limite superior da ordem de G. Assim sendo, este método aplica-se perfeitamente ao grupo $E_{/\mathbb{F}_q}$, pois*

$$m \geq q + 1 + 2\sqrt{q},$$

tendo em conta o teorema de Helmut Hasse. Por exemplo, para resolver um PLD em $E_{/\mathbb{F}_{41}}$ pode-se considerar $n = 54$. Em contrapartida, este método requer o armazenamento de muitos valores quando se constroi a tabela dos "baby ateps" e "giant steps".

Exemplo 2.9 *Seja*

$$P = (32, 737), Q = (592, 97) \in E_{/\mathbb{F}_{1009}}(0, 71, 602).$$

Seja o subgrupo G, de ordem 53, de $E_{/\mathbb{F}_{1009}}(0, 71, 602)$ gerado por P e seja $Q = lP$. Quer determinar-se $l = \mathrm{dlog}_P(Q)$. Tem-se

$$l = \lceil \sqrt{53} \rceil a + b = 8a + b.$$

Primeiro determina-se a tabela dos valores baby steps, $R_b = Q - bP$ *onde* $0 \leq b \leq 7$.

Os valores de giant steps, $S_a = a\,(8P)$ *são determinados e guardados numa tabela. Sempre que se determina um giant step verifica-se a sua ocorrência na tabela dos baby steps, pois o algoritmo termina quando isso acontecer.*

Tem-se então

b	$R_b = Q - bP$	a	$S_a = a\,(8P)$
0	(592, 97)	0	$\mathcal{O}$
1	(728, 450)	1	(996, 855)
2	(728, 450)	2	(200, 652)
3	(996, 154)	3	(378, 304)
4	(817, 136)	4	(609, 357)
5	(365, 715)	5	(304, 583)
6	(627, 606)	6	(592, 97)
7	(150, 413)		

A igualdade obtém-se quando $b = 0$ *e* $a = 6$, *tem-se então*

$$(855 + 154) \div 1060 = 0.951\,89...$$

$$l = 8a + b = 8 \times 6 = 48.$$

Nota 2.10 *Note-se que não é necessário determinar os giant steps para* $a > 6$. *Também, poder-se-ia ter parado em* S_1. *Nesse caso ter-se-ia*

$$S_1 = -R_3 \Leftrightarrow 8P = -Q + 3P \Leftrightarrow Q = -5P$$

então $l \equiv -5 \pmod{53} \equiv 48 \pmod{53}$, logo

$$l = 48.$$

Adaptação do BSGS quando se sabe que o LD ou a ordem de G estão num determinado intervalo Às vezes sabe-se de antemão que o LD l ou a ordem n do grupo G estão num intervalo $[h, i]$. É o caso concreto das curvas elípticas, pois

$$q + 1 - 2\sqrt{q} \leq \#E_{/\mathbb{F}_q} \leq q + 1 - 2\sqrt{q}.$$

Nesse caso os baby steps são da forma

$$R_b = Q - (h + b)\,P$$

e os giant steps são da forma

$$S_a = a\left(\left\lceil\sqrt{i-h}\right\rceil P\right),$$

onde $0 \leq a, b \leq \left\lceil\sqrt{i-h}\right\rceil - 1$.

Quando R_b for igual a S_a ter-se-á

$$l = h + b + a \times \left\lceil\sqrt{i-h}\right\rceil.$$

Os métodos de Pollard - ρ e λ.

O método ρ

Desenvolvido por Pollard, este método, também, é aplicado para resolver o PLD num grupo comutativo finito G de ordem n. Tem quase a mesma complexidade de tempo que o método de Shanks, mas consome menos recursos no armazenamento de dados.

Seja G um grupo comutativo finito de ordem n, gerado por P, e seja $Q \in G$. Seja $Q = lP$, onde se quer determinar o LD $l = \text{dlog}_P(Q)$.

O método baseia-se na tiragem aleatória com reposição de elementos em G. Quando um determinado elemento, depois da sua primeira tiragem, voltar a ser tirado, dir-se-á que houve uma *colisão*.

Os elementos aleatórios em G são da forma

$$\{P_i = a_i P + b_i Q\}_{i \geq 0}$$

para inteiros a_i e b_i já conhecidos. Quando se obtiver a colisão $P_{j_0} = P_{i_0}$, ter-se-á

$$a_{j_0} P + b_{j_0} Q = a_{i_0} P + b_{i_0} Q \Leftrightarrow (a_{j_0} - a_{i_0})\, P = (b_{i_0} - b_{j_0})\, Q.$$

Assim, se $mdc\ (b_{i_0} - b_{j_0}, n) = d$ obter-se-á

$$l \equiv \frac{a_j - a_i}{b_i - b_j} \left(\text{mod } \frac{n}{d}\right).$$

Assim sendo, ter-se-á d escolhas para o número l. Se d for um número inteiro pequeno (tendo em conta os recursos computacionais existentes) poder-se-á experimentar todos os possíveis valores até se obter $Q = lP$.

Nota 2.11 *Num criptosistema baseado em curvas elípticas, muitas vezes, n é um número primo —embora, geralmente, $\#E_{/\mathbb{F}_q} = c \times p$, para um primo p muito grande e um número inteiro positivo c pequeno — e nesse caso, $d = 1$ ou $d = n$. Se $d = n$, os coeficientes de P e de Q são múltiplos de n, logo obter-se-á uma relação trivial. Se $d = 1$, obter-se-á o valor de l procurado.*

Para se obter os elementos aleatórios escolhe-se uma função $f : G \longrightarrow G$ que se comporta como uma função aleatória. Começa-se com um elemento P_0 e calcula-se as iterações $P_{i+1} = f(P_i)$. Como G é finito, há-de haver dois indices i_0 e j_0, $i_0 < j_0$ tal que $P_{i_0} = P_{j_0}$ e assim

$$P_{i_0+1} = f(P_{i_0}) = f(P_{j_0}) = P_{j_0+1} \text{ e,}$$

$$P_{i_0+l} = P_{j_0+l},$$

para todo $l \geq 0$. Assim sendo, a sequência P_i será periódica de período $j_0 - i_0$ (ou um divisor de $j_0 - i_0$).

Uma forma para se obter os elementos aleatórios é seguinte (ver em [6, p. 489]):

Seja $r > 1$ número inteiro — pode ser $3 \leq r \leq 100$ — e uma partição de G em $G_1, ..., G_r$ cujas ordens são aproximadamente iguais. Considera-se os elementos

$$M_s = m_s P + n_s Q$$

com m_s e n_s inteiros aleatórios escolhidos no intervalo $[0, n-1]$ e $1 \leq s \leq r$. Tem-se então

$$P_{i+1} = f(P_i) = P_i + M_s \qquad \text{se } P_i \in G_s.$$

Exemplo 2.10 *Seja*

$$P = (32, 737), Q = (592, 97) \in E_{/\mathbb{F}_{1009}}(0, 71, 602).$$

Seja G o subgrupo, de ordem 53, de $E_{/\mathbb{F}_{1009}}(0, 71, 602)$ gerado por P e seja $Q = lP$. Quer-se determinar $l = \mathrm{d}\log_P(Q)$.

Escolhe-se

$$r = 3,$$

$$f : \begin{cases} E_{/\mathbb{F}_{1009}}(0, 71, 602) & \longrightarrow & E_{/\mathbb{F}_{1009}}(0, 71, 602) \\ (x, y) & \longmapsto & (x, y) + M_{x \pmod 3 + 1} \end{cases}$$

e tem-se

$$M_1 = 2P + 0Q = (8, 623),$$
$$M_2 = 1P + 1Q = (654, 118),$$
$$M_3 = 3P + 4Q = (555, 82).$$

Calcula-se os "elementos aleatórios" partindo-se de $P_0 = P$ e aplicando a relação

$$P_{i+1} = f(P_i) = P_i + M_s, \tag{2.9}$$

onde $s = x \pmod 3 + 1$.

i	$P_i = a_iP + b_iQ$
0	$1P + 0Q = (32, 737)$
1	$4P + 4Q = (200, 357)$
2	$7P + 8Q = (759, 545)$
3	$9P + 8Q = (241, 691)$
4	$10P + 9Q = (711, 716)$
5	$12P + 9Q = (759, 545)$

Tem-se então

$$P_2 = P_5 \Leftrightarrow 7P + 8Q = 12P + 9Q,$$

logo

$$l = \frac{12 - 7}{8 - 9} \pmod{53} = -5 \pmod{53} = 48.$$

Nota 2.12 *Segundo [23, p. 148], em vez de se armazenar todos os elementos aleatórios P_i, poder-se-á determinar o par (P_i, P_{2i}) para $i = 1, 2, 3, ...$ e se armanezará apenas o último par calculado. Para a comparação dos seus componentes P_i e P_{2i}. O par $(P_{i+1}, P_{2(i+1)})$ calcular-se-á da seguinte forma,*

$$P_{i+1} = f(P_i), \qquad P_{2(i+1)} = f(f(P_{2i})).$$

Isso melhora a complexidade de espaço mas , em contrapartida, conduz a um pouco mais de cálculo o que vai piorar a complexidade de tempo.

A justificação é seguinte:

Supondo que $i > i_0$ e que i é um múltiplo de d, então $2i$ e i diferem por um múltiplo de d. Como f é periódica de período d, então haverá uma colisão $P_i = P_{2i}$ — e como $d \leq j_0$ e $i_0 < j_0$ haverá colisão para $i < j_0$.

Exemplo 2.11 *Vai determinar-se* $l = \mathrm{d}\log_P(Q)$, *nas condições do exemplo anterior. Compara-se:*

i	P_i	P_{2i}
1	$(200, 357)$	$(759, 545)$
2	$(759, 545)$	$(711, 716)$
3	$(241, 691)$	$(241, 691)$

Tem-se a relação

$$P_3 = P_6 \Leftrightarrow 9P + 8Q = 14P + 9Q \Longleftrightarrow -5P = Q,$$

então

$$l \equiv -5 \pmod{53} = 48.$$

Nota 2.13 *A complexidade de tempo do método* ρ — de Pollard — *é* $\frac{\sqrt{\pi n}}{2}$ *passos, onde* n *é a ordem do ponto* P — *da equação* $Q = lP$. *Se o algoritmo for executado em* r *processadores em paralelo o tempo de execução será* $\frac{\sqrt{\pi n}}{2r}$ *passos. O* método rho quando executado em r processadores em paralelo é considerado o melhor método para atacar um criptosistema baseado em curvas elípticas *de uma forma geral (ver em [14, p. 9]).*

O método λ

Da mesma forma que o anterior, usa-se a função aleatória f. Só que em vez de se utilizar apenas um ponto inicial P_0, usa-se vários pontos iniciais, $P_0^{(1)}, ..., P_0^{(r)}$. Assim sendo, há sequências de elementos aleatórios definidas por

$$P_{i+1}^{(l)} = f\left(P_i^{(l)}\right), \qquad 1 \leq l \leq r, \qquad i = 0, 1, ...$$

Para pôr este método em prática, em vez de um computador, necessita-se de r computadores a funcionarem em paralelo. Quando a colisão for obtida entre os elementos aleatórios gerados pelos vários computadores, ter-se-á então a relação que permitirá resolver o PLD, como acontece no método rho.

Quando r for igual a dois, isto é, quando houver apenas duas sequências de elementos aleatórios, as duas sequências poderão coincidir num ponto e, assim sendo, coincidirão para todos os outros pontos a partir desse.

Utilizando o método λ, a colisão acontece no máximo em $\sqrt{n}$ passos.

O método de λ é, muitas vezes, denominado *método de canguru.*

Algoritmos de carácter específico.

Os algoritmos de carácter específico consistem no estabelecimento de isomorfismos entre o grupo $E_{/\mathbb{F}_p}$ e um grupo G, onde o PLD é mais fácil de resolver. Geralmente, as complexidades de tempo desses algoritmos são *sub-exponenciais.* Sendo assim, esses algoritmos, quando bem implementados, são considerados verdadeiros *ataques* aos criptosistemas para os quais foram concebidos.

Seja $P, Q \in E_{/\mathbb{F}_p}$ e seja o grupo $\langle P \rangle$, de ordem n, gerado por P e $Q = lP$. Se se estabeler, convenientemente, um isomorfismo

$$\Psi : \langle P \rangle \longrightarrow G,$$

então

$$\mathrm{d}\log_P(Q) = \mathrm{d}\log_{\Psi(P)} \Psi(Q).$$

Vai-se destacar dois desses ataques (ver [23, p. 143-165]):

1. *Ataque a um criptosistema baseado em curvas anómalas* $E(\mathbb{F}_p)$, *onde* p *é um número primo.*

2. *Ataque a um criptosistema baseado em curvas supersingulares — O ataque de MOV*

O ataque de MOV – derivado de Menezes, Okamoto e Vanstone

O *ataque de MOV* é um método utilizado para resolver o PLD num grupo de uma curva elíptica. O método consiste na redução do PLD num grupo $E_{/\mathbb{F}_q}$ a um PLD num grupo $\mathbb{F}_{q^k}^{\times}$,para um certo valor de k, utilizando o "*Weil pairing*". Essa redução é muito útil tendo em conta que o PLD no grupo $\mathbb{F}_{q^k}^{\times}$ pode ser resolvido, mais facilmente — utilizando o método de "*index calculus*" (ver em [23, p. 144]), se o inteiro k não for muito grande.

Como $\overline{\mathbb{F}_q} = \cup_{i\geq 1}\mathbb{F}_{q^i}$, a ideia é obter o menor valor de k tal que $E[n] \subseteq \mathbb{F}_{q^k}$, onde n é a ordem do grupo $\langle P\rangle$ gerado por $P \in E_{/\mathbb{F}_q}$— onde se quer resolver o PLD dado pela equação $Q = lP$.

Tem-se então:

Passo 1. Determinar o menor inteiro k tal que $E[n] \subseteq E_{/\mathbb{F}_{q^k}}$.

Passo 2. Determinar um ponto $R \in E[n]$ tal que $g = e_n(P, R)$ tem ordem n.

Passo 3. calcular $a = e_n(Q, R)$.

Passo 4. Calcular

$$l = \mathrm{d}\log_g(a) \qquad \text{em } \mathbb{F}_{q^k}.$$

Note-se que o l obtido no *Passo 4* é realmente o LD de Q na base P, pois:

$$\begin{aligned} a &= e_n(Q, R) \\ &= e_n(lP, R) \\ &= e_n(P, R)^l \qquad \text{(por bilinearidade)} \\ &= g^l. \end{aligned}$$

Nota 2.14 *O tempo de execução deste algoritmo é, em geral, exponencial em* $\log q$. *Isso deve-se, sobretudo, ao facto de não se ter apresentado um algoritmo para obter o ponto* R *e o inteiro positivo* k *— cujo tempo de execução é, em geral, exponencialmente grande.*

No entanto para uma curva supersingular *o tempo de execução diminui-se consideravelmente. O valor de* R *pode ser obtido com mais facilidade tendo em conta a pouca diversidade da estrutura de grupo. Um grupo de uma curva elíptica é, em geral, isomorfo a um grupo da forma* $\mathbb{Z}_{/d_1} \oplus \mathbb{Z}_{/d_2}$, *para* $d_1 \mid d_2$ *e* $d_1 \mid q-1$. *A extensão do grupo isomorfo a um grupo de uma curva elíptica supersingular é da forma* $\mathbb{Z}_{/c\times d_1} \oplus \mathbb{Z}_{/c\times d_1}$, *para um valor de* c *conveniente. Isso limita a escolha do ponto* R.

As curvas supersingulares são divididas em seis categorias e para cada uma dessas pode determinar-se o valor de k *tal que* $E[n] \subseteq E_{/\mathbb{F}_{q^k}}$. *O valor de* k,*nesse caso, é menor ou igual a seis, permitindo assim, determinar mais rapidamente o valor de* k *(ver [23, p. 131]).*

Para o caso de uma curva elíptica supersingular $E(\mathbb{F}_q)$com traço t igual a zero, tem-se a seguinte proposição.

Proposição 2.14 *Seja o grupo $E_{/\mathbb{F}_q}$ tal que $\#E_{/\mathbb{F}_q} = q+1-t$, com $t=0$. Se existir um ponto $P \in E_{/\mathbb{F}_q}$ de ordem n, então $E[n] \subseteq E_{/\mathbb{F}_{q^2}}$ (ver [23, p. 156]).*

O ataque de MOV relativamente a um grupo $E_{/\mathbb{F}_q}$ de uma curva elíptica supersingular $E(\mathbb{F}_q)$ é assim apresentado.

Seja o subgrupo $\langle P \rangle$, de ordem n, gerado por $P \in E_{/\mathbb{F}_q}$ — grupo de uma curva elíptica supersingular $E(\mathbb{F}_q)$ — e $Q = lP$. Quer determinar-se o l.

Passo 1. Determinar o menor valor de k tal que $E[n] \subseteq \mathbb{F}_{q^k}$.

Passo 2. Tirar aleatoriamente um ponto $R' \in E_{/\mathbb{F}_{q^k}}$ e determinar $R = \left(\frac{c \times n_1}{n}\right) R'$.

Passo 3. Calcular $g = e_n(P, R)$ e $a = e_n(Q, R)$.

Passo 4. Calcular

$$l' = \mathrm{d}\log_g a \qquad \text{em } \mathbb{F}_{q^k}.$$

Passo 5. Verificar se $Q = l'P$. Em caso afirmativo, $l = l'$. Caso contrário, a ordem de g deverá ser menor do que n, logo voltar-se-á ao *Passo 2* e escolher-se-á um novo ponto R.

Nota 2.15 *Embora eficaz, pode-se evitar esse tipo de ataque, verificando se o n não divide $q^k - 1$, para pequenos valores de k, onde o PLD em $\mathbb{F}_{q^k}^{\times}$ é menos difícil de resolver. Considera-se que o PLD em $\mathbb{F}_{q^k}^{\times}$ tem o mesmo ou até maior grau de dificuldade que o inicial, se k for maior ou igual a 20.*

Por isso é que um criptosistema baseado em curvas supersingulares com $t = 0$ é considerado vulnerável ao ataque de MOV.

Outro tipo de criptosistema vulnerável a esse tipo de ataque, é aquele que é baseado em curvas de traço 2, isto é, $\#E_{/\mathbb{F}_q} = q - 1$ [14, p. 12]

Note-se que para além do ataque do MOV, existe um outro semelhante baseado no conceito "Tate pairing" *(ver [23, p. 90, 157]), evocando o nome de* John Tate, *matemático contemporâneo conhecido principalmente pelas suas contribuições em teoria algébrica de números e geometria algébrica.*

Ataque à um criptosistema baseado em curvas anómalas $E(\mathbb{F}_p)$, onde p é um número primo.

Seja $E(\mathbb{F}_p)$ uma curva anómala, onde p é um número primo. Segundo [3, p. 13], . Semaev, Smart e Satoh e Araki, independentemente, mostraram como se deve estabelecer um isomorfismo eficiente entre o grupo $E_{/\mathbb{F}_p}$ e o grupo aditivo $\mathbb{F}_p^+$. O isomorfismo

$$\Psi : E_{/\mathbb{F}_p} \longrightarrow \mathbb{F}_p^+.$$

permite obter um algoritmo para a resolução do PLD em $E_{/\mathbb{F}_p}$com tempo de execução polinomial. Por isso é que curvas anómalas definidas sobre o corpo $\mathbb{F}_p$, com p primo, são evitadas em criptosistemas baseados em curvas elípticas.

Capítulo 3

Algoritmos de factorização e primalidade usando curvas elípticas

3.1 Algoritmo de factorização

Vai-se descrever um método de factorização de um número natural n baseado em curvas elípticas, conhecido por *Método de Lenstra.*

Este método de factorização utilizando as curvas elípticas deve-se aos irmãos holandeses *Hendrik Lenstra* e *Arjen Lenstra* e é análogo ao *método* $p-1$ *de Pollard*, anteriormente introduzido pelo matemático inglês *John M. Pollard*, sendo que no primeiro usa-se o grupo dado por uma curva elíptica e no seguinte usa-se o grupo multiplicativo de um corpo finito (ver [16, p. 192]).

Sendo assim, vai, de uma forma muito breve, fazer-se uma pequena abordagem ao método $p-1$ de Pollard e, logo de seguida, vai ver-se o método de factorização de Lenstra.

Método $p-1$ de Pollard.

O *método* $p-1$ *de Pollard* é muito eficaz quando se quer factorizar um número natural n de que um número primo — desconhecido — p é divisor e tal que $p-1$ não tenha nenhum factor maior do que um certo limite B prefixado.

Passo 1 Escolhe-se um número inteiro k que é múltiplo de todos os inteiros menores do que o limite B — k pode ser $B!$ ou mínimo múltiplo comum entre os inteiros menores ou iguais ao limite B.

Passo 2 Escolhe-se um número natural a tal que $1 < a < n$.

Passo 3 Calcula-se o $mdc\,(a,n)$. Se o $mdc\,(a,n)$ for maior do que 1 estará obtido um divisor de n, senão passa-se para o passo seguinte.

Passo 4 Calcula-se $D = mdc\left(a^k - 1, n\right)$. Se $1 < D < n$ estará obtido um divisor de n. Se $D = 1$, voltar-se-á ao *Passo 1* e escolher-se-á um valor de k maior. Se $D = n$, voltar-se-á ao *Passo 2* e escolher-se-á um outro valor de a.

Nota 3.1 *Sabe-se que o conjunto dos elementos diferentes de zero dum corpo* $\mathbb{Z}_{/p}$ *forma um grupo* $\mathbb{Z}^*_{/p}$ *de ordem* $p-1$. *Sendo assim, se* p *for um número primo divisor de* n *(que se quer factorizar) tal que* $p-1$ *seja um produto de potências de números primos menores do que o limite* B, *então* $(p-1) \mid k$. *Assim sendo, para todo* $a \in \mathbb{Z}^*_{/p}$ *ter-se-á*

$$a^k \equiv 1 \pmod{p}\,.$$

Logo $p \mid (a^k - 1)$ *e, consequentemente,* $p \mid mdc\,(a^k - 1, n)$. *A única hipótese de não se obter um divisor próprio de* n *no* Passo 4 *é caso*

$$a^k \equiv 1 \pmod{n},$$

isto é,

$$n \mid (a^k - 1).$$

Isto mostra que o método $p-1$ *de Pollard funciona efectivamente.* O "ponto fraco" *deste método é que ele deixa de ser eficiente quando todo o divisor primo* p *de* n *for tal que* $p-1$ *seja divisível por números primos — ou potências de números primos — relativamente grandes, isto é, maiores do que o limite* B *prefixado.*

Exemplo 3.1 *Vai factorizar-se o número* $n = 540143$. *Escolhe-se*

$$B = 8,$$

$$k = mmc\,(1, 2, ..., 8) = 840$$

e

$$a = 2.$$

Tem-se

$$D = mdc\,(a^k - 1, n) = 421.$$

Isto dá a seguinte factorização,

$$n = 421 \times 1283.$$

Nota 3.2 *Se se quizer factorizar o número* $n = 491389$ *ter-se-á, necessariamente, de escolher um* B *que seja maior ou igual a* 191, *pois*

$$n = 383 \times 1283$$

e tem-se

$$383 - 1 = 2 \times 191$$

e

$$1283 - 1 = 2 \times 641.$$

Excepto para $a \equiv 0, \pm 1 \pmod{383}$, *todos os outros valores* a *têm ordens módulo* 383 *iguais a* 191 *ou a* 382. *Semelhantemente, excepto para* $a \equiv 0, \pm 1 \pmod{1283}$, *todos os outros valores de* a *têm ordens módulo* 1283 *iguais a* 641*ou a* 1282. *Por isso, a menos que* k *seja divisível por* 191 *— ou* 641 *— provavelmente ter-se-á*

$$mdc\,(a^k - 1, n) = 1$$

no Passo 4.

Posto isto, vai ver-se em que consiste o *método de Lenstra.*

Método de Lenstra

No método dos irmãos *Lenstra* substitui-se o grupo $\mathbb{Z}^*_{/p}$ pelo grupo $E_{/\mathbb{F}_p}$ — de uma curva elíptica E sobre o corpo $\mathbb{F}_p$ — e o número a por um ponto $P \in E_{/\mathbb{F}_p}$, onde p é um número primo divisor de n (que se quer factorizar). Semelhantemente ao método $p-1$ de Pollard, determina-se um número k, mas da seguinte forma:

O número k deve ser divisível por potências de números primos (menores ou iguais a um certo limite B — prefixado) menores que um certo limite C — prefixado, isto é,

$$k = \prod_{p \leq B} p^{\alpha_p}, \tag{3.1}$$

onde $\alpha_p = \dfrac{\log C}{\log p}$ é o maior expoente tal que $p^{\alpha_p} \leq C$.

Se a ordem $\#E_{/\mathbb{F}_p}$ do grupo $E_{/\mathbb{F}_p}$ dividir k ter-se-á

$$kP = \mathcal{O}.$$

Este facto, permite obter um factor de n — como se pode ver mais abaixo.

Nota 3.3 *Pode-se ver ainda que o método de Lenstra oferece uma vantagem — entre outras — em relação ao método $p-1$ de Pollard, pois, a ordem $\#E_{/\mathbb{F}_p}$ varia muito se se mudar os parâmetros da equação que define a curva elíptica E, dando mais hipóteses de se obter a condição*

$$\#E_{/\mathbb{F}_p} \mid k.$$

Antes de apresentar o teorema que serve de fundamento ao método, atente-se às seguintes observações:

- A equação
$$y^2 = x^3 + ax + b \pmod{n} \tag{3.2}$$
siginifica que os parâmetros a e b são inteiros módulo n. Neste caso $E_{/\mathbb{Z}_{/n}}$ representa o conjunto de pares (x, y), onde x, $y \in \mathbb{Z}_{/n}$, que satisfazem a equação 3.2; note-se ainda que se n for um número primo, $E_{/\mathbb{Z}_{/n}}$ será o conjunto dos pontos racionais de uma curva elíptica E definida sobre o corpo $\mathbb{F}_n$.

- Seja $P = (x, y)$. A notação $P \pmod{n}$ significa que as coordenadas x e y pertencem a $\mathbb{Z}_{/n}$.

Considera-se o seguinte teorema:

Teorema 3.1 *Seja E uma curva elíptica dada pela equação*

$$y^2 = x^3 + ax + b,$$

onde a, $b \in \mathbb{Z}$ e seja n um inteiro positivo tal que

$$mdc\left(4a^3 + 27b^2, n\right) = 1.$$

Sejam, também,

$$P_1 = (x_1, y_1), P_2 = (x_2, y_2) \in E, P_1 \neq -P_2,$$

cujas coordenadas têm denominadores primos com n. Então $P_1 + P_2$ tem coordenadas com denominadores primos com n se e só se não existir um primo p que divide n tal que

$$P_1 \pmod{p} + P_2 \pmod{p} = \mathcal{O},$$

para $P_1 \pmod{p}$,$P_2 \pmod{p}$ e $\mathcal{O} \in E_{/\mathbb{F}_p}$ (ver [16, p. 194]).

Dado um número composto n, pretende-se determinar um divisor d de n tal que $1 < d < n$. Começa-se por escolher uma curva elíptica E dada pela equação

$$y^2 = x^3 + bx + c,$$

$b, c \in \mathbb{Z}$ e um ponto $P \in E$.

Uma vez escolhido o par (E, P), escolhe-se um número k como na equação 3.1, por exemplo.

Vai-se determinar $kP \pmod{n}$. Esse cálculo não oferece grande problema, a não ser que $x_2 - x_1$ e $2y_1$ — das fórmulas da soma dos pontos numa curva elíptica — não tenham simétrico módulo n. Se o cálculo de $kP \pmod{n}$ não for possível, então, pelo teorema anterior, existirá um k_1, $k_1 \leq k$, tal que

$$k_1 P = \mathcal{O} \pmod{p}$$

para um p primo que divide n, isto é, k_1 é múltiplo da ordem de P em $E_{/\mathbb{F}_p}$. Na tentativa de calcular o inverso módulo n de um denominador divísível por p — usando o *algoritmo de Euclides* — obtém-se o máximo divisor comum entre esse denominador e o n. Esse máximo divisor comum poderá ser um divisor próprio de n — isto é, um divisor d tal que $1 < d < n$ — a não ser que seja o próprio n, isto é que o denominador seja um múltiplo de n — o que significaria que $k_1 P = \mathcal{O}$ em $E_{/\mathbb{F}_p}$ para todo p divisor de n.

Se a escolha (E, P) não for boa — isto é, se para cada p divisor de n o grupo $E_{/\mathbb{F}_p}$ tiver ordem divisível por um primo muito grande e, por conseguinte, não se tiver $kP = \mathcal{O}$ em $E_{/\mathbb{F}_p}$ para o valor de k determinado — escolher-se-á um novo par e começar-se-á tudo de novo.

Assim é muito provável que ao calcular $kP \pmod{n}$, para um k que seja múltiplo da ordem de P em $E_{/\mathbb{F}_p}$ para algum p divisor de n, se obtenha um divisor próprio de n.

Eis a seguir o algoritmo de Lenstra passo a passo.

Seja n um número inteiro ímpar e composto no qual se quer obter um factor.

Passo 1. Verifica-se se $mdc\,(n, 3) = 1$ — podendo deste modo utilizar-se a equação

$$y^2 = x^3 + bx + c \text{ —}$$

e se n não é da forma m^r, $r \in \mathbb{N}$ e $r \geq 2$.

Passo 2. Escolhe-se três números inteiros b, x_1 e y_1 entre 1 e n.

Passo 3. Calcula-se $c = y_1^2 - x_1^3 - bx_1 \pmod{n}$. Seja $E_{/\mathbb{Z}_{/n}}$ dada pela equação

$$y^2 = x^3 + bx + c$$

e seja $P = (x_1, y_1) \in E_{/\mathbb{Z}_{/n}}$.

Passo 4. Verifica-se se $mdc\,(4b^3 + 27c^2, n) = 1$ — garantindo que o polinómio

$$f(x) = x^3 + bx + c$$

tenha raízes distintas em $\mathbb{F}_p$ para todo p divisor de n. Se

$$1 < mdc\left(4b^3 + 27c^2, n\right) < n,$$

estará obtido um divisor próprio de n e se

$$mdc\left(4b^3 + 27c^2, n\right) = n$$

então escolher-se-á um outro valor de b.

Passo 5. Escolhe-se um valor de k como na equação 3.1, por exemplo, ou k pode ser o

$$mmc\,(1,2,3,4,...,K)$$

para um certo valor de K prefixado.

Passo 6. Calcula-se $kP \pmod n$. Se o cálculo não for possível então obter-se-á um divisor de n, que pode ser um divisor próprio de n ou o próprio n. Nesse último caso ir-se-á ao *Passo 5* e diminuir-se-á o valor de k. Se o cálculo de $kP \pmod n$ ocorrer sem sobressalto, ir-se-á ao *passo 3* e começar-se-á tudo de novo.

Exemplo 3.2 *Vai-se factorizar o número inteiro ímpar* $n = 493$.
Escolhe-se $P = (1,1)$ *e* $b = 1$, *por conseguinte* $c = -1$ *e tem-se*

$$E_{/\mathbb{Z}_{/n}}(0,1,-1) \text{ — dada pela equação } y^2 = x^3 + x - 1.$$

Escolhe-se $B = 3$ *e* $C = 34$, *assim sendo tem-se*

$$k = 2^5 \times 3^3 = 2^9 + 2^8 + 2^6 + 2^5.$$

Por ser $mdc(4b^3 + 27c^2, n) = mdc\,(31, 493) = 1$, *garante-se a existência do grupo* $E_{/\mathbb{F}_p}(0,1,-1)$ *para todo* p *primo divisor de* n.
Vai-se calcular

$$kP \pmod n = \left(2^9 P \pmod n + 2^8 P \pmod n + 2^6 P \pmod n + 2^5 P \pmod n\right) \pmod n \tag{3.3}$$

Calcula-se em primeiro lugar os produtos de P *pelos escalares* 2^i, *para* $i = 1, 2, ..., 9$. *Para os cálculos que se seguem vai-se utilizar o software PARI/GP (ver em [26]).*

$P_1 = 2P = (2, 490)$.
$P_2 = 2P_1 = 2^2 P = (480, 217)$.
$P_3 = 2P_2 = 2^3 P = (280, 292)$.
$P_4 = 2P_3 = 2^4 P = (410, 3)$.
$P_5 = 2P_4 = 2^5 P = (480, 276)$.
$P_6 = 2P_5 = 2^6 P = (280, 201)$.
$P_7 = 2P_6 = 2^7 P = (410, 490)$.
$P_8 = 2P_7 = 2^8 P = (480, 217)$.
$P_9 = 2P_8 = 2^9 P = (280, 292)$.

Tem-se então $kP \pmod n = \mathcal{O}$ — *aplicando o método descrito na equação 3.3* —, *consequentemente não se pode aplicar o Teorema 3.1.*
No entanto, se se calcular $kP \pmod n$ *da maneira natural,*

$$2P, 3P = 2P + P, ..., kP = (k-1)\,P + P, \tag{3.4}$$

obter-se-á os resultados que se seguem:

$2P = (2, 490)$.
$3P = 2P + P = (13, 47)$.
$4P = 3P + P = (480, 217)$.
$5P = 4P + P = (79, 146)$.
$6P = 5P + P = (7, 405)$.
$7P = 6P + P = (34, 242)$.
$8P = 7P + P = (280, 292)$.

$9P = 8P + P = (363, 459)$.
$10P = 9P + P = (331, 99)$.
$11P = 10P + P = (17, 302)$.
$12P = 11P + P = (126, 20)$.
$13P = 12P + P = (11, 313)$.
$14P = 13P + P = (429, 55)$.
$15P = 14P + P = (319, 357)$.
$16P = 15P + P = (410, 3)$.
$17P = 16P + P = (18, 152)$.

O cálculo de $18P = 17P + P$ *é impossível pois* 17 *— obtido aplicando as fórmulas da equação 2.6 — não tem simétrico módulo* $n = 493$. *Pelo Teorema 3.1 existe um número primo* p *divisor de* n *tal que*

$$(17P + P) \pmod{p} = \mathcal{O}$$

no grupo $E_{/\mathbb{F}_p}$ *da curva elíptica* E *sobre o corpo* $\mathbb{F}_p$.

Calcula-se $mdc\,(493, 17) = 17$ *— que é um número primo — e então* $(17P + P) \pmod{17} = \mathcal{O}$ *no grupo* $E_{/\mathbb{F}_{17}}$, *como já se tinha constatado; então* 17 *é um número primo divisor de* $n = 493$. *Tem-se então*

$$493 = 17 \times 29.$$

Nota 3.4 *Note-se que a escolha de* B, *de* C *e por conseguinte de* k, *dá necessariamente um divisor próprio de* n *aplicando o método de Lenstra. Pois, pelo teorema de Hasse se um divisor primo* p *de* n *for tal que* $p + 1 + 2\sqrt{p} < C$ *e a ordem de* $E_{/\mathbb{F}_p}$ *não for divisível por nenhum primo maior que* B, *então* k *será múltiplo dessa ordem e, consequentemente,* $kP = \mathcal{O}$ *em* $E_{/\mathbb{F}_p}$.

Tem-se então, $17 + 1 + 2\sqrt{17} < 34$ *— podendo eventualmente tomar-se* $C = 27$ *que é, também, maior do que* $17 + 1 + 2\sqrt{17}$ *— e*

$$\#E_{/\mathbb{F}_{17}}(0, 1, -1) = 18 = 2 \times 3^2.$$

Assim k *é múltiplo da ordem de* $E_{/\mathbb{F}_{17}}(0, 1, -1)$ *e, por conseguinte,* $kP = \mathcal{O}$ *no grupo dessa curva elíptica.*

Este exemplo deixa bem clara a importância da escolha de um método para o cálculo de kP, *pois nem todos os métodos permitem obter um divisor próprio de* n.

Exemplo 3.3 *Vai factorizar-se o número* $n = 491389$.

Escolhe-se $P = (1, 1)$ *e* $b = 1$ *— os mesmos do Exemplo 3.2 — por conseguinte* $c = -1$ *e tem-se*

$$E_{/\mathbb{Z}_{/n}}(0, 1, -1) \text{ — dada pela equação } y^2 = x^3 + x - 1.$$

Calcula-se

$$mdc\left(4b^3 + 27c^2, n\right) = mdc\,(31, 491389) = 1.$$

Isto garante que a equação

$$y^2 = x^3 + x - 1$$

define uma curva elíptica E *sobre* $\mathbb{F}_p$ *para todo divisor primo* p *de* n.

Escolhe-se $B = 11$ *e* $C = 16$, *assim sendo tem-se*

$$k = 2^4 \times 3^2 \times 5 \times 11 = 55440 = 2^{15} + 2^{14} + 2^{12} + 2^{11} + 2^7 + 2^4.$$

Vai calcular-se

$$kP \pmod{n}$$

utilizando o método na equação 3.3. Calcula-se em primeiro lugar os produtos $2^i P \pmod{n}$, *para* $i = 1, 2, 3, ..., 15$. *Tem-se então:*

$P_1 = 2P = (2, 491386)$.
$P_2 = 2P_1 = 2^2P = (477740, 52324)$.
$P_3 = 2P_2 = 2^3P = (385818, 265513)$.
$P_4 = 2P_3 = 2^4P = (342132, 330770)$.
$P_5 = 2P_4 = 2^5P = (147575, 318221)$.
$P_6 = 2P_5 = 2^6P = (217137, 5139)$.
$P_7 = 2P_6 = 2^7P = (99932, 356689)$.
$P_8 = 2P_7 = 2^8P = (166961, 317962)$.
$P_9 = 2P_8 = 2^9P = (401733, 165884)$.
$P_{10} = 2P_9 = 2^{10}P = (122361, 30380)$.
$P_{11} = 2P_{10} = 2^{11}P = (257065, 241811)$.
$P_{12} = 2P_{11} = 2^{12}P = (298561, 31040)$.
$P_{13} = 2P_{12} = 2^{13}P = (340966, 16874)$.
$P_{14} = 2P_{13} = 2^{14}P = (448956, 222249)$.
$P_{15} = 2P_{14} = 2^{15}P = (412520, 383112)$.

Posto isso, calcula-se $kP \pmod{n}$. *Tem-se:*

$Q = (2^{15}P \pmod{n} + 2^{14}P \pmod{n}) \pmod{n} = (261590, 132134)$.
$M = (Q + 2^{12}P \pmod{n}) \pmod{n} = (380317, 478023)$.
$N = (M + 2^{11}P \pmod{n}) \pmod{n} = (144538, 229277)$.
$R = (N + 2^7P \pmod{n}) \pmod{n} = (215115, 472388)$.

Não é possível determinar o ponto

$$kP \pmod{n} = \left(R + 2^4P \pmod{n}\right) \pmod{n},$$

pois o número 127017 *— obtido aplicando as fórmulas da equação 2.6 — não tem simétrico módulo* $n = 491389$. *Pelo Teorema 3.1, existe um número primo* p *divisor de* n *tal que*

$$\left(R + 2^4 \pmod{p}\right) \pmod{p} = \mathcal{O}$$

no grupo $E_{/\mathbb{F}_p}$ *da curva elíptica* E *sobre o corpo* $\mathbb{F}_p$.

Calcula-se $mdc(n, 127017) = 1283$ *— que é um número primo — e tem-se*

$$\left(R + 2^4 \pmod{1283}\right) \pmod{1283} = \mathcal{O} \tag{3.5}$$

no grupo $E_{/\mathbb{F}_{1283}}$; *então* 1283 *é um número primo divisor de* 491389. *Tem-se então*

$$491389 = 1283 \times 383.$$

Nota 3.5 *Nota-se que a escolha de* $B = 11$ *e* $C = 16$ *é boa — no sentido de permitir a factorização de* n *— pois*

$$\#E_{/\mathbb{F}_{1283}}(0, 1, -1) = 1283 + 1 - 52 = 1232 = 2^4 \times 7 \times 11,$$

o que significa que todos os divisores primos de $\#E_{/\mathbb{F}_{1283}}(0, 1, -1)$ *são menores ou iguais a* B *e* $\#E_{/\mathbb{F}_{1283}}(0, 1, -1)$ *divide* k *— justificando a equação 3.5 — para o* C *considerado.*

Exemplo 3.4 *Vai-se factorizar o número* $n = 99966867641$.

Escolhe-se $P = (1, 3)$, $b = 1$ *e por conseguinte,* $c = 7$. *Tem-se* $E_{/\mathbb{Z}_n}$ *definida pela equação*

$$y^2 = x^3 + x + 7.$$

Calcula-se

$$mdc\left(4b^3 + 27c^2, n\right) = mdc\left(1327, 99966867641\right) = 1.$$

Escolhe-se $B = 17$ *e* $C = 16$. *Tem-se*

$$k = 2^4 \times 3^2 \times 5 \times 7 \times 11 \times 13 \times 17 = 12252240$$

Vai-se calcular

$$kP \pmod{n}.$$

Esse cálculo leva a uma expressão indefinida, pois o número 977 *— obtido ao aplicar as fórmulas da equação 2.6 — não tem simétrico módulo* n.

Calcula-se o

$$mdc\left(977, n\right) = 102320233.$$

Tem-se então

$$n = 977 \times 102320233.$$

Como o número $n_1 = 10232033$ *não é primo, vai-se factorizá-lo.*

Escolhe-se $P = (1, 3)$, $b = 1$ *e por conseguinte,* $c = 7$. *Tem-se* $E_{/\mathbb{Z}_{n_1}}$ *definida pela equação*

$$y^2 = x^3 + x + 7.$$

Escolhe-se $B = 17$ *e* $C = 16$. *Tem-se*

$$k = 2^4 \times 3^2 \times 5 \times 13 = 9360.$$

Vai-se calcular

$$kP \pmod{n_1}.$$

Esse cálculo leva, novamente, a uma expressão indefinida, pois o número 4774599 *— obtido ao aplicar as fórmulas da equação 2.6 — não tem simétrico módulo* n_1.

Calcula-se

$$mdc\left(4774599, n_1\right) = 977.$$

Tem-se então,

$$n_1 = 977 \times 104729$$

e, por conseguinte,

$$n = 977^2 \times 104729.$$

3.2 Algoritmo de primalidade

Segundo [16, p. 187], o *método de estudo da primalidade aplicando as curvas elípticas, deve-se o S. Goldwasser, o J. Kilian e a A.O.L. Atkin* e é *análogo* ao *método de teste de primalidade de Pocklington* que se baseia no grupo $\mathbb{Z}_{/n}$.

Antes de apresentar o método de estudo de primalidade aplicando as curvas elípticas, vai-se ver em que consiste o *método de teste de primalidade de Pocklington.*

Teorema 3.2 *Seja n um número inteiro positivo.*
Supõe-se que existe um número primo p divisor de $n-1$ tal que

$$p > \sqrt{n} - 1.$$

Se existir um número inteiro a tal que

$$a^{n-1} \equiv 1 \pmod{n} \qquad e \qquad mdc\left(a^{\frac{n-1}{p}} - 1, n\right) = 1, \tag{3.6}$$

então n será um número primo (ver [16, p. 187]).

Nota 3.6 *Este método é uma boa "ferramenta" para o estudo de primalidade de um número natural n desde que se conheça um divisor primo $p > \sqrt{n} - 1$ de $n-1$.*
Note-se que o inteiro a que se escolhe deverá satisfazer sempre a condição

$$a^{n-1} \equiv 1 \pmod{n},$$

caso n seja primo, mas poderá não satisfazer a condição

$$mdc\left(a^{\frac{n-1}{p}} - 1, n\right) = 1.$$

No entanto, se a satisfazer, este método permitirá avaliar com certeza, se n será ou não um número primo.
A maior dificuldade deste método prende-se com a factorização do número $n-1$ — a ordem do subgrupo $\mathbb{Z}^*_{/n}$ do corpo $\mathbb{Z}_{/n}$, caso n seja um número primo. Como já se disse atrás, a factorização é uma tarefa muito difícil. A mesma dificuldade põe-se também no método de estudo da primalidade aplicando as curvas elípticas, contudo as curvas elípticas dispõem de uma grande vantagem pois, mudando os parâmetros das equações que as representam mudam-se significativamente as ordens dos respectivos grupos — como se verá mais adiante — e existe um algoritmo que permite determinar a ordem de um grupo $E_{/\mathbb{F}_n}$ caso n seja um número primo.

Exemplo 3.5 *Vai fazer-se um teste de primalidade ao número $n = 29$. Tem-se:*
O número 7 divide $n - 1 = 28$ e $7 > \sqrt{29} - 1$.
Escolhe-se $a = 2$ e tem-se:

$$2^{28} \equiv 1 \pmod{29}$$

e

$$mdc\left(2^4 - 1, 29\right) = 1,$$

logo, pelo Teorema 3.6, o número $n = 29$ é um número primo.

Vai-se ver agora o método de estudo da primalidade aplicando as curvas elípticas. Considera-se o seguinte teorema.

Teorema 3.3 *Seja n um número inteiro positivo, seja $E_{/\mathbb{Z}_{/n}}$ o conjunto de pontos dado pela equação*

$$y^2 = x^3 + bx + c$$

e seja m um número inteiro.

Supõe-se que existe um número primo p que divide m e que é maior do que

$$\left(\sqrt[4]{n}+1\right)^2.$$

Se existir um ponto $P \in E_{/\mathbb{Z}_{/n}}$ tal que:

$$mP = \mathcal{O} \qquad e \qquad \left(\frac{m}{p}\right) P \neq \mathcal{O}, \tag{3.7}$$

então n será um número primo (ver em [16, p. 188]).

Nota 3.7 *O número m referido no teorema anterior será a ordem de $E_{/\mathbb{F}_n}$ se n for primo. Por isso, nos estudos de primalidade utilizando as curvas elípticas parte-se do príncipio que n é um número primo — pois n já se passou por um teste probabilístico de primalidade — e toma-se*

$$m = \#E_{/\mathbb{F}_n}.$$

Note-se que o número m faz o papel de $n-1$ no método de Pocklington.

Eis a seguir o método de estudo da primalidade utilizando curvas elípticas passo a passo.

Passo 1. Escolhe-se três números inteiros b, x_1 e y_1 entre 1 e n e calcula-se

$$c = y_1^2 - x_1^3 - bx_1 \,(\text{mod } n).$$

Então $P = (x_1, y_1)$ é um elemento do conjunto $E_{/\mathbb{Z}_{/n}}$ dada pela equação

$$y^2 = x^3 + bx + c \,(\text{mod } n).$$

Passo 2. Determina-se o número m — número de pontos de $E_{/\mathbb{Z}_{/n}}$.

Passo 3. Escreve-se m na forma, $m = k \times p$, para $k \geq 2$ e $p > \left(\sqrt[4]{n}+1\right)^2$ provavelmente primo. Se não se puder fazer isso, escolher-se-á um outro triplo, b, x_1 e y_1, e começar-se-á tudo de novo.

Passo 4. Calcula-se mP e kP.

Passo 5. Se se obtiver uma expressão indefinida — quando se obtém um denominador que não tem simétrico módulo n — no cálculo de mP e kP, obter-se-á um factor não trivial de n, logo n é composto.

Passo 5. Se $mP \neq \mathcal{O}$ então n será composto — pois se n for primo, m será a ordem do grupo $E_{/\mathbb{F}_n}$ e a ordem de todo elemento $P \in E$ é um divisor de m, logo terá de ser $mP = \mathcal{O}$.

Passo 6. Se $mP = \mathcal{O}$ e $kP \neq \mathcal{O}$, pelo teorema anterior n será primo se p for primo.

O problema reduz-se ao estudo de primalidade de p que é menor ou igual a $\frac{n}{2}$.

Começa-se substituindo n por p. Assim, obtém-se um processo recursivo com t repetições de um teste de primalidade, onde $t \leq \log_2 n$.

Quando tudo estiver pronto, obter-se-á um número primo p_t e, por conseguinte, os números $p_{t-1}, p_{t-2}, ..., p_1 = p$ serão todos números primos e finalmente n será verdadeiramente um número primo.

Nota 3.8 *Em vez do* Passo 1 *e* Passo 2, *poder-se-ia determinar um número m no intervalo*

$$\left[n+1-\sqrt{n}, n+1+\sqrt{n}\right]$$

tal que

$$m = k \times p$$

para um primo

$$p > \left(\sqrt[4]{n}+1\right)^2$$

e de seguida definir-se-ia a equação da curva elíptica $E(\mathbb{F}_n)$ *e determinar-se-ia um ponto* $P \in E_{/\mathbb{F}_n}$ *satisfazendo a condição 3.7 do Teorema 3.2. Contudo, ter-se-ia de conhecer a distribuição de números primos no intervalo*

$$\left[n+1-\sqrt{n}, n+1+\sqrt{n}\right].$$

Note-se também que este método prende-se com a mesma dificuldade do método anterior, isto é, com a factorização do número m. A vantagem aqui, é que se pode variar a equação da curva elíptica E e por conseguinte obter diferentes valores de m e, assim sendo, poder-se-á obter um valor de m cuja factorização é mais fácil de se obter — tendo em conta os recursos computacionais existentes.

Segundo [6, p. 598], se for utilizada a teoria da multiplicação complexa *(ver [23, p. 311]) na construção de curva elíptica, o número m será determinado de uma forma mais eficiente, isto é com uma complexidade de tempo polinomial.*

Exemplo 3.6 *Vai fazer-se um teste de primalidade ao número* 29.

m é um número inteiro pertencente ao intervalo

$$\left[30-\sqrt{29}, 30+\sqrt{29}\right]$$

que pode ser decomposto num produto de um número primo p — maior ou igual a 13, *uma vez que*

$$11 < \left(\sqrt[4]{29}+1\right)^2 < 13 \ —$$

por um número k maior ou igual a 2, *isto é, o m pode ser igual a*

$$26 = 2 \times 13{,}39 = 3 \times 13{,}34 = 2 \times 17 \text{ ou } 38 = 2 \times 19.$$

Então, há que escolher $E_{/\mathbb{F}_{29}}$ *de tal forma que a sua ordem seja um desses números e depois determinar o ponto P pertencente ao grupo* $E_{/\mathbb{F}_{29}}$ *que satisfaça a condição 3.7 do Teorema 3.3.*

Por exemplo, a escolha das equações

$$y^2 = x^3 + x^2 - 1 \qquad e \qquad y^2 = x^3 + 2x^2 - 2$$

não servirá para o propósito uma vez que os grupos das curvas elípticas $E_{/\mathbb{F}_{29}}$, $E'_{/\mathbb{F}_{29}}$ *definidas por estas equações têm ordens iguais a* 28 *e* 27, *respectivamente. Escolhendo a equação*

$$y^2 = x^3 + 3x^2 - 3$$

obtém-se uma curva elíptica $E_{/\mathbb{F}_{29}}$ *de ordem*

$$38 = 2 \times 19.$$

Como 19 *é um número primo maior do que* 13 *falta apenas obter um ponto*

$$P \in E_{/\mathbb{F}_{29}}$$

cuja a ordem é diferente de 2, *isto é,*

$$2P \neq \mathcal{O}.$$

Como o polinómio

$$f(x) = x^3 + 2x^2 - 2$$

não tem raízes em $\mathbb{F}_{29}$, *então os pontos de* $E_{/\mathbb{F}_{29}}$ *têm ordens diferentes de* 2, *logo pode-se escolher qualquer ponto de* $E_{/\mathbb{F}_{29}}$ *para o ponto* P. *Nessas condições, segundo o Teorema 3.3, o número* 29 *é um número primo.*

Exemplo 3.7 *Vai-se fazer um teste de primalidade ao número* $n = 4999$.
m deve ser um número pertencente ao intervalo $[5000 - \sqrt{4999}, 5000 + \sqrt{4999}]$.
O número primo p deve ser maior ou igual a 89, *pois* $83 < \left(\sqrt[4]{4999} + 1\right)^2 < 89$.
Escolhe-se

$$b = 1,\ x_1 = 1 \ e\ y_1 = 2.$$

Tem-se então

$$c = 2^2 - 1^3 - 1 = 2$$

e, por conseguinte, a equação

$$y^2 = x^3 + x + 2$$

define a curva elíptica E *sobre* $\mathbb{F}_{4999}$ *caso* 4999 *seja um número primo. Note-se que sob esse pressuposto, o ponto*

$$P = (1, 2)$$

pertence ao grupo $E_{/\mathbb{F}_{4999}}$ *e a ordem* $\#E_{/\mathbb{F}_{4999}}$ *do grupo* $E_{/\mathbb{F}_{4999}}$ *é*

$$\#E_{/\mathbb{F}_{4999}} = 4984 = 2^3 \times 7 \times 89.$$

Assim existirá um número m — *igual a* 4984 — *e um número primo* p — *igual a* 89 — *maior que* $\left(\sqrt[4]{n} + 1\right)^2$ *que divide* m. *Então*

$$m = k \times p,$$

onde

$$k = 56 \ e\ p = 89.$$

Porém, apesar de se ter

$$mP = 4984P = \mathcal{O},$$

não se pode concluir nada acerca da primalidade de $n = 4999$, *pois*

$$\frac{m}{p}P = \frac{4984}{89}P = 56P = \mathcal{O},$$

No entanto, se se escolher uma curva elíptica E *definida pela equação*

$$y^2 = x^3 + x + 7$$

e o ponto

$$P = (1, 3)\,,$$

tem-se os seguinte resultados:

O número

$$m = \#E_{/\mathbb{F}_{4999}} = 4994;$$

Existe um número primo

$$p = 227$$

que divide m *e que é maior que* $\left(\sqrt[4]{4999} + 1\right)^2$.

Tem-se então

$$m = k \times 227,$$

onde $k = 22$,

$$mP = \mathcal{O} \qquad e \qquad kP = (302, 4056) \neq \mathcal{O}.$$

Logo, pelo Teorema 3.3, o número

$$n = 4999$$

é primo.

Nota 3.9 *Note-se que não se pode aplicar o método de Pocklington para testar a primalidade do número* $n = 4999$, *pois,*

$$n - 1 = 4998 = 2 \times 3 \times 7^2 \times 17$$

e, por conseguinte, não existe um número primo

$$p > \sqrt{4999} - 1$$

que divide $n - 1$.

Exemplo 3.8 *Vai fazer-se um teste de primalidade ao número* $n = 104729$ *— obtido na factorização do número* 99966867641 *no Exemplo 3.4.*

$m = 105060 = 2^2 3 \times 5 \times 17 \times 103$

Escolhe-se

$$P = (2, 4)\,,\ b = 3$$

e, por conseguinte, a curva E *sobre* $\mathbb{F}_n$ *definida pela equação*

$$y^2 = x^3 + 3x + 2.$$

Nesse caso, o número

$$m = 104214 = 2 \times 3 \times 11 \times 1579$$

é divisível por um primo $p = 1579$ *maior do que* $\left(\sqrt[4]{104729}+1\right)^2$.

Assim sendo, tem-se

$$mP = \mathcal{O} \qquad e \qquad 66P = (29371, 28521) = \mathcal{O}.$$

Logo, pelo Teorema 3.1, o número

$$n = 104729$$

é primo.

Nota 3.10 *Note-se, mais uma vez, que a escolha do ponto* P *e do parâmetro* b *deve ajustar-se às condições do Teorema 3.1. Por exemplo, a escolha de*

$$P = (2, 3)\,,\ b = 1$$

e, consequentemente, da curva E *sobre* $\mathbb{F}_n$ *definida pela equação*

$$y^2 = x^3 + x - 1,$$

não se ajusta àquelas condições, pois, para esse tem-se

$$m = 105078 = 2 \times 3 \times 83 \times 211,$$

e, como se constata, não existe nenhum número primo

$$p > \left(\sqrt[4]{n}+1\right)^2$$

que divide m.

Note-se, mais uma vez, que o método de Pocklington não é aplicável ao estudo de primalidade do número $n = 104729$, *pois não existe nenhum número primo*

$$p > \sqrt{n} - 1$$

que divide

$$n - 1 = 2^3 \times 13 \times 19 \times 53.$$

3.3 Prática da criptografia com curvas elípticas

Os problemas de factorização e de primalidade têm grande importância em criptografia e como se disse atrás, a segurança do sistema criptográfico RSA baseia-se na enorme dificuldade em factorizar um número inteiro muito grande. Nesse âmbito, os métodos de factorização e de primalidade aplicando as curvas elípticas têm uma grande importância. Sendo assim, vai-se apresentar um estudo dos programas *PARI* e *SAGE* no que tange aos métodos utilizados na factorização e no estudo de primalidade e ver, efectivamente, qual é a importância dos métodos das curvas elípticas para estes programas.

PARI e *SAGE* são softwares concebidos, fundamentalmente, para dar vazão às múltiplas necessidades advenientes de resoluções de problemas em Matemática, visando, essencialmente, a rapidez e a precisão nos cálculos. A razão por que se apresentam estes programas prende-se com a sua importâncias no que respeita às operações com curvas elípticas.

São utilizados por especialistas nos estudos da factorização e primalidade baseados em curvas elípticas.

Oferecem uma grande vantagem pelo facto de serem gratuitos. Sendo assim, podem ser considerados boas alternativas aos programas *MAGMA*, *MAPLE*, *MATHEMATICA* e *MATLAB* — programas estes que realizam também as operações relativas às curvas elípticas, mas cujos acessos pressupõem custos (elevados em alguns casos).

Para a análise do tempo gasto pelos dois programas na factorização e no estudo de primalidade vai-se utilizar um computador com as seguintes características:

Processador: Intel(R) Core(TM) Duo CPU T8300 @ 2.40GHz 2.40 GHZ;
Memória: 3,00 GB;
Tipo de Sistema: Sistema Operativo de 32 bits.

Antes de se avançar para a análise dos métodos utilizados por estes programas na factorização e na primalidade, vai-se em breves traços, apresentar as principais operações com curvas elípticas que intervêm na utilização desses programas.

O programa PARI

"PARI/GP is free software, covered by the GNU General Public License, and comes WITHOUT ANY WARRANTY WHATSOEVER."

Para este trabalho vai-se utilizar a seguinte versão do programa PARI:

GP/PARI CALCULATOR Version 2.3.2 (released)
i686 running cygwin (ix86 kernel) 32-bit version
compiled: Mar 28 2007, gcc-3.4.4 (cygming special, gdc 0.12, using dmd 0.125)
(readline v5.2 enabled, extended help not available)
Copyright (C) 2000-2006 The PARI Group

Toda linha de entrada é antecedida por *?* e todo resto é resultado produzido. Muitas vezes os resultados são antecedidos pelo simbolo *%n*, onde n é um número natural.

Para além disso, existe a função

?

cujo valor de entrada é uma função qualquer e o resultado a respectiva descrição. Ainda, a função *?* sem nenhum valor de entrada tem como resultado os tópicos de ajuda.

Assim sendo, tem-se:

? ?
Help topics: for a list of relevant subtopics, type ?n for n in
0: user-defined identifiers (variable, alias, function)
1: Standard monadic or dyadic OPERATORS
2: CONVERSIONS and similar elementary functions
3: TRANSCENDENTAL functions
4: NUMBER THEORETICAL functions
5: Functions related to ELLIPTIC CURVES
6: Functions related to general NUMBER FIELDS
7: POLYNOMIALS and power series
8: Vectors, matrices, LINEAR ALGEBRA and sets
9: SUMS, products, integrals and similar functions
10: GRAPHIC functions
11: PROGRAMMING under GP
12: The PARI community
Also:
? functionname (short on-line help)
?\ (keyboard shortcuts)
?. (member functions)

Sendo assim pode-se muito facilmente saber quais são as funções relativas às curvas elípticas utilizadas pelo PARI, basta utilizar o comando

?5

Apresenta-se de seguida o quadro das operações utilizadas no programa PARI:
A função,

ellinit([u,a,v,b,c]),

onde u, v, a, b e c são coeficientes dos termos que compõem a equação

$$y^2 + uxy + vy = x^3 + ax^2 + bx + c$$

duma curva elíptica $E(\mathbb{K})$, gera o vector:

$$[u, a, v, b, c, b_2, b_4, b_6, b_8, c_4, c_6, \Delta, j(E), (...)]$$

onde b_2, b_4, b_6, b_8, c_4 e c_6 são as constantes apresentadas na equação 2.2, Δ é o *discriminante* da curva elíptica, $j(E)$ é o *j-invariante* e o símbolo (...) representa os outros elementos apresentados no referido vector cuja importância para este trabalho não é assim tão relevante. Contudo, se se quiser saber as informações acerca dos elementos representados por (...), pode-se utilizar o comando

?ellinit

que o PARI aprsenta a discrição completa da função *ellinit*.

Exemplo 3.9 *A expressão*

$$\text{ellinit([Mod(0,11),Mod(0,11),Mod(0,11),Mod(1,11),Mod(3,11)]),} \tag{3.8}$$

gera um vector com as informações acerca da curva elíptica $E(\mathbb{F}_{11})$ *definida pela equação*

$$y^2 = x^3 + x + 3,$$

onde a função

Mod(x,y)

produz o inteiro x módulo y. *Tem-se:*

? e0=ellinit([Mod(0,11),Mod(0,11),Mod(0,11),Mod(1,11),Mod(3,11)])
%3 = [Mod(0, 11), Mod(0, 11), Mod(0, 11), Mod(1, 11), Mod(3, 11), Mod(0, 11), Mod(2, 11),Mod(1, 11), Mod(10, 11), Mod(7, 11), Mod(4, 11), Mod(8, 11), Mod(3, 11), 0, 0, 0,0, 0, 0]

A função

e0.disc

produz o valor de discriminante da curva elíptica *e0.* Tem-se, por exemplo o comando,

? e0.disc
%4 = Mod(8, 11)

Deve-se constatar que o valor imediatamente acima é igual ao 12º valor do vector gerado pela função da equação 3.8.

A função

ellisoncurve(e,p),

verifica se um ponto representado por p pertence a uma curva elíptica representada por e, produzindo o valor *1* em caso afirmativo e *0* em caso negativo.

Exemplo 3.10

? q=[Mod(4,11),Mod(-4,11)]
%6 = [Mod(4, 11), Mod(7, 11)]
? ellisoncurve(e0,q)
%7 = 1

isto é, o ponto

$$q = (4, -4)$$

pertence ao grupo

$$E(0,1,3)_{/\mathbb{F}_{11}}$$

da curva elíptica $E(\mathbb{F}_{11})$.

As funções

elladd(e,p,q)

ellsub(e,p,q)

e

ellpow(e,p,n)

são utilizadas para adicionar e subtrair dois pontos p e q e multiplicar um ponto p por um número natural n, respectivamente.

Exemplo 3.11

? q1=[Mod(4,11),Mod(-4,11)]
%8 = [Mod(4, 11), Mod(7, 11)]
? r1=[Mod(4,11),Mod(4,11)]
%9 = [Mod(4, 11), Mod(4, 11)]
? elladd(e0,r1,q1)
%10 = [0]

O vector [0] *representa o elemento neutro* $\mathcal{O}$ *do grupo*

$$E(0,1,3)_{/\mathbb{F}_{11}}.$$

Exemplo 3.12 *Se se quiser o simétrico de um ponto, pode-se utilizar a função* ellsub. *Tem-se, por exemplo:*

? ellsub(e0,[0],q1)
%11 = [Mod(4, 11), Mod(4, 11)]

Exemplo 3.13 *Se se quiser o produto de um número natural* n *por um ponto* P, *pode-se utilizar a função* ellpow. *Tem-se, por exemplo, o dobro do ponto*

$$P = (4, -4)$$

pertencente ao grupo

$$E(0,1,3)_{/\mathbb{F}_{11}}:$$

? P=[Mod(4,11),Mod(-4,11)]
%2 = [Mod(4, 11), Mod(7, 11)]
? ellpow(e0,P,2)
%3 = [Mod(7, 11), Mod(10, 11)]

A mesma operação pode ser realizada utilizando a função elladd, *isto é:*

? elladd(e0,P,P)
%4 = [Mod(7, 11), Mod(10, 11)]

Exemplo 3.14 *Se se quer determinar a ordem*

$$\#E(0,1,3)_{/\mathbb{F}_{11}} = 11 + 1 - t$$

procede-se da seguinte forma:

Determina-se-á o traço t *da curva elíptica* $E(\mathbb{F}_{11})$, *utilizando a função* ellap, *isto é:*

? ellap(e0,11)
%5 = -6

Calcula-se a ordem do grupo

$$E(0,1,3)_{/\mathbb{F}_{11}}$$

aplicando o Teorema de Helmut Hasse:

$$\#E(0,1,3)_{/\mathbb{F}_{11}} = 11 + 1 - t = 11 + 1 - (-6) = 18.$$

Se se quiser saber a ordem de um determinado ponto P pertencente a um grupo $E_{/\mathbb{K}}$ de uma curva elíptica $E(\mathbb{K})$, pode-se utilizar a função *ellorder*. Contudo, convém ressaltar que o programa PARI determina a ordem de um determinado ponto operando sobre o corpo $\mathbb{Q}$ — corpo dos números racionais. Entretanto, pode-se verificar se um número natural m é a ordem de um determinado ponto P pertencente a um determinado grupo $E_{/\mathbb{F}_q}$.

Verifica-se, assim, que a ordem do ponto

$$P = (4, -4),$$

pertencente ao grupo

$$E(0,1,3)_{/\mathbb{F}_{11}},$$

é 9, pois

```
? c=([Mod(0,11),Mod(0,11),Mod(0,11),Mod(1,11),Mod(3,11)])
%1 = [Mod(0, 11), Mod(0, 11), Mod(0, 11), Mod(1, 11), Mod(3, 11)]
? ellinit(c)
%2 = [Mod(0, 11), Mod(0, 11), Mod(0, 11), Mod(1, 11), Mod(3, 11), Mod(0, 11),
 Mod(2, 11), Mod(1, 11), Mod(10, 11), Mod(7, 11), Mod(4, 11), Mod(8, 11),
Mod(3, 11), 0, 0, 0, 0, 0, 0]
? for(k=1,9,print(ellpow(%2,[Mod(4,11),Mod(-4,11)],k)))
%3=
[Mod(4, 11), Mod(7, 11)]
[Mod(7, 11), Mod(10, 11)]
[Mod(1, 11), Mod(7, 11)]
[Mod(6, 11), Mod(4, 11)]
[Mod(6, 11), Mod(7, 11)]
[Mod(1, 11), Mod(4, 11)]
[Mod(7, 11), Mod(1, 11)]
[Mod(4, 11), Mod(4, 11)]
[0]
```

Posto isso, vai-se analisar a capacidade do programa PARI no que tange *ao estudo de primalidade e de factorização de um número natural n.*

Estudo da primalidade com o PARI

Para o teste de primalidade de um número natural n, o PARI incorpora a função

isprime,

cuja saída é *1* caso n seja primo e *0* no caso contrário. Para se obter informações acerca desta função, usa-se o seguinte comando

?isprime.

Sendo assim, tem-se:

```
? ?isprime
isprime(x,{flag=0}): true(1) if x is a (proven) prime number, false(0) if not.
If flag is 0 or omitted, use a combination of algorithms. If flag is 1, the
primality is certified bythe Pocklington-Lehmer Test. If flag is 2, the
primality is certified using the APRCL test
```

Isto é, há dois valores de entrada na função *isprime*, o número n para o qual se quer fazer o teste de primalidade e a entrada opcional *{flag=[...]}*:

1. Se *[...]=1*, o PARI usa o *teste de primalidade de Pocklinton-Lehmer* ;
2. Se *[...]=2*, o PARI usa o *teste de primalidade de Adleman-Pomerance-Rumely-Cohen-Lenstra (APRCL)* (ver em [6, p. 599]);
3. Se se omitir a entrada opcional *{flag=[...]}* ou se *[...]=0*, o PARI usa a combinação dos seguintes testes de primalidade: *teste de Baillie-PSW* (ver em [33] e [25]), *teste* $p-1$ *de Selfridge* (ver em [?]) e *teste de APRCL.*

Independentemente das opções de *flag* utilizada, quando se utiliza a função *isprime* leva-se muito tempo para se passar o certificado de primalidade a um número n, principalmente, quando n é um número com mais de mil dígitos. Por isso, antes de utilizar a referida função, utiliza-se a função

ispseudoprime,

que testa se um determinado número é um *pseudoprimo* — número, que pode ser primo ou composto, mas que passa numa sequência de testes para os quais a maioria dos números compostos não passa; por exemplo, tem-se os *números de Carmichael* (ver [11, p. 126]), de que 561 é um exemplo, isto é,

$$561 = 3 \times 11 \times 17$$

e no entanto obdece a condição do "*Pequeno Teorema de Fermat*" (ver [11, p. 125]), isto é,

$$a^{561} \equiv a \pmod{561},$$

para todo $a \in \mathbb{Z}$. Tem-se a descrição da função:

? ?ispseudoprime
ispseudoprime(x,{n}): true(1) if x is a strong pseudoprime, false(0) if not. If n is 0 or omitted, use BPSW test, otherwise use strong Rabin-Miller test for n randomly chosen bases.

Deve-se dizer ainda que no *teste de Baillie-Pomerance-Selfridge-Wagstaff (BPSW test)* aplicado número n são utilizados o *teste de Rabin-Miller* (ver [?]) para a base 2 seguido do *teste de Lucas* para a sequência $(P, -1)$, onde P é o menor inteiro positivo tal que

$$P^2 - 4$$

não seja um quadrado (mod x).

Exemplo 3.15 *Se se quer saber se o número*

$$n = 168888113180471771881$$

é um número primo ou não, utilizando o programa PARI, dever-se-á proceder duma das seguintes formas:

1.

```
? isprime(168888113180471771881,{flag=1})
%1=
[2 13 1]
[3 3 1]
[5 2 1]
[31531 2 1]
[803501 2 1]

? isprime(168888113180471771881,{flag=2})
%3 = 1
```

2.

```
? isprime(168888113180471771881)
%1 = 1
```

Na primeira opção o PARI emite alguns parâmetros obtidos no *teste de primalidade de Pocklinton-Lehmer*. Nas duas outras opções, ele emite o valor *1* que, como já se disse atrás, significa que o número em causa é um número primo.

Portanto, verifica-se que o programa PARI não usa o *método de estudo da primalidade aplicando as curvas elípticas.*

Estudo da factorização com o PARI

Para a factorização de um número natural n, o PARI incorpora a função

factorint.

Para se obter informações sobre a referida função, usa-se o comando

?factorint.

Tem-se:

> ? ?factorint
> *factorint(x,{flag=0}): factor the integer x. flag is optional, whose binary digits mean 1: avoid MPQS, 2: avoid first-stage ECM (may fall back on it later), 4: avoid Pollard-Brent Rho and Shanks SQUFOF, 8: skip final ECM (huge composites will be declared prime).*

Isto é, a função *factorint* faz uma combinação dos métodos, *SQUFOF de Shanks* (ver [29]), *ρ de Pollard-Brent* (ver [6, pp. 601-603] e [?]) , *MPQS* (ver [6, p. 611]) e o de *Lenstra-Montgomery — aplicação das curvas elípticas* — para obter os factores *pseudoprimos* de um número natural n.Tem-se:

factorint(n,{flag=[...]}),

onde n é o número que se quer factorizar, *{flag=[...]}* é opcional e *[...]* pode ser:

1. 1 (um) caso se queira evitar o MPQS;
2. 2 (dois) caso se queira evitar a primeira etapa do método de factorização aplicando as curvas elípticas;

3. 4 (quatro) se se queira evitar o método ρ de Pollard-Brent e o método de Shanks, SQUFOF;
4. 8 (oito) se se quiser evitar o método de factorização aplicando as curvas elípticas — faz-se esta opção, por exemplo, quando o número n já passou num teste de primalidade.

Exemplo 3.16 *Vai-se factorizar o número*

$$n = 534367789899878489279405948783469580359078894.$$

```
? factorint(534367789899878489279405948783469580359078894,{flag=1})
%2 =
[2 1]
[3 1]
[313 1]
[10061 1]
[22727 1]
[7368215844839 1]
[168888113180471771881 1]
```

O resultado aqui apresentado, tem 7 vectores diferentes, cada um composto por uma linha e duas colunas: o elemento da primeira coluna é um factor pseudoprimo de n *enquanto que o elemento da segunda, é o seu respectivo expoente, na decomposição do número n em factores. Sendo assim, tem-se:*

$$n = 2 \times 3 \times 313 \times 10061 \times 22727 \times 7368215844839 \times 168888113180471771881.$$

Posto isso, há que fazer o teste de primalidade a cada um dos factores. Para isso, usa-se a função isprime, *já vista anteriormente. Tem-se, então:*

```
? isprime(%2,{flag=2})
%3 =
[1 0]
[1 0]
[1 0]
[1 0]
[1 0]
[1 0]
[1 0]
```

Isto é, todos os factores obtidos na factorização do número n são realmente números primos — ilustrado pelo valor 1 *nas primeiras colunas das matrizes — enquanto que os expoentes não são números primos — ilustrado pelo valor* 0 *nas segundas colunas das matrizes.*

A função *factorint* que se utiliza para factorizar um número inteiro n, quando se utiliza o programa PARI, não é eficaz muito menos eficiente, uma vez que não devolve à primeira os factores primos do número n que se quer factorizar e pelo facto de se estar obrigado a fazer um teste de primalidade a cada um dos factores obtidos — o que consome muitos recursos (computacionais e outros).

Daí que, de acordo com os resultados aqui apresentados sobre o estudo de primalidade e o estudo de factorização de um número inteiro, não se considera o programa PARI um instrumento de grande utilidade na factorização de números inteiros e, consequentemente, de grande aplicação na criptografia actual.

O programa SAGE

SAGE é um programa matemático de código aberto e livremente disponível sob os termos da GNU General Public License. A execução atual é primeiramente devido a *William Stein.* É uma biblioteca do *Python* (ver em [27]) com intérprete personalizado. Escreve-se no *Python*, no *C++*, e no *C.*

O programa SAGE pode ser utilizado em diversas áreas em Matemática, nomeadamente: Álgebra Comutativa, Álgebra Linear, Teoria de Grupos, Cálculo Combinatório, Teoria de Números, etc. Também, este programa é muito útil para a prática da criptografia e, em particular, criptografia com curvas elípticas.

O SAGE fornece uma relação especial às diversas bibliotecas de fonte abertas importantes: o SINGULAR para a Álgebra Comutativa, o GAP para a Teoria dos Grupos, a biblioteca de MWRANK de John Cremona para as curvas elípticas, etc.

Para este trabalho vai-se utilizar a seguinte versão deste programa:

Sage version 3.1.4, Release Date: 2008-10-20

Apresenta-se algumas funções que podem ser utilizadas para operar com curvas elípticas:

A função

EllipticCurve

é utilizada de várias formas para a *obtenção de uma curva elíptica*:

1. *EllipticCurve([u,a,v,b,c])*: gera uma curva elíptica definida pela equação

$$y^2 + uxy + vy = x^3 + ax^2 + bx + c,$$

onde v, a, b e c são elementos do corpo que contém u. Se todos elementos u, v, a, b e c forem números inteiros, o SAGE gera uma curva elíptica sobre o corpo $\mathbb{Q}$;

2. *EllipticCurve([b,c])*: gera uma curva elíptica definida pela equação

$$y^2 = x^3 + bx + c,$$

isto é, os coeficientes u, v e a são nulos;

3. *EllipticCurve(R,[u,a,v,b,c])*: gera uma "curva elíptica" definida pela equação

$$y^2 + uxy + vy = x^3 + ax^2 + bx + c,$$

sobre o anel R;

4. *EllipticCurve(j)*: gera uma curva elíptica com j-invariante j;

5. *EllipticCurve(label)*: gera uma curva elíptica sobre o corpo $\mathbb{Q}$ para "*Cremona database*" (ver em [28]), onde *label* é uma expressão que obedece as regras de *Cremona database.*

Exemplo 3.17 *Para gerar uma curva elíptica definida pela equação*

$$y^2 = x^3 + x + 3$$

sobre o corpo $\mathbb{F}_{11}$ *procede-se da seguinte forma,*

```
sage:EllipticCurve(GF(11),[1,3])
Elliptic Curve defined by y^2 = x^3 + x + 3 over Finite Field of size 11
```

Exemplo 3.18 *Para gerar uma curva elíptica sobre um corpo* $\mathbb{F}_p$, $p > 10^9$, *definida pela equação*

$$y^2 = x^3 + bx + c,$$

onde b e c são elementos aleatórios do corpo $\mathbb{F}_p$, *procede-se da seguinte forma:*

```
sage:k=GF(next_prime(10^9)
sage:E=EllipticCurve(k,[k.random_element(),k.rando_element()])
sage:E
Elliptic Curve defined by y^2 = x^3 + 591317976*x + 255800667 over Finite Field
of size 1000000007
```

Note-se que a função

$$next\_prime(n)$$

gera o menor número primo superior a n.

Se se quiser saber a ordem de um ponto P, *escolhido aleatoriamente, do grupo* $E_{/\mathbb{F}_p}$, *procede-se da seguinte forma:*

$$sage{:}P=E.random\_element()$$

e o SAGE escolhe aleatoriamente o ponto P;

$$sage{:}P$$

obtém-se a coordenada do ponto P; *para este caso*

$$P = (361725399, 448295296)\,.$$

Com a seguinte linha de comando,

$$sage{:}P.order()$$

obtém-se a ordem do ponto P, *isto é, o valor* 500030866.

Se se quer saber o cardinal do grupo $E_{/\mathbb{F}_p}$, *procede-se da seguinte forma:*

$$sage{:}E.cardinality()$$

o resultado é 1000061732.

Adiciona-se dois pontos Q *e* R *quaisquer de* $E_{/\mathbb{F}_p}$, *procedendo da seguinte forma:*

$$sage{:}\ R=E.random\_element();Q=E.random\_element()$$

para se obter aleatoriamente os pontos $R, Q \in E_{/\mathbb{F}_p}$. *O ponto-e-vírgula (;) separa dois linhas de código;*

$$sage{:}R;Q$$

para se obter as coordenadas dos pontos Q e R escolhidos, isto é,

$$Q = (316420461, 398988046)\,,\ R = (347719103, 326570196)\,.$$

sage:R+Q

para se obter a soma dos pontos Q e R, isto é, o ponto de $E_{/\mathbb{F}_p}$ com a seguinte coordenada

$$R + Q = (45635224, 200467534)\,.$$

O produto de um ponto de $E_{/\mathbb{F}_p}$ por um escalar obtém-se com a seguinte linha de comando, por exemplo:

*sage:P.order()*P*

é igual ao ponto $\mathcal{O}$;

*sage:10004567*R*

é o ponto de $E_{/\mathbb{F}_p}$ com a seguinte coordenada

$$10004567 \times R = (697715765, 619216598)\,.$$

Deve-se salientar que o programa SAGE dispõe do comando

time

que determina o tempo decorrido e o tempo gasto pelo CPU na execução de uma determinada operação e apresenta o respectivo resultado. Este comando deve ser posicionado no início de uma linha de input.

Exemplo 3.19

*sage: time next_prime(10^25)*next_time(10^55)*
CPU times: user 0.07 s, sys 0.00 s, total 0.07
wall time: 0.07 s
10000000000000000000000013000000000000000000000000000\
21000000000000000000000273.

O "wall time" — que representa o tempo decorrido desde o início até o fim da operação — pode ser maior do que "CPU times" desde que outro programa esteja a correr para além do programa SAGE.

Vai analisar-se agora o programa SAGE no que tange à factorização e ao estudo de primalidade.

Estudo da factorização com o SAGE

Para a factorização de um número inteiro n, o programa SAGE dispõe da função

*factor(n, proof=none, int_=false, algorithm='pari', verbose=0, **kwds)*

onde,

- n: é o número que se quer factorizar;
- *proof*: é um valor boleano (*true* or *false*) ou *none* — por defeito *none*;

- *int_*: é um valor boleano — por defeito false;
- *algorithm*: é um string e assume os valores: *'pari'*, *'kash'* e *'magma'* (estes dois últimos devem estar instalados no computador);
- *verbose*: é um valor inteiro — por defeito 0 (zero) — e permite activar o "debug" caso se esteja a utilizar a biblioteca do PARI.

Salienta-se que a função *factor* utiliza a biblioteca do programa PARI para factorizar. Se se quer utilizar as curvas elípticas na factorização deve-se utilizar a função

$$ecm.factor$$

que dá acesso ao algoritmo *GMP ECM* — algoritmo optimizado aplicando as curvas elípticas. Recorda-se que o programa PARI também usa o algoritmo das curvas elípticas, só que esse é menos eficaz e eficiente se comparado com *GMP ECM*.

Exemplo 3.20 *Vai-se factorizar o número*

$$n = 534367789899878489279405948783469580359078894$$

— do ***Exemplo 3.3*** *— e o tempo de execução:*

1. *Aplicando o método GMP ECM, obtém-se*

   sage:time ecm.factor(534367789899878489279405948783469580359078894)
   CPU time: user 0.00 s, sys 0.07 s, total 0.07 s
   wall time: 2.01 s
   [2, 3, 313, 10061, 22727, 7368215844839, 168888113180471771881]

   onde cada elemento da lista anterior é um factor primo do número n.

2. *Utilizando a biblioteca do PARI, obtém-se*

   sage: time factor(534367789899878489279405948783469580359078894)
   CPU time: user 0.09 s, sys 0.01 s, total 0.10 s
   wall time: 0.10 s
   [2, 3, 313, 10061, 22727, 7368215844839, 168888113180471771881].

Exemplo 3.21 *Vai-se factorizar um número natural n igual ao produto do menor número primo*

$$p > 2^{41}$$

e o menor número primo

$$q > 2^{301};$$

tem-se:

*sage: n=next_prime(2^41)*next_prime(2^301);n*
895897896882121678495183137093227317966068840963537 74\
3285791696138951906898578037171073021459395982 2041
sage: len(n.str(10))
103
sage: len(n.str(2))
343

sage: time f=ecm.factor(n)
CPU times: user: 0.00 s, sys: 0.01 s, total: 0.01 s
Wall time: 0.68 s
sage:f
[2199023255579, 4074071952668972172536891376818756322 1029\
3678733187250127228089870876259952667341236679 4779]

sage: g=time factor(n)
CPU times: user: 7.05 s, sys: 0.01 s, total: 7.06 s
Wall time: 7.06 s.

Note-se que a função

len(x.str(y))

determina o número de dígitos do número x *na base* y.

Atente-se aos dois exemplos seguintes:

Exemplo 3.22

sage: p=next_prime(2^101);p
25353012004564588029934064108 33
sage: q=next_prime(p);q
25353012004564588029934064109 01
sage: n=p*q;s=n.str(10);len(s)
61
sage: time ecm.factor(n)
CPU times: user: 0.00 s, sys: 0.19 s, total: 0.19 s
Wall time: 91.39 s
[25353012004564588029934064108 33, 25353012004564588029934064108 33901].

Isto é, para factorizar o número

$$n = 25353012004564588029934064108 33 \times 25353012004564588029934064109 01$$

— número com sessenta e um dígitos decimais, resultado do produto de dois números primos consecutivos de trinta e um dígitos decimais cada — com o método das curvas elípticas optimizado levou-se pouco mais do que um minuto e meio, enquanto que no Exemplo 3.21 levou-se menos que um minuto e dez segundos.

Exemplo 3.23

sage: r=next_prime(2^102);r;len(r.str(10))
5070602400912917605986812821771
31
sage: o=next_prime(r);o;len(o.str(10))
5070602400912917605986812821829
31
*sage: m=r*o;len(m.str(10))*
62
sage: time ecm.factor(m)
CPU times: user: 0.01 s, sys: 0.46 s, total: 0.47 s
Wall time: 1426.48 s
[5070602400912917605986812821771, 5070602400912917605986812821829].

Isto é, para factorizar o número

$$m = 5070602400912917605986812821771 \times 5070602400912917605986812821829$$

— número com sessenta e dois dígitos decimais e resultado do produto de dois números primos consecutivo de trinta e um dígitos decimais cada — levou-se pouco mais que vinte e três minutos, utilizando o método das curvas elípticas optimizado do SAGE.

Estes dois exemplos traduzem aquilo que se considera a grande desvantagem do método das curvas elípticas na factorização de um número natural n, isto é, desde que n seja produto de dois números primos, p e q, muito próximos um do outro — fazendo com que a diferença entre $\sqrt{n}$ e o menor deles, diga-se p, seja o menor possível — o método torna-se pouco eficiente, pois o k da *equação 3.1* deve ser divisível por p ou por q, o que o torna um valor muito grande — recorda-se que o método de factorização aplicando as curvas elípticas é tanto mais eficiente quanto maior for a diferença entre $\sqrt{n}$ e o menor dos factores primos de n.

Estudo da primalidade com o SAGE

Para o estudo de primalidade de um número inteiro positivo n o programa SAGE dispõe da função

is_prime(n, flag=0),

cuja saída é *True*, caso n seja primo e *False*, caso contrário. A opcional *flag* assume valores inteiros conforme o que se segue:

- *flag=0* — por defeito: permite a combinação dos métodos no estudo da primalidade;
- *flag=1*: permite a certificação de primalidade utilizando o método de *Pocklington-Lehmer*;
- *flag=2*: permite a certificação de primalidade utilizando o método *APRCL*.

Note-se que o programa SAGE *não usa o método de estudo de primalidade utilizando o método das curvas elípticas.*

Exemplo 3.24 *Vai-se passar um certificado de primalidade aos dois factores primos do número n do exemplo anterior:*

sage: f[-1]
407407195266897217253689137681875632210293678733187250 1\
272280898708762599526673412366794779
sage:f[-2]
2199023255579
sage: is_prime(f[-1])
True
sage: is_prime(f[-2])
True.

Note-se que sendo L = [L0, L1, L2, ..., Ln] uma lista com n + 1 elementos o SAGE permite destacar cada elemento da lista L usando o seguinte comando:

- L[0] = L[-n-1]*: é o primeiro elemento da lista;*
- L[1]=L[-n]*: é o segundo elemento da lista;*
- ....
- L[n] = L[-1]*: é o último elemento da lista.*

Nota 3.11 *Um* primo de Mersenne *é um número primo do tipo* $2^n - 1$*, onde n é um número natural. O 44º primo de Mersenne é*

$$p = 2^{32582657} - 1.$$

De acordo com o que se segue o número p tem 9 808 358 *dígitos decimais e* 32 582 657 *dígitos binários. Tem-se:*

sage: p=2^32582657-1
sage: len(p.str(10))
9808358
sage: len(p.str(2))
32582657

Apresenta-se os primeiros 20 dígitos e os últimos 50 dígitos do número primo p, conforme o que se segue:

sage: time p.str(10)[:20]
CPU times: user: 80.41 s, sys: 4.96 s, total: 85.37 s
wall time: 85.70 s
11601953396142409686
sage: time p.str(10)[-50:]
CPU times: user: 69.54 s, sys: 0.64 s, total: 70.18 s
wall time: 70.18 s
33212445737104635692000092659011752880154053967871.

Os algoritmos utilizados nos programas SAGE e PARI não permitem passar certificado de primalidade ao número p, visto que este é um número muito grande. Note-se que o método de estudo de primalidade utilizando as curvas elípticas pode ser utilizado para passar certificado de primalidade ao número p.

Capítulo 4

Criptografia com curvas elípticas: âmbito e limitações

A importância das curvas elípticas na criptografia pode ser encarada de uma das seguintes formas:

1. Pela sua importância na factorização;
2. Pela sua importância no estudo da primalidade;
3. Como segurança num sistema criptográfico.

O *algoritmo para a troca de chaves* desenvolvido por *Diffie-Hellman* e o *criptosistema de chave pública* desenvolvido por *El Gamal* — a maioria dos sistemas criptográficos segue o módulo de *El Gamal*, que vem na linha de pensamento do *Diffie-Helman* — baseiam-se no pressuposto de que determinados problemas matemáticos são de difícil resolução, como é o caso do *problema do logaritmo discreto no grupo* $\mathbb{F}_p^*$.

Recorde-se que o *sistema RSA* se baseia na dificuldade em se resolver, em princípio, a seguinte equação na variável x:

$$x^e \equiv c \pmod{N},$$

onde e, c, N são conhecidos, embora os valores de p e q na factorização $N = p \times q$, sejam desconhecidos. Por outras palavras, a segurança do *sistema RSA* baseia-se na dificuldade em determinar raízes e-ésimas *modulo* N. Contudo, sabe-se também que a equação imediatamente acima pode ser resolvida desde que se conheça a factorização do número N , isto é, a segurança do *sitema RSA* baseia-se também na hipotética *dificuldade em se factorizar*. Então, para se obter um sistema cada vez mais seguro, é necessário determinar números cada vez mais difíceis de factorizar e sendo assim, põe-se o problema de obter números primos p e q cada vez maiores de forma que $N = p \times q$ seja muito difícil de factorizar.

No capítulo da *factorização* e da *primalidade*, o *método das curvas elípticas* tem um papel muito importante, visto como generalização dos métodos de factorização e de estudo de primalidade utilizando o grupo multiplicativo $\mathbb{Z}_{/n}^{\times}$ (*Método* $p-1$ *de Pollard* e o *método de teste de primalidade de Pocklington*), onde n é um número que se quer factorizar ou de que se quer passar o certificado de primalidade. Neste aspecto, a grande vantagem do método das curvas elípticas prende-se com o facto de existirem várias curvas elípticas *modulo* n e, consequentemente, de diferentes ordens para o correspondente grupo; sendo

assim, quando uma curva não funciona para o propósito em vista utiliza-se uma outra curva — como já se viu na Secção 3.1.

Contudo, para o propósito da criptografia, hoje em dia usa-se valores de $N = p \times q$, onde p e q são números primos que têm no mínimo setenta e cinco dígitos decimais muito *próximos um do outro.* Neste caso o método de factorização desenvolvido por *Pomerance*, "*quadratic sieve*" (ver [16, p. 160]) supera o método das curvas elípticas pois: segundo [11, pp. 307,308] o tempo de execução do *método das "curvas elípticas"* na factorização de um número N depende do tamanho do seu menor divisor primo p (como já se tinha visto anteriormente) e é da ordem de

$$O\left(e^{\sqrt{2(\log p)(\log(\log p))}}\right)$$

e o do *método do "quadratic sieve"* depende do tamanho de N e é da ordem de

$$O\left(e^{\sqrt{(\log N)(\log(\log N))}}\right).$$

Pode concluir-se daqui que sendo p e q valores muito próximos os tempos de execução de um e do outro se aproximam, mas o *método do "quadratic sieve"* supera o das *"curvas elípticas"* porque, as suas etapas são mais rápidas que no *método das "curvas elípticas"*. Contudo o *método das "curvas elípticas"* não deixa de ser um excelente método de factorização de um número N muito grande, principalmente quando a diferença entre um dos factores primos de N para $\sqrt{N}$ é grande, pois o seu tempo de execução depende do menor divisor primo de N.

É de salientar ainda que o método *"number field sieve"* é conhecido como o melhor método para factorizar um número $N = p \times q$, onde p e q são números primos aproximadamente iguais (ver [11, p. 158]).

No que tange ao estudo da *primalidade*, note-se que o método das curvas elípticas é aplicado eficientemente para passar certificado de primalidade a um número com mais de mil casas decimais. A parte crucial do algoritmo é obter uma curva elíptica que obedeça às condições do *Teorema 3.7*, o que pode ser conseguido com a *teoria da multiplicação complexa* (ver [23, p. 197]).

Juntamente com o *método APRCL*, é considerado um dos melhores métodos para passar certificado de primalidade a um número N muito grande (ver [6, p. 597]). Cite-se [30]:

"ECPP is the fastest known general-purpose primality testing algorithm. ECPP has a running time of $O\left((\ln N)^4\right)$" — aqui N é o número a que se quer passar o teste de primalidade.

No que diz respeito à *segurança num sistema criptográfico* o que se procura são ainda *problemas matemáticos de difícil resolução* para servir de *funções de uma via.*

Segundo *Neal Koblitz*, há dois aspectos importantes que o levaram a propor a implementação do "grupo das curvas elípticas" na criptografia (ver [15]):

1. *A grande flexibilidade na escolha do grupo*, isto é, para cada número primo p existe um e um só grupo multiplicativo $\mathbb{F}_p^*$, enquanto que há vários grupos $E_{/\mathbb{F}_p}$;

2. *A dificuldade em se resolver o problema do logaritmo discreto* num grupo originado em curvas elípticas.

De facto, a resolução do *problema do logaritmo discreto baseado num grupo de curvas elípticas - ECDLP* ou PLDCE - é mais difícil que a resolução do *problema do logaritmo*

discreto em $\mathbb{F}_p^*$, que como já se disse é originalmente a base da segurança do sistema criptográfico RSA, conforme se pode ler em[11, p. 296]:

"*The principal reason that elliptic curves are used in cryptography is the fact that there are no index calculus algorithms known for ECDLP*".

O método baseado no "*index calculus*" (ver [23, p. 144]), aplicado à resolução do problema do logaritmo discreto em $\mathbb{F}_p^*$ proporciona um tempo de execução subexponencial, o que não acontece para o caso PLDCE. Cite-se [11, p. 296]:

"*The fastest known algorithm to solve ECDLP in* $E_{/\mathbb{F}_p}$ *take approximately* $\sqrt{p}$ *steps*".

Os métodos que permitem resolver o PLDCE com mais eficiência foram apresentados na *Secção 2.3*, no entanto desde que a curva seja escolhida convenientemente (no sentido de evitar determinadas curvas vulneráveis aos ataques, como são os casos das curvas elípticas supersingulares) a resolução do PLDCE torna-se cada vez mais difícil.

Há uma relação directa entre a segurança de um criptosistema assimétrico e o comprimento da chave privada utilizada. Em geral quanto maior for a chave tanto mais seguro se tornará um sistema criptográfico, mas em contrapartida consumirá mais recursos computacionais (memória, processador, etc). Neste particular, estudos apontam, para que os criptosistemas baseados em curvas elípticas oferecem vantagens em relação ao sistema RSA. Passo a citar [24]:

" ... *Recommended RSA key size for most applications is 2048 bits. For equivalent security using ECC, you need a key of only 224 bits.*"

"*The smaller ECC keys mean the cryptographic operations that must be performed by the communicating devices can be squeezed into considerably smaller hardware, that software applications may complete cryptographic operations with fewer processor cycles, and operations can be performed that much faster, while still guaranteeing equivalent security.*
"

Isto é, para o mesmo nível de segurança, o sistema RSA, baseado num grupo $\mathbb{F}_p^*$ terá de utilizar uma chave muito maior do que o mesmo sistema baseado em curvas elípticas.

A possibilidade de utilizar chaves de pequenos comprimentos e no entanto garantir a segurança dos criptosistemas, permite a aplicação dos criptosistemas baseados em curvas elípticas em pequenos aparelhos de comunicação, com menores ciclos de processamento e com menor espaço de tempo na execução das operações (ver [24]). Estes aspectos permitem, como é óbvio, menor aquecimento dos aparelhos, menos consumo de energia, menos consumo de memória e os programas são executados com mais rapidez.

Contudo, a *computação quântica* constitui uma grande ameaça aos sistemas criptográficos actuais, uma vez que o *algoritmo de Shor* usado em computação quântica para a factorização e resolução do problema do logaritmo discreto *tem tempo polinomial de execução* (ver[11, p. 483]); não existe ainda, porém, nenhum *hardware* — conhecido — adequado para permitir o uso de um *software* baseado no algoritmo de Shor.

À parte isto, existem problemas matemáticos muito mais difíceis do que o PLDCE, que se forem bem implementados num sistema criptográfico, permitirão ainda maior segurança do que a que o PLDCE proporciona a um criptosistema. São exemplos disso fundamentais problemas computacionais associados a um "lattice" (ver [11, p. 383)]):

- "*The shortest vector problem (SVP)*" — consiste em encontrar o menor vector diferente de zero associado a um "lattice";
- "*The closest vector problem (CVP)*" — dado um vector $w \in \mathbb{R}^n$ com $w \notin L$, onde L representa um "lattice", consiste em encontrar um vector $v \in L$ tal que $\|w - v\|$ seja o menor possível.

Enfim, os *grupos de classes* associados a um corpo de números, intervindo na *teoria dos corpos de classes* (ver [6] p. 547) poderão talvez vir a ser uma alternativa, vantajosa em certos casos, aos grupos baseados em curvas elípticas, mas pelo menos por agora não é o que sucede; note-se embora, que recorrendo aos grupos de classes não existe, tal como já sucedia no caso das curvas elípticas, um algoritmo com tempo de execução subexponencial para resolver o problema do logaritmo discreto (ver [1]).

Conclusão

O estudo do tema "*Criptosistemas baseados em curvas elípticas: âmbito e limitações*" permitiu constatar-se os seguintes aspectos:

Os criptosistemas baseados em curvas elípticas continuam a ser boas opções em criptosistemas de chaves públicas tendo em conta a dificuldade em se resolver o problema do logaritmo discreto em curvas elípticas, PLDCE. Nesse sentido, o melhor que se conseguiu até os dias de hoje é um algoritmo com tempo de execução exponencial para resolver o PLDCE.

O ponto crucial de criptosistemas baseados em curvas elípticas é saber se vai ou não encontrar-se um algoritmo que permita a resolução do PLDCE num tempo subexponencial.

No que diz respeito à sua eficiência, passo a citar [24]:

"*...if you're trying to make your devices smaller—and if you need to do asymmetric cryptography, you need ECC. If you're trying to make them run longer on the same battery, and produce less heat, and you need asymmetric cryptography, you need ECC. And if you want an asymmetric cryptosystem that scales for the future, you want ECC. And if you just want the most elegant, most efficient asymmetric cryptosystem going, you want ECC.*"

Deve-se dizer que o método de factorização aplicando as curvas elípticas continua a ser uma boa ferramenta para factorizar números inteiros muito grandes, embora a sua eficiência diminua consideravelmente quando se factoriza números inteiros da forma $N = p \times q$, onde p e q são números primos aproximadamente iguais, pois o número k da *equação 3.1* dever ser divisível por um dos divisores primos p de N, pelo que, se N for muito grande e p e q estiverem muito próximos um do outro e consequentemente de $\sqrt{N}$, o valor de k irá aumentar consideravelmente e, sendo assim, mais tempo e mais recursos computacionais serão necessários para a obtenção dos factores de N.

No que concerne o estudo da primalidade deve-se dizer que o método das curvas elípticas é um dos melhores que são utilizados para passar certificado de primalidade a um número N qualquer. A maior dificuldade que se encontra na aplicação desse método, é a obtenção do valor de m que satisfaça as condições do *Teorema 3.7*. Uma forma de ultrapassar essa dificuldade é escolher aleatoriamente uma curva elíptica $E\left(\mathbb{Z}_{/n}\right)$ e determinar a respectiva ordem até se obter um valor de m desejado, mas isso pode levar muito tempo; outra forma é a utilização da teoria da multiplicação complexa para obter a equação da curva $E\left(\mathbb{Z}_{/n}\right)$ de tal modo que a ordem seja um número m adequado.

Bibliografia

[1] BAIER, Harald [*et al.*]. *Stork Cryptography Workshop.*

[2] BARBOSA, Júlio César. *Criptografia de Chave Pública baseada em Curvas Elípticas.* Monografia final de curso de mestrado em redes. Rio de Janeiro: COPPE/UFRJ, Fev, 2003.

[3] BECKER, Anja. *Methods of Fault Analysis Attacks on Elliptics Curve Cryptosystems.* Diploma Thesis, Department of Computer Science - Darmstadt University of Technology, Darmstadt, September 2006.

[4] BLAKE, Ian F.; SEROUSSI, Gadiel & SMART, Nigel. *Elliptic Curves in Cryptography.* London Mathematical Society Lecture Note Series; 265. Cambridge: University Press, 1999.

[5] BRESSOUD, David M.. *Factorization and Primality Testing.* New York: Springer-Verlag, 1989.

[6] COHEN, Henri [*et al.*]. *Handbook of Elliptic and Hyperelliptic Curve Cryptography.* Discrete Mathematics and Its Applications. Boca Raton: Chapman & Hall/CRC Taylor & Francis Group, 2006.

[7] CRANDALL, Richard and POMERANCE, Carl. *Prime Numbers*: A Computational Perspective. New York: Springer-Verlag, 2001.

[8] DIFFIE, Whitfield and HELLMAN, Martin. *New directions in cryptography.* Whitfield Diffie and Martin E. Hellman

[9] FREY, Gerard & LANGE, Tanja. *Mathematical Background of Public Key Crypto graphy.* Séminaires & Congrés 11, p. 41-73, 2005.

[10] HANKERSON, Darrel; MENEZES, Alfred e VANSTONE, Scott. *Guide to Elliptic Curve Cryptography.* New York: Springer - Verlag, 2003.

[11] HOFFSTEIN, J.; PIPHER J. and SILVERMAN J. H. *An Introduction to Mathematical Cryptography.* Undergraduate Texte in Mathematics. New York: Springer Science+Business Media,LLC, 2008.

[12] KNOOP, Sarah. *Supersingular Curves and the Weil Pairing in Elliptic Curve Cryptography.* Dezembro 04.

[13] JOYNER, David; WILLIAM, Stein; et al. *SAGE Tutorial. Janeiro de 2008.*

[14] KOBLITZ, Neal; MENEZES, Alfred e VANSTONE, Scott. *Designs, Code e Cryptography.* Boston: Kluwer Academic Publishers, 19, 2000.

[15] KOBLITZ, Neal. *Algebraic Aspects of Cryptolography.* Algorithms and Computation in Mathematics Vol. 3, Berlin Heidelberg: Springer-Verlag, 1998.

[16] KOBLITZ, Neal. *A Course in Number Theory and Cryptography.* Graduate Texts in Matthematics. 2nd. ed., New York: Springer - Verlag, 1994.

[17] MENEZES, Alfred. *Evaluation of Security Level of Cryptography*: The Elliptic Curve Discrete Logarithm Problem (ECDLP). University of Waterloo, December 14, 2001.

[18] MENEZES, Alfred J.; OORSCHOT, Paul C. van e VANSTONE, Scott A.. *Handbook of Applied Cryptography.* Discrete Mathematics and Its Applications. New York: CRC Press LLC, 1996.

[19] MOLLIN R. A. *On Factoring.* Int. J. Contemp. Math. Sciences, Vol. 3, 2008, no. 33.

[20] PIETILÄINEN, Henna. *Elliptic Curve in Cryptography on Smart Card.* Department of Computer Science - Faculty of Information and Technology - Helsinki University of Technology, Helsinki, October 30, 2000.

[21] SILVERMAN, Joseph H.; TATE, John. *Rational Points on Elliptic Curves.* Undergraduate Texts in Mathematics. New York: Springer, 1992.

[22] VASCO, M^a Isabel González. *Criptosistemas basados em Teoría de Grupos.* Tesis doctoral, Departamento de Matemáticas, Universidad de Oviedo, Julho 2003.

[23] WASHINGTON, Lawrence C..*Elliptic Curves*: Number Theory and Cryptography. Discrete Mathematics and its Aplications. Second edition, Boca Raton [etc.]: Chapman & Hall/CRC Taylor & Francis Group, 2008.

Sites consultados

[24] *An intro to Elliptical Curve Cryptography.* Jul. 20, 2004. Acedida em 10, Março, 2009.
http://www.deviceforge.com/articles/AT4234154468.html

[25] NICELY, Tomas R. *The Baillie-PSW primality test.* Acedida em: 20, Dezembro, 2008.
http://www.trnicely.net/misc/bpsw.html

[26] PARI/GP Development . Acedida em 20, Novembro,2008.
http://pari.math.u-bordeaux.fr/.

[27] Python Software Foundation. *Python.* Acedida em 31, Janeiro, 2009.
http://www.python.org/

[28] STEIN, William. *Cremona's tables of elliptic curves.* Acedida em 03, Março, 2009.
http://www.sagemath.org/doc/ref/module-sage.databases.cremona.html

[29] WANLESS, James. *SQUFOF. Acedida em: 12, Dezembro, 2008.*
http://factorization.blogspot.com/2007/11/squfof.html.

[30] WEISSTEIN, Eric W. *Elliptic Curve Primality Proving.* From MathWorld–A Wolfram Web Resource.Acedida em 07, Janeiro, 2009.

http://mathworld.wolfram.com/EllipticCurvePrimalityProving.html

[31] WEISTEIN, Eric W. *Lucas-Lehmer Test.* From MathWorld–A Wolfram Web Resource. Acedida em 07, Janeiro, 2009.

http://mathworld.wolfram.com/Lucas-LehmerTest.html.

[32] WEISSTEIN, Eric W. *Pollard Rho Factorization Metod.* From Mathworld — A wolfram Web Resourse. Acedida em: 12, Dezembro, 2008.

http://mathworld.wolfram.com/PollardRhoFactorizationMethod.html.

[33] WEISSTEIN, Eric W. *Baillie-PSW Primality Test.* From Mathworld — A wolfram Web Resourse. Acedida em: 19, Dezembro, 2008.

http://mathworld.wolfram.com/Baillie-PSWPrimalityTest.html

[34] WEISSTEIN, Eric W. *Rabin-Miller Strong Pseudoprime Test.* From MathWorld–A Wolfram Web Resource. Acedida em: 07, Janeiro, 2009.

http://mathworld.wolfram.com/Rabin-MillerStrongPseudoprimeTest.html