CRIPTOGRAFIA DE CHAVE PÚBLICA BASEADA EM CURVAS ELÍPTICAS

CURSO DE MESTRADO EM REDES - COPPE/UFRJ

MONOGRAFIA FINAL DE CURSO – COS 762 – FEVEREIRO/2003

AUTOR: ***JULIO CESAR BARBOSA***

ORIENTADOR: ***LUIS FELIPE M. DE MORAES***

1. Introdução

O objetivo desse trabalho é apresentar o uso de curvas elípticas como uma promissora técnica de criptografia com chave pública. Essa técnica foi inicialmente proposta em 1985 por Victor Miller [1] e Neal Koblitz [2] como uma atraente forma de implementação de um sistema de chave pública em algumas das aplicações existentes.

Esse texto pretende abordar o assunto apresentando, primeiramente, a motivação para o estudo dessa técnica, onde serão incluídos alguns argumentos, bem como estudos comparativos com dados concretos encontrados em trabalhos anteriores. Dessa forma, pretende-se começar chamando a atenção para o potencial associado ao uso dessa técnica. Num segundo momento, serão introduzidas noções de álgebra baseada em curvas elípticas sobre o conjunto dos *reais*, incluindo representações gráficas para cada uma dessas operações. Numa tentativa de convergir a teoria de curvas elípticas para uma técnica de criptografia, os itens seguintes abordarão aplicações dessas curvas sobre um conjunto finito de *inteiros* (incluindo a álgebra aplicada a esse conjunto) e como essa técnica irá se transformar em um sistema de criptografia com chave pública. Ao final, serão apresentados um exemplo simplificado da aplicação de criptografia com curvas elípticas, comentários e considerações finais sobre essa técnica. Todas as referências, que foram de grande relevância na confecção desse trabalho, também serão citadas nessa parte final.

2. Motivação

Dentro do contexto de criptografia, os sistemas existentes apoiam-se no fato de existirem problemas matemáticos que, dado o elevado nível de trabalho envolvido na sua resolução, tornam-se de difícil solução. Consequentemente, buscamos proteção no fato de que nosso "adversário" não conseguirá, mesmo contando com as mais modernas ferramentas computacionais, reverter a função de criptografia (na qual o sistema se baseia) e acessar os parâmetros do nosso sistema em um tempo aceitável.

Em termos de criptografia computacional, um problema matemático é dito "de difícil solução" quando, mesmo aplicando-se o algoritmo mais eficiente para resolvê-lo, esse leva um longo período para que sua execução se conclua. Esse tempo de execução possui uma relação direta com o tamanho dos dados de entrada do algoritmo utilizado. Cientistas da área defendem [5] o fato de que, em geral, um problema de fácil solução tem o tempo de execução polinomial, enquanto problemas de difícil solução tem esse tempo em formato exponencial. Consequentemente, estaremos interessados em saber o quanto um problema se torna difícil (tempo de execução) com o aumento do tamanho de sua entrada e, adicionalmente, em selecionar problemas que maximizem esse tempo, sempre que quisermos obter um sistema de criptografia mais seguro.

Sistemas de criptografia com chave pública (sistemas assimétricos [9]) foram inicialmente propostos por Whitfield Diffie e Martin Hellman em 1976 [15]. Esses sistemas trabalham com duas chaves diferentes, independentes e não facilmente deriváveis [8]: A chave pública, utilizada na codificação de uma mensagem cifrada, e, a chave privada, utilizada na sua descodificação [4][8]. Opcionalmente, quando se deseja assinar digitalmente uma mensagem, o emprego da chave pública e privada se inverte, ou seja, o remetente "assina" (codifica) a mensagem através de sua chave privada, enquanto o destinatário somente conseguirá descodificar essa mensagem aplicando a chave pública do remetente. A segurança está em poder armazenar a chave privada em segurança e ser computacionalmente impossível obter essa chave, a partir da mensagem cifrada e da chave pública correspondente.

Tem se observado que, ao se projetar sistemas criptográficos de chave pública, é necessário também haver um compromisso entre o nível de segurança e o tempo de resposta que se deseja obter [9]. Nesse aspecto, quanto mais desenvolvidos forem as ferramentas e algoritmos utilizados para violação dos sistemas de criptografia

existentes, maiores têm que ser os parâmetros (chaves) e, consequentemente, maior o esforço no trabalho de codificação e descodificação dos textos cifrados. É nesse ponto que as propostas e idéias discutidas devem ser avaliadas e comparadas entre si.

Os próximos dois itens apresentam as classes de sistemas de criptografia com chave pública (incluindo criptografia com curvas elípticas) e comparam esses sistemas quanto à segurança e eficiência de processamento. Os argumentos e conclusões apresentados, reforçam o fato da criptografia de curvas elípticas ser uma excelente opção, não obstante um maior grau de complexidade para seu entendimento. Esses tópicos servem, consequentemente, como forte motivação ao seu estudo.

2.1. Classes de Sistemas de Criptografia com Chave Pública

Muitas opções de sistemas de criptografia de chave pública já foram propostas, mas a maioria foi declarada comprovadamente insegura ou inviável em termos de uso prático. Atualmente, existem somente três tipos de sistemas de criptografia com chave pública considerados seguros e eficientes [5]. Esses sistemas estão classificados de acordo com o problema matemático em que eles se baseiam [5] [6]:

I. Sistemas de **fatoração de inteiros** (*Integer Factorization Systems – IFS*), baseados no **problema de fatoração de inteiros** (*Integer Factorization Problem – IFP*)

II. Sistemas de **logaritmo discreto** (*Discrete Logarithm System – DLS*), baseados no **problema do logaritmo discreto** (*Discrete Logarithm Problem – DLP*)

III. Sistemas de **curva elíptica** (*Elliptic Curve Discrete Logarithm System – ECDLS*), baseados no **problema do logaritmo discreto em curvas elípticas** (*Elliptic Curve Discrete Logarithm Problem – ECDLP*)

A tabela abaixo, exemplifica algumas das aplicações de cada uma dessas técnicas em implementações reais:

IFS	DLS	ECDSL
RSA Rabin-Williams	ElGamal DAS – US Gov. Diffie-Hellman/KE Schnorr	ECC

Tabela 2.1: Exemplo de sistemas de chave pública

Os problemas em que esses sistemas de criptografia com chave pública se baseiam estão definidos, mais detalhadamente, na tabela a seguir:

Problema: Fatoração de Inteiros	
Definição	Dado um número *n* que é o produto de dois valores primos grandes p e q (ou seja, $n = pq$), determinar p e q.
Base	Enquanto encontrar números primos grandes é uma tarefa relativamente fácil, o problema de fatorar o produto desses valores é considerada uma tarefa computacionalmente intratável [12].
Problema: Logaritmo Discreto	
Definição	Dado um primo p , o conjunto $Z_p = \{0, 1,..., p-1\}$, y, $g \in Z_p$,$(g > 0)$, determinar $x \in Z_p$ $(0 \le x \le p-2)$, onde $y = g^x \pmod p$
Base	O problema de calcular o $x = logd_g(y)$ (logaritmo discreto na base g), sendo g e y primos extensos, também é considerada uma tarefa computacionalmente intratável [13].
Problema: Logaritmo Discreto em Curvas Elípticas	
Definição	Dada uma curva elíptica E, definida por um conjunto finito de pontos de natureza F_q (q é o número elementos do conjunto) - $E(F_q)$ - e os pontos P, $Q \in E(F_q)$, determinar o inteiro l $(0 \le l \le q-1)$ tal que $Q = lP$
Base	Ao passo que é relativamente fácil determinar o ponto $Q = lP$ (veremos isso ao definir as operações em curvas elípticas), determinar l dados Q, $P \in E(F_q)$ é bem mais difícil [9]

Tabela 2.2: Características dos problemas na criptografia com chave pública

2.2. Comparação Entre Sistemas de Criptografia com Chave Pública

Ao se comparar os três sistemas de criptografia com chave pública apresentados no último item, iremos abordar dois aspectos básicos nessa avaliação: segurança e eficiência. Outros aspectos também são discutidos em [5], tais como: aceitação pública, interoperabilidade e detalhamento técnico, todos obtidos através da publicação de padrões ("standards" [5][18]) desses sistemas.

2.2.1 Segurança

O escopo dessa análise de segurança em sistemas criptográficos com chave pública, limita-se na segurança "teórica", ou seja, em como violar esses sistemas. Discussões acerca de segurança em termos práticos, tais como confiança pessoal em funcionários/responsáveis pelos sistemas, proteção e isolamento físico ou de acesso, políticas e medidas de segurança dentro das empresas, entre outros, embora sejam de grande relevância no contexto de segurança, não serão considerados nessa discussão.

A primeira pergunta ao se examinar a segurança de sistemas de criptografia com chave pública é: Violar o sistema requer, realmente, que resolvamos o problema matemático em que ele se baseia? Segundo [5], uma prova matemática formal é fornecida com cada uma dessas propostas. Baseando-se nesse argumento, vemos que a única forma de quebrar esses sistemas é por intermédio de algoritmos que tentem resolver, da forma mais eficiente possível, o problema proposto.

Os algoritmos propostos para resolver cada um dos problemas citados no item anterior, dividem-se em dois tipos [5]: algoritmos específicos ("special-purpose algoritms") e algoritmos genéricos. Em termos gerais, os primeiros tratam somente alguns casos isolados, onde parâmetros específicos são utilizados nesses sistemas, enquanto os demais não fazem qualquer tipo de restrição quanto à parametrização.

Algoritmos específicos se baseiam em determinados aspectos de "fraqueza" que podem ser aplicados em cada um dos casos. É razoável assumir que esses aspectos devem ser evitados ao se projetar os sistemas criptográficos. No caso da fatoração de inteiros, um algoritmo rápido pode ser projetado quando se utiliza fatores primos (p e q) pequenos. De forma similar, o problema do logaritmo de números discretos pode ser resolvido quando utilizamos fatores primos pequenos. Finalmente, no caso de curvas elípticas, duas pequenas classes de curvas denominadas

supersingulares (“supersingular elliptic curves”) e anômalas (“anomalous elliptic curves”) também apresentam aspectos de vulnerabilidade e, consequentemente, também podem ser tratadas por algoritmos específicos [5].

Os algoritmos genéricos, ao contrário, se propõem a resolver qualquer configuração encontrada em cada um desses problemas. Independente dos parâmetros utilizados, esses algoritmos sempre chegam à resposta [5]. A questão agora é: Quanto tempo esses algoritmos levam para encontrar a solução? A ordem de grandeza desse tempo irá nos permitir avaliar, dados os parâmetros de entrada, o quanto um sistema de criptografia com chave pública é seguro.

Ao avaliar e comparar as opções de sistemas de criptografia com chave pública, os cientistas da área se baseiam nos algoritmos genéricos e qual a complexidade (número de passos X tamanho da entrada) que cada um deles oferece. Os problemas de fatoração de inteiros e de logaritmos discretos admitem, em geral, algoritmos que executam em tempo sub-exponencial [5]. Esses problemas também são considerados “difíceis”, mas não tão difíceis quanto os que necessitam de algoritmos puramente exponenciais. Formalmente [5], o tempo de execução para os melhores algoritmos para esses problemas é:

$$O(\exp((c+O(1)))(\ln n)^{1/3}(\ln\ln n)^{2/3})$$, onde c é constante e $n = pq$

Por outro lado, o melhor algoritmo genérico para o problema dos logaritmos discretos em curvas elípticas é puramente exponencial e seu tempo de execução é [5]:

$$O(\sqrt{p})$$

Baseados nos valores de complexidade apresentados acima, podemos observar que o problema de logaritmos discretos em curvas elípticas é considerado mais “difícil” de resolver que os demais. Como exemplo concreto dessa superioridade, a Figura 2.1 apresenta o tempo necessário para violar um sistema ECC em comparação com as aplicações RSA e DSA. Esse esforço para quebra dos sistemas está apresentado em unidades de MIPS (número de anos que uma máquina, capaz de executar um milhão de instruções por segundo, leva para resolver o problema).

Vale ressaltar que, uma medida atualmente considerada como parâmetro “razoável” de segurança é 10^{12} MIPS (10^{12} anos) [6].

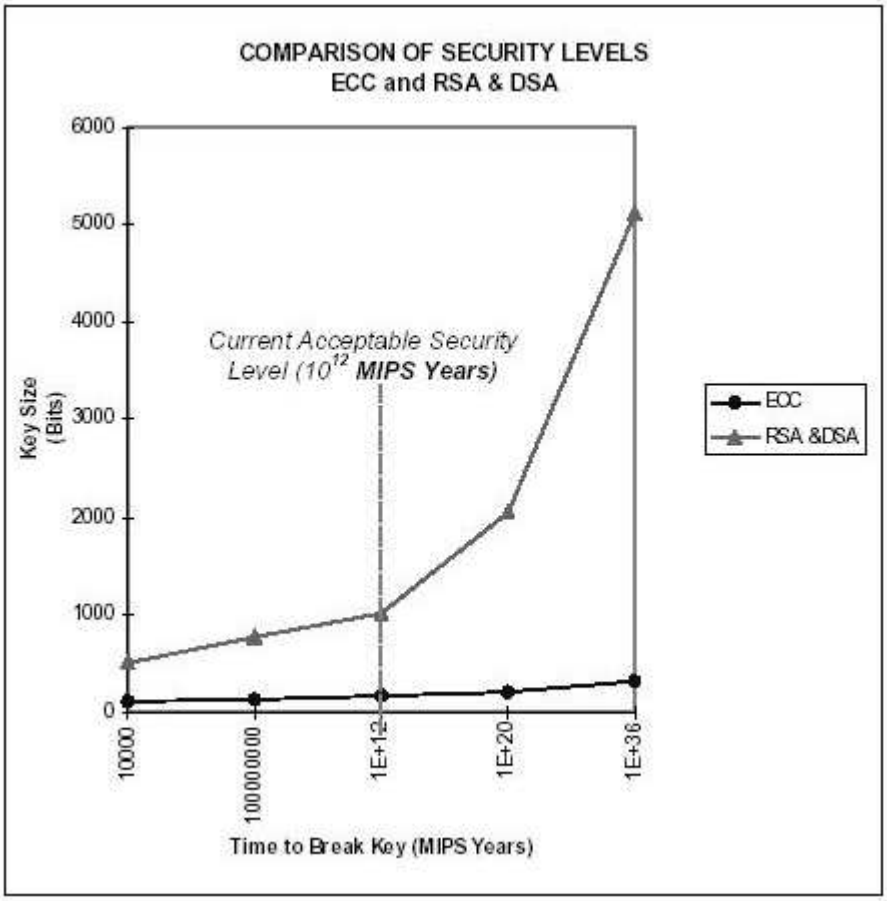

Figura 2.1: Segurança X Tamanho de Chave (ECC, RSA e DSA) [5]

Podemos analisar na figura acima o fato de que, para um nível de segurança razoável (10^{12} MIPS), enquanto o RSA e o DSA necessitam de 1024 bits, o ECC precisa de somente 160 bits para o tamanho de chave. Um outro fato interessante é que o aumento do nível de segurança (MIPS maior) necessita de um aumento bem mais expressivo do tamanho das chaves do RSA e DSA, em comparação ao ECC. Isso evidencia que o aumento dos atuais parâmetros de segurança, irá exigir um crescimento do tamanho da chave bem mais significativo no caso do RSA e DSA do que no ECC.

O próximo item, que prossegue o trabalho comparativo em termos de eficiência, voltará a abordar os aspectos referentes ao tamanho de chave entre essas opções de sistemas criptográficos.

2.2.2 Eficiência

A discussão acerca da eficiência de cada um dos sistemas de criptografia descritos aqui, leva em consideração os seguintes fatores: Carga computacional, tamanho de chave e tamanho de banda. Para uma comparação mais justa, os dados apresentados levam em consideração o mesmo nível de segurança para todas as propostas (ECC, RSA ou DSA).

I. **Carga Computacional**: Mede a eficiência com que os algoritmos podem implementar as transformações com as chaves públicas e privadas (sistema em operação). As melhores implementações de cada um dos sistemas (“state-of-the-art implementations”) indicam [5] que o ECC executa, aproximadamente, 10 vezes mais rápido que o RSA ou DSA.

II. **Tamanho de Chave**: Conforme visto no item anterior, o ECC também apresenta grande vantagem nesse aspecto. A tabela abaixo consolida essa vantagem:

	Parâmetros do Sistema	Chave Pública (bits)	Chave Privada (bits)
RSA	n/a	1088	2048
DSA	2208	1024	160
ECC	481	161	160

Tabela 2.3: Tamanho dos parâmetros e do par de chaves [5]

III. **Tamanho de Banda**: Corresponde a quantos bits (a mais) temos que transmitir após criptografar ou assinar uma mensagem, em relação a mensagem original. Todas as três opções apresentam

valores parecidos nesse quesito, com o ECC se destacando exclusivamente nos casos em que queremos processar mensagens pequenas [5]. Se visualizarmos os sistemas de criptografia com chave pública como eficiente ferramenta de troca de chave de seção (usa transformação de mensagens pequenas), essa vantagem do ECC torna-se ainda mais significativa. As tabelas abaixo, demonstram como o ECC torna-se, isoladamente, a melhor opção quando transformamos textos pequenos (100 bits) em relação às mensagens mais longas (assinaturas de 2000 bits de comprimento):

	Tamanho da assinatura (bits)
RSA	1024
DSA	320
ECC	320

Tabela 2.4: Assinatura em mensagens longas (2000 bits) [5]

	Mensagem codificada (bits)
RSA	1024
ElGamal	2048
ECC	321

Tabela 2.5: Criptografia de mensagens curtas (100 bits) [5]

Definitivamente, uso da criptografia com chave publica baseada em curvas elípticas é uma excelente opção, não somente em termos de nível de segurança, como também em todos os principais pontos relativos à eficiência de operação. Dessa forma, concluímos a apresentação da motivação ao estudo dessa tão promissora

técnica (item 2), passando a abordar, nos próximos itens, a técnica de criptografia com curvas elípticas de forma mais especifica.

3. Álgebra de Curvas Elípticas

Antes de entrarmos no estudo da aplicação direta de curvas elípticas em criptografia, esse item propõe-se a introduzir alguns conceitos básicos sobre curvas elípticas, suas propriedades e a álgebra envolvida (álgebra de curvas elípticas).

É importante frisar que curvas elípticas não são elipses. Elas têm esse nome pois são definidas como um objeto matemático (uma curva) descrito por uma equação cúbica [8], as mesmas que usamos para calcular o comprimento de arco de uma elipse [10]. Essas curvas, que podem assumir diversas formas (dependendo dos parâmetros utilizados), possuem propriedades interessantes [3] e nosso interesse nelas está justamente nessas propriedades. Em particular, podemos definir, a partir do conjunto de soluções (pontos) dessa curva, operações específicas e um elemento identidade, como veremos mais adiante.

Equações cúbicas para curvas elípticas têm a seguinte forma geral [11]:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_5 \qquad (3.1)$$

Inicialmente, assumiremos que os valores de x e y pertencem ao plano dos reais. De forma mais genérica, eles poderiam ser valores inteiros, complexos, base canônica ou qualquer outro tipo de elemento de corpo [11]. Nesse item, iremos nos ater ao plano dos reais, onde visualização dessas curvas é mais natural.

A equação abaixo [11] é uma forma mais simplificada da equação 3.1:

$$y^2 = x^3 + a_4x + a_5 \qquad (3.2)$$

Ao desenharmos a curva acima com os parâmetros $a_4 = -4$ e $a_5 = 0.67$, obtemos o seguinte gráfico:

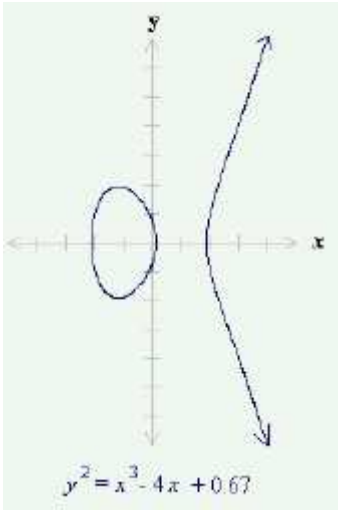

Figura 3.1: Exemplo de Curva Elíptica [3]

É interessante notar o quão diferente o aspecto dessa curva pode ser, a partir da variação dos parâmetros a_4 e a_5. Nos gráficos abaixo, podemos visualizar variações de formato de curvas elípticas, a partir de uma pequena alteração dos parâmetros da mesma.

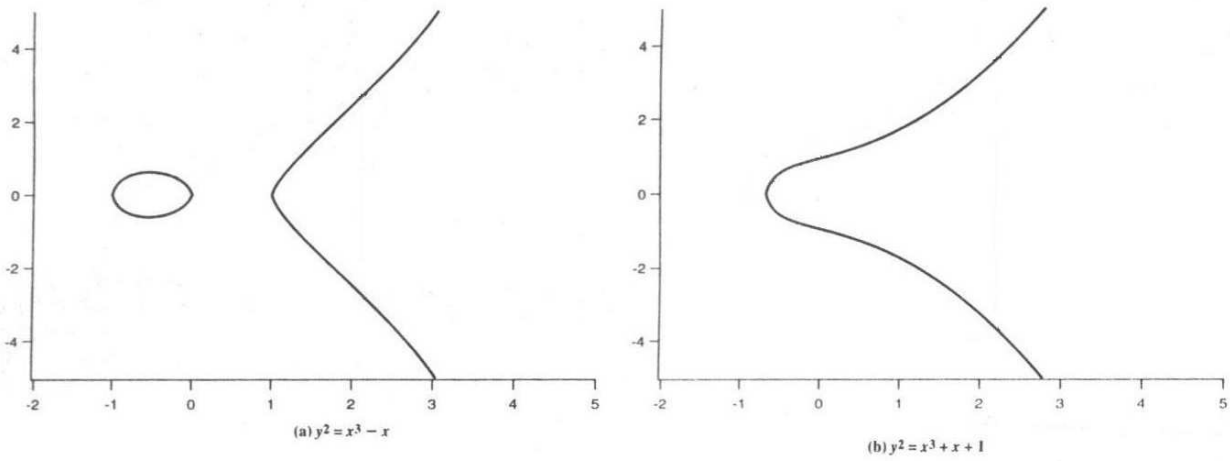

Figura 3.2: Variações de curvas elípticas [8]

Podemos definir uma “soma” de dois pontos pertencentes a uma curva elíptica (álgebra para curvas elípticas, [11]) como sendo um outro ponto na curva. Um elemento identidade (citado anteriormente) nessa álgebra seria um ponto O (chamado de ponto no infinito), tal que a soma de qualquer outro ponto da curva a esse ponto resulta no próprio ponto (equivalente à soma de um inteiro > 0 com 0, na álgebra tradicional). Dessa forma, dada uma curva elíptica E e os pontos $P, O \in E$:

$$O = -O \qquad (3.3)$$

$$P + O = P \qquad (3.4)$$

Graficamente, a “soma elíptica” de dois pontos P e Q presentes em uma curva elíptica é obtida através de uma reta que atravesse esses dois pontos, atravessando também um terceiro ponto dessa curva, que representa o resultado dessa soma “rebatido” no eixo horizontal. Também em termos gráficos, o ponto O está localizado em um lugar infinitamente distante, sobre o eixo vertical. A figura abaixo representa o ponto resultante (R) da soma de P e Q, matematicamente:

$$R = P + Q \qquad (3.5)$$

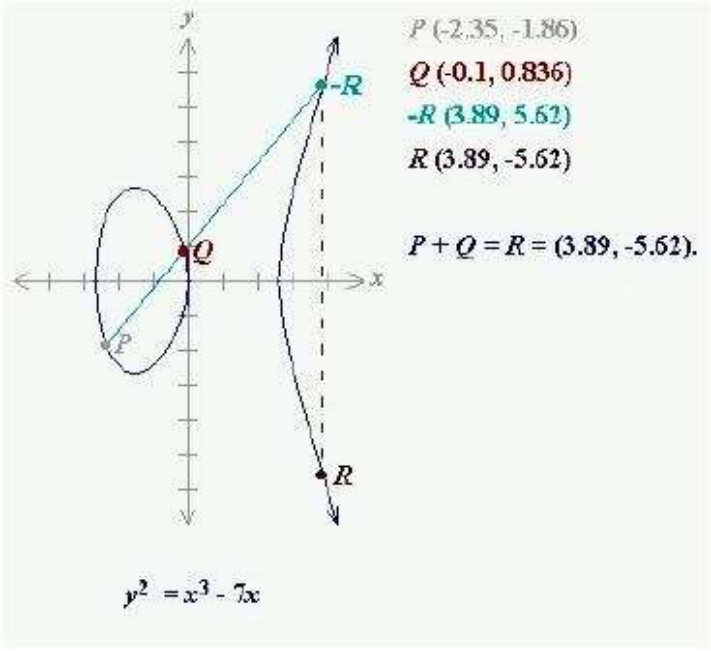

Figura 3.3: Soma elíptica ($R = P + Q$) [3]

Dessa forma, poderíamos representar operações de soma nas curvas apresentadas na Figura 3.2 como:

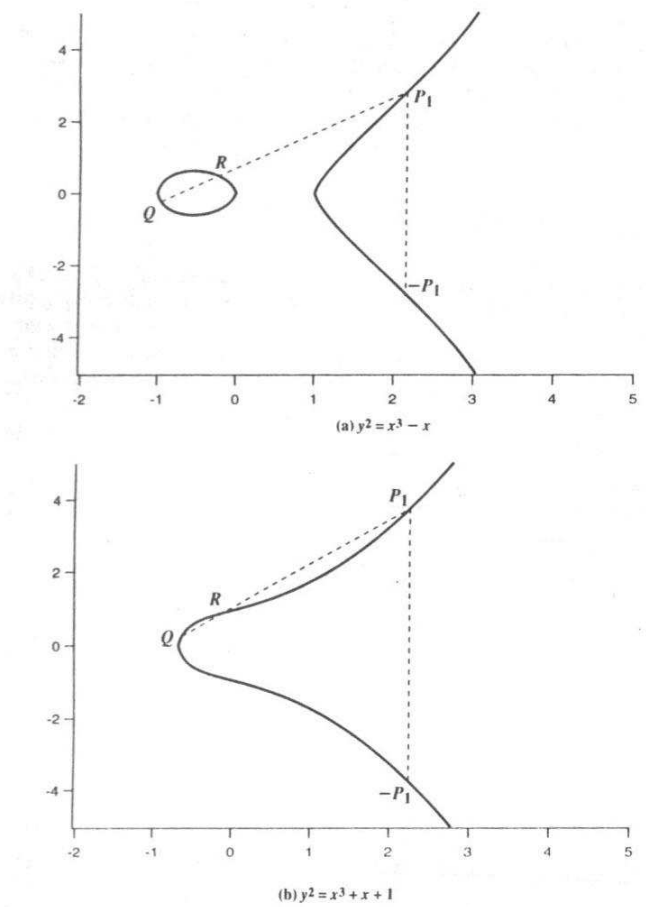

Figura 3.4: Soma em curvas elípticas ($Q + R = -P_1$) [8]

Adicionalmente, podemos representar as operações de soma de um ponto com ele mesmo ("dobrar" o ponto) como um ponto "rebatido" no eixo horizontal, a partir do ponto de interseção entre a reta tangente ao ponto que se deseja "dobrar" e a curva. A soma de um ponto com seu "negativo", assim como a "dobra" de um ponto $P = (x_p, y_p)$, onde $y_p = 0$, nos leva ao ponto O. As próximas figuras exemplificam esses três casos. Matematicamente, teríamos:

$$R = 2P = P + P \qquad (3.6)$$

$$P = (x_p, y_p) + [(-P) = (x_p, -y_p)] = O \qquad (3.7)$$

$$P = (x_p, y_p) \text{ onde } y_p = 0 \Rightarrow 2P = O \qquad (3.8)$$

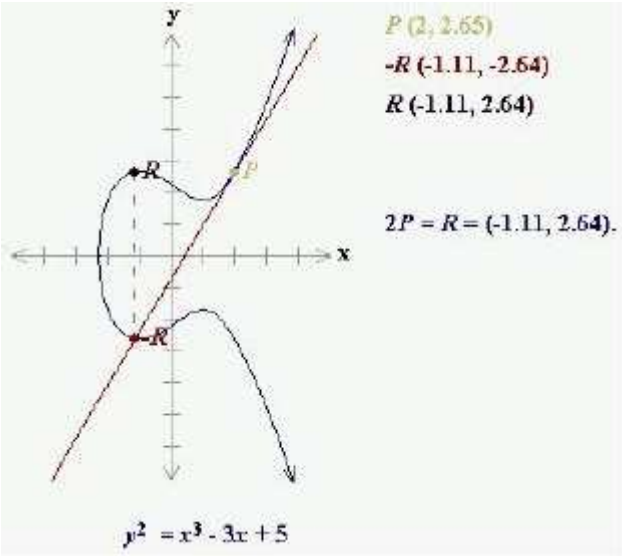

Figura 3.5: Soma de um mesmo ponto ($R = P + P = 2P$) [3]

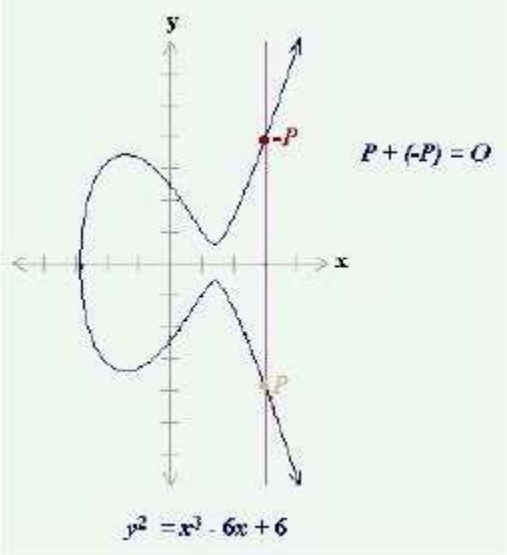

Figura 3.6: Soma de um ponto com seu oposto ($P + (-P) = O$) [3]

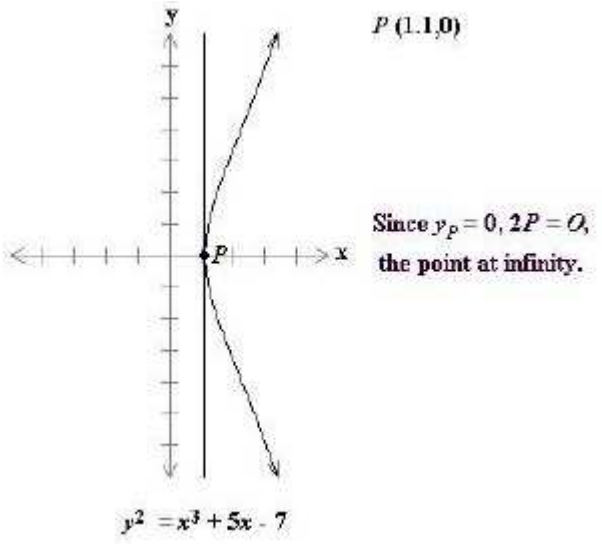

Figura 3.7: “Dobro” de um ponto (x_p, y_p), onde $y_p = 0$ [3]

A multiplicação de um ponto em uma curva elíptica por um valor inteiro qualquer (maior que 2), segue a mesma forma da equação (3.6), ou seja, basta somar o ponto a ele mesmo tantas vezes quanto for o fator de multiplicação. Dessa forma, teríamos, matematicamente:

$$R = lP = P + P + ... + P \quad (l \text{ vezes}) \qquad (3.9)$$

A título de exemplo [11], a partir da Figura 3.5, se estivéssemos interessados em calcular $Q = 15P$, bastaria reaplicar, continuamente, as operações de soma e soma de mesmo ponto, de forma a obter o valor desejado (também um ponto na curva):

$$Q = 15P = P + 2(P + 2(P + 2P)) \qquad (3.10)$$

Adição e multiplicação de pontos de uma curva elíptica são as principais operações feitas nessas curvas. Essas operações podem ser implementadas tanto em hardware quanto em software e existem algoritmos matemáticos bastante eficientes para realizá-las [14]. Essa característica confere à criptografia com curvas elípticas uma boa opção também quanto ao desempenho de operação, conforme citado no item 2.2.2 (Carga Computacional).

Ao aplicar essas operações de forma restrita a um grupo finito de inteiros (próximo item), os proponentes da criptografia com curva elíptica observaram o potencial dessa técnica (problema ECDLP). Conforme citado na Tabela 2.2, eles se basearam no fato de que, embora seja relativamente fácil determinar o ponto $Q = lP$, determinar o inteiro l dados Q, $P \in E(F_q)$ (E é uma curva elíptica no grupo F_q) é bem mais difícil e não possui solução em tempo sub-exponencial.

4. Curvas Elípticas sobre Corpos Finitos

Embora estejamos mais familiarizados com o uso de curvas elípticas aplicada ao plano dos reais, seu uso em sistemas criptográficos não seria prático e tão pouco preciso. Isso de deve, principalmente, devido à problemas com arredondamento, "trucagem" de valores e limites [3]. Aplicações de criptografia necessitam de uma aritmética rápida e precisa, que pode ser obtida através do uso de corpos inteiros finitos [5]. Por essas razões, a partir desse item, consideraremos somente curvas elípticas definidas sobre um conjunto finito de valores inteiros, denominado "campo finito": Fq.

Duas das mais eficientes e freqüentemente utilizadas opções de Fq são [3][5]: **Curvas elípticas sobre corpos finitos primos** ($Fq = Fp$) e **curvas elípticas sobre corpos finitos de característica dois** ($Fq = F_{2m}$). Essas curvas também são denominadas, respectivamente, **curvas em campos de característica *p*** e **curvas em campos de característica *2*** [14]. As duas opções são tratadas, separadamente, nos próximos itens.

4.1. Curvas Elípticas sobre Corpos Finitos Primos (Fp)

O conjunto Fp é composto por valores de 0 a $p-1$ ($Fp = \{0, 1, ..., p-1\}$) e todas as operações devem ser finalizadas calculando-se o resto da divisão por p. Dessa forma, sempre alcançamos resultados dentro de Fp.

Uma curva elíptica sobre o campo Fp é definida, a partir da equação (3.2), pela equação a seguir [16]:

$$y^2 \pmod p = x^3 + ax + b \pmod p \qquad (4.1)$$

onde $a, b \in Fp$. A curva elíptica inclui todos os pontos (x, y) que satisfazem a equação (4.1), mais o ponto O, e $x, y \in Fp$.

Adicionalmente, é importante citar que, para garantir que $(x^3 + ax + b)$ não possua fatores repetidos (a curva realmente forme um grupo), devemos obedecer à seguinte restrição [3][8][16]:

$$4a^3 + 27b^2 \pmod{p} \neq 0 \qquad (4.2)$$

As equações a seguir definem as operações de soma de pontos numa curva elíptica sobre Fp [3][14]:

$P_3(x_3, y_3) = P_1(x_1, x_1) + P_2(x_2, y_2)$, onde $P_3 \neq O$

$$x_3 = \lambda^2 - x_1 - x_2 \pmod{p} \qquad (4.3)$$

$$y_3 = (x_1 - x_3)\lambda - y_1 \pmod{p}, \text{ onde:} \qquad (4.4)$$

$$x1 \neq x2 \Rightarrow \lambda = \frac{(y2 - y1)}{(x2 - x1)} \pmod{p}$$

$$x_1 = x_2, y_2 \neq 0 \Rightarrow \lambda = \frac{(3x_1^{\,2} + a)}{(2y_1)} \pmod{p}$$

4.2. Exemplo de Curvas Elípticas sobre Fp (F_{23}) [16]

Considerando uma curva elíptica $E(F_{23})$ com a equação $y^2 = x^3 + x$ ($a = 1$ e $b = 0$), podemos observar que o ponto $(9, 5)$ satisfaz a equação, desde que:

$$y^2 \pmod{p} = x^3 + x \pmod{p}$$

$$(5)^2 \pmod{23} = (9)^3 + (9) \pmod{23}$$

$$25 \bmod 23 = (729 + 9) \bmod 23$$

$$25 \bmod 23 = 738 \bmod 23$$

$$2 = 2 \quad \Rightarrow \quad (9, 5) \in E(F_{23})$$

Nesse exemplo, os 23 pontos que satisfazem a equação são:

(0,0) (1,5) (1,18) ***(9,5)*** *(9,18) (11,10) (11,13) (13,5)*
(13,18) (15,3) (15,20) (16,8) (16,15) (17,10) (17,13) (18,10)
(18,13) (19,1) (19,22) (20,4) (20,19) (21,6) (21,17)

Esses pontos que, junto com o ponto O, compõem a curva elíptica dada, também podem ser apresentados sob a forma gráfica. Conforme a figura abaixo, podemos verificar que o conjunto de pontos, embora não se assemelhe ao formato das curvas apresentadas no plano dos reais, define uma simetria em torno do eixo $y = 11{,}5$. Esses pontos "simétricos" pelo eixo imaginário, definem todos os pontos e seus respectivos "negativos" na curva (ver equação (3.7)).

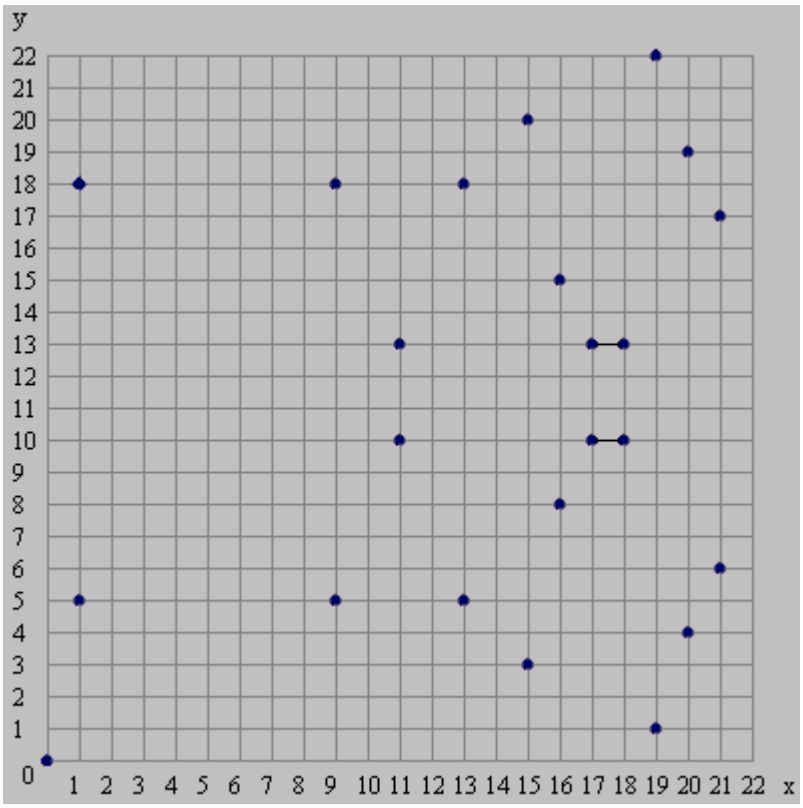

Figura 4.1: Curva Elíptica $y^2 = x^3 + x$ (mod p) em F_{23} [16]

4.3. Curvas Elípticas sobre Corpos Finitos de Característica Dois (F_{2m})

O conjunto F_{2m} é composto por strings de m bits e possui dois principais tipos de representação: **Representação polinomial** e **representação de base ótima**. Segundo [17], para implementações de hardware, a representação de base ótima é a melhor opção, ao passo que em software, o uso da representação polinomial é mais indicado. Devido ao caráter mais simplificado, a representação polinomial é a preferida em todas as publicações de cunho tutorial, logo, aqui também, estaremos exemplificando F_{2m} através do seu emprego. Em [17] é possível ver, de forma simplificada, o uso da representação de base ótima. Uma terceira representação, chamada **representação de subcampos**, também pode ser conferida em [18].

A conveniência do uso de um conjunto de strings de m bits está no fato de nos aproximarmos mais da representação "natural" de palavras de dados de m bits em memória [3]. Como resultado do campo F_{2m} ser "de característica 2", a equação que representa uma curva elíptica em F_{2m} é ligeiramente diferente das equações (3.2) e (4.1), apresentadas anteriormente. Vale ressaltar que, essa equação (apresentada abaixo) é aplicável tanto nas representações polinomiais de F_{2m} quanto nas de base ótima [16][17], possuindo como única restrição: $a, b \in F_{2m}$ e $b \neq 0$ [3][16].

$$y^2 + xy = x^3 + ax^2 + b \quad (em\ F_{2m}) \qquad (4.5)$$

A representação polinomial de F_{2m} é definida por um **conjunto de polinômios binários** (coeficientes binários) **de grau** $\leq m-1$, conforme definido a seguir [3][16]:

$$F_{2m} = \{a_{m-1}\, x^{m-1} + a_{m-2}\, x^{m-2} + \ldots + a_1 x + a_0 \mid a_i \in \{0, 1\}\} \qquad (4.6)$$

Os elementos desse conjunto F_{2m} (que contém 2^m elementos) também podem ser escritos na forma de vetor: $(a_{m-1}, \ldots, a_1, a_0)$. Dessa forma, poderíamos ter um conjunto F_{2^4} representado, de forma equivalente, pelos conjuntos abaixo (com $2^4 = 16$ elementos):

$$F_{2^4} = \{(0001), (0010), (0100), (1000), (0011), (0110), (1100), (1011), (0101), (1010),\ldots\}$$
$$= \{(1), (x), (x^2), (x^3), (x + 1), (x^2 + x), (x^3 + x^2), (x^3 + x + 1), (x^2 + 1), (x^3 + x),\ldots\}$$

Assim como em Fp, quando operamos elementos desse conjunto, o resultado deve produzir um polinômio também em F_{2m}, de forma que todas as operações devolvam valores fechados nesse conjunto. Isso é obtido através do uso de um polinômio irredutível de ordem m: $f(x)$ [3][16]. Esse polinômio, também utilizado ao se determinar uma curva elíptica em F_{2m}, tem o seguinte formato [16]:

$$f(x) = x^m + f_{m-1}\, x^{m-1} + f_{m-2}\, x^{m-2} + \ldots + f_1 x + f_0 \qquad (4.7)$$

Operações de adição (XOR bit a bit) e multiplicação (módulo $f(x)$) são as mais importantes, pois delas podem ser derivadas subtração e exponenciação (multiplicação múltipla) em F_{2m} [16].

Outra definição importante dentro do contexto do uso do F_{2m}, é o elemento gerador desse conjunto, chamado de g. Um elemento gerador g é um dos elementos de F_{2m}, a partir do qual podemos gerar todo o conjunto, bastando calcular as potências desse elemento [16]. Dessa forma, quando apresentamos o exemplo de F_{24} na última página, na verdade, utilizamos como gerador $g = (0010)$ (em forma de polinômio: $\{x\}$) e a função irredutível $f(x) = x^4 + x + 1$. Os demais elementos daquele conjunto foram resultado da potenciação desse valor quando, então, obtivemos:

$F_{24} = \{g^0, g^1, g^2, g^3, g^4, g^5, g^6, g^7, g^8, g^9, g^{10}, g^{11}, g^{12}, g^{13}, g^{14}, g^{15}\}$, onde:

$g^0 = \{1\} = \{0x^3 + 0x^2 + 0x + 1\} = (0001)$

$g^1 = \{x\} = \{0x^3 + 0x^2 + 1x + 0\} = (0010)$

$g^2 = \{x^2\} = \{0x^3 + 1x^2 + 0x + 0\} = (0100)$

$g^3 = \{x^3\} = \{1x^3 + 0x^2 + 0x + 0\} = (1000)$

$g^4 = \{x^4\} \bmod f(x) = \{-x - 1\} = \{0x^3 + 0x^2 - 1x - 1\} = (0011)$ Obs: $g^i \bmod f(x)$, $i \geq 4$, $\Rightarrow$ resultado em F_{24}

$g^5 = \{x^5\} \bmod f(x) = \{-x^2 - x\} = \{0x^3 - 1x^2 - 1x + 0\} = (0110)$

$g^6 = \{x^6\} \bmod f(x) = \{-x^3 - x^2\} = \{-1x^3 - 1x^2 + 0x + 0\} = (1100)$

$g^7 = \{x^7\} \bmod f(x) = \{-x^3 + x + 1\} = \{-1x^3 + 0x^2 + 1x + 1\} = (1011)$

$g^8 = \{x^8\} \bmod f(x) = \{x^2 + 2x + 1\} = \{0x^3 + 1x^2 + 2x + 1\} = (0101)$

$g^9 = \{x^9\} \bmod f(x) = \{x^3 + 2x^2 + x\} = \{1x^3 + 2x^2 + 1x + 0\} = (1010)$

$g^{10} = \{x^{10}\} \bmod f(x) = \{2x^3 + x^2 - x - 1\} = \{2x^3 + 1x^2 - 1x - 1\} = (0111)$

$g^{11} = \{x^{11}\} \bmod f(x) = \{x^3 - x^2 - 3x - 2\} = \{1x^3 - 1x^2 - 3x - 2\} = (1110)$

$g^{12} = \{x^{12}\} \bmod f(x) = \{-x^3 - 3x^2 - 3x - 1\} = \{-1x^3 - 3x^2 - 3x - 1\} = (1111)$

$g^{13} = \{x^{13}\} \bmod f(x) = \{-3x^3 - 3x^2 + 1\} = \{-3x^3 - 3x^2 + 0x + 1\} = (1101)$

$g^{14} = \{x^{14}\} \bmod f(x) = \{-3x^3 + 4x + 3\} = \{-3x^3 + 0x^2 + 4x + 3\} = (1001)$

$g^{15} = \{x^{15}\} \bmod f(x) = \{4x^2 + 6x + 3\} = \{0x^3 + 4x^2 - 6x + 3\} = (0001)$

Em aplicações reais de curvas elípticas sobre F_{2m}, o valor de m deve ser tal que permita a geração de uma tabela (similar a apresentada acima) grande o suficiente, de forma a tornar o sistema imune a quebras [16]. Hoje em dia, conforme visto na Figura 2.1, o uso de $m = 160$ tem se mostrado uma boa opção.

As equações a seguir, definem as operações de soma de pontos numa curva elíptica sobre F_{2m}. Podemos observar que essas equações [3][14] são ligeiramente diferentes das que tínhamos em Fp (ver equações (4.3) e (4.4)).

$P_3(x_3, y_3) = P_1(x_1, x_1) + P_2(x_2, y_2)$, onde $P_3 \neq O$

$x_3 = \lambda^2 + \lambda + a + x_1 + x_2 \quad (em\ F_{2m})$ (4.8)

$y_3 = (x_1 + x_3)\lambda + x_3 + y_1 \quad (em\ F_{2m})$, onde: (4.9)

$$x1 \neq x2 \Rightarrow \lambda = \frac{(y2 + y1)}{(x2 + x1)} (em\ F_2m)$$

$$x_1 = x_2 \neq 0 \Rightarrow \lambda = \frac{(x_1^{\ 2} + y_1)}{(x_1)} (em\ F_2m)$$

4.4. Exemplo de Curvas Elípticas sobre F_{2m} (F_{24}) [16]

Considerando agora uma curva elíptica $E(F_{24})$ com a equação $y^2 + xy = x^3 + g^4x^2 + 1$ ($a = g^4$ e $b = g^0 = 1$), podemos observar que o ponto (g^5, g^3) satisfaz a equação, desde que:

$$y^2 + xy = x^3 + g^4x^2 + 1$$

$$(g^3)^2 + (g^5)(g^3) = (g^5)^3 + g^4 (g^5)^2 + 1$$

$$g^6 + g^8 = g^{15} + g^{14} + 1$$

$$(1100) + (0101) = (0001) + (1001) + (0001)$$

$$(1001) = (1001) \quad \Rightarrow \quad (g^5, g^3) \in E(F_{24})$$

Nesse exemplo, os 15 pontos que satisfazem a equação são:

$$(1, g^{13})\ (g^3, g^{13})\ (g^5, g^{11})\ (g^6, g^{14})\ (g^9, g^{13})\ (g^{10}, g^8)\ (g^{12}, g^{12})$$

$$(1, g^6)\ (g^3, g^8)\ \mathbf{(g^5, g^3)}\ (g^6, g^8)\ (g^9, g^{10})\ (g^{10}, g)\ (g^{12}, 0)\ (0, 1)$$

Esses pontos que, junto com o ponto O, compõem a curva elíptica dada, também podem ser apresentados sob a forma gráfica, conforme a figura abaixo:

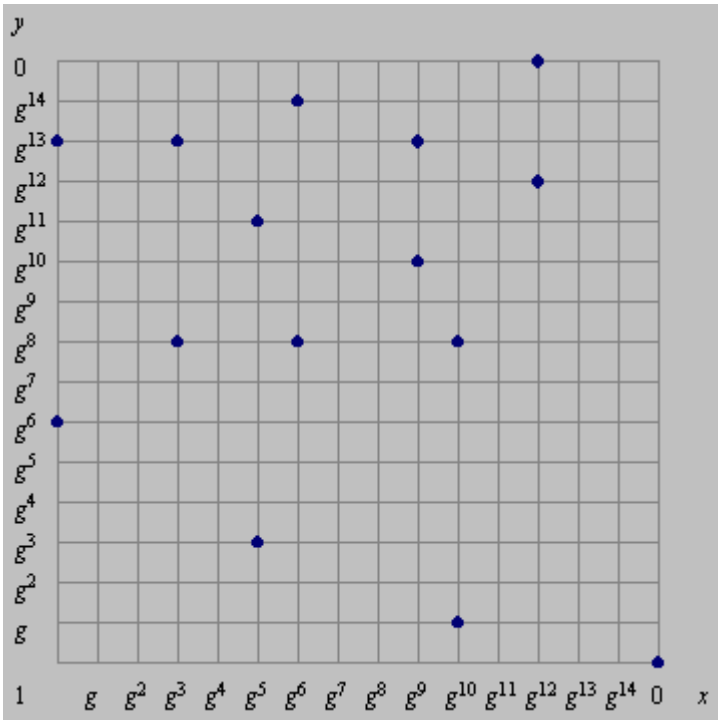

Figura 4.2: Curva Elíptica $y^2 + xy = x^3 + g^4x^2 + 1$ em F_{24} [16]

A solução do ECDLP com corpos primos ou de característica dois, apresenta, aparentemente, a mesma dificuldade para instâncias que utilizam mesmo tamanho (aproximado) para p ou 2^m. Segundo [5] e [18], não existe estudo matemático que

tenha comprovado se o ECDLP sobre F_{2m} é mais difícil ou fácil de resolver, se comparado ao Fp.

5. Aplicação de Curvas Elípticas em Criptografia

Ao se projetar um sistema de criptografia baseado em curvas elípticas, é necessário determinar, num primeiro momento, quais são as características gerais do sistema, onde serão definidos todos as operações e parâmetros genéricos que todos os componentes irão utilizar. Num segundo momento, cada um dos usuários desse sistema terá que definir seus parâmetros pessoais (chaves), de forma a viabilizar sua participação no sistema.

Ao se determinar as características gerais do sistema, precisamos realizar os seguintes passos [6][18]:

I. Definir a natureza de seu campo finito Fq (Fp ou F_{2m});

II. Selecionar a representação para os elementos em Fq (polinomial, base ótima, subcampos, etc);

III. Implementar aritmética e operações em Fq (ver equações (4.3), (4.4), (4.8) e (4.9));

IV. Selecionar uma curva apropriada em Fq (quais parâmetros utilizar para a curva)

V. Definir um ponto gerador em $E(Fq)$

VI. Definir o mapeamento da mensagem original em pontos de uma curva (“embedding”)

Os itens I, II e III, citados acima, já foram abordados durante nossa descrição de curvas elípticas sobre corpos finitos primos (tópico **4** e subsequentes).

O item IV aborda o problema de como melhor escolher os parâmetros da curva elíptica, de forma a tornar seu sistema mais seguro. No tópico 2.2.1, quando discutimos aspectos de segurança e citamos a existência de algoritmos específicos para inversão das funções de criptografia, observamos o fato de que algumas classes de curva (supersingulares e anômalas) deveriam ser evitadas. Da mesma forma, existem técnicas que possuem o único propósito de auxiliar na escolha de parâmetros

de curva apropriados. Entre essas técnicas, podemos citar: **método baseado no teorema de Hanssen**, o **método global**, o **método da multiplicação de complexos** e o **método randômico**. A descrição desses métodos pode ser obtida em [18]. Discussões mais extensas sobre o que deve ser evitado e quais as "boas práticas" em termos de parametrização dessas curvas também podem ser vistas em [3].

Além da definição da própria curva $E(Fq)$, como parte dos parâmetros globais que devem ser mantidos públicos, está um ponto denominado **ponto gerador** [19] ou **ponto base** [3] $G \in E(Fq)$. Abordado no item V, esse ponto é uma referência que irá permitir a realização da criptografia. Segundo [3], esse ponto é análogo à base g do problema DLP (conforme descrito na Tabela 2.2, determinar x, dados g e $y = g^x \pmod{p}$). O ponto G é obtido a partir da escolha de um valor n primo grande tal que $nG = O$ (ponto no infinito) [3][19]. Todos os pontos $Pi \in E(Fq)$ têm uma "ordem" n_i, tal que, $n_iPi = O$, dessa forma, valor n é denominado "ordem" de G [3]. Também segundo [3], algo importante, mais nem sempre divulgado, é o fato que n deve ser grande o suficiente de forma a inviabilizar a obtenção de todos os múltiplos de G : G, $2G$, $3G$, $4G$, ..., $(n-1)G$. Mais adiante, iremos visualizar como n e G participam do processo de criptografia.

Um assunto pouco esclarecido nos textos que se propõem a introduzir a técnica de criptografia com curvas elípticas está sendo abordado no item VI, acima. Tudo que discutimos até agora sobre curvas elípticas, trata somente da base por trás emprego de curvas elípticas, de suas propriedades, dos pontos dessa curva e da aritmética, operações e natureza desses pontos. Faltava introduzir uma relação clara entre a mensagem original, representada por uma mensagem de texto puro ("plain text") e os pontos de uma curva elíptica, sobre os quais realizamos todas as operações comentadas. Esse procedimento, essencial ao uso da criptografia com curva elíptica, é denominado "message embedding" e consiste em uma forma de mapear a mensagem original (texto puro) em pontos de uma curva elíptica. Isso corresponde a colocar a mensagem original "sobre" [11] a curva elíptica definida. Esses pontos, depois de sofrerem operações parametrizadas pelas chaves individuais de um usuário, dão origem a um outro conjunto de pontos, que representa os pontos originais "cifrados". Aqui, observamos uma clara diferença entre os métodos de criptografia tradicional e a técnica baseada em curvas elípticas: Ao invés de texto cifrado, transmitimos um conjunto de "pontos cifrados" [8]. Esses "pontos cifrados", ao serem recebidos pelo destinatário, são convertidos, através da chave correspondente, nos pontos originais. Nesse momento, ao aplicarmos a rotina de mapeamento

invertida, recuperamos texto original e todo o "ciclo criptográfico" se fecha. O mapeamento de caracteres em pontos é citado em [8] e pode ser visto de forma exemplificada em [11].

Uma vez estabelecidos todos os parâmetros e características gerais nas quais o sistema de criptografia com curvas elípticas deve se basear, basta que cada usuário determine seus parâmetros individuais, ou seja, seu par de chaves pública/privada, e os demais parâmetros locais necessários ao resto da implementação. Dessa forma, cada usuário possui um **valor** $n_A < n$ (conforme citado anteriormente, n é a ordem do ponto gerador G) como **chave privada** que possui, como **chave pública correspondente**, um **ponto** $P_A = n_A G$. Observe que o ponto P_A é derivado de n_A e, assim como G, $P_A \in E(Fq)$. Conforme descrito no último parágrafo, iremos supor que a mensagem original M (nesse contexto, consideraremos a mensagem como sendo de tamanho de 1 caracter) tenha sido mapeada em um ponto P_M, dentro do grupo finito de pontos de uma curva elíptica $E(Fq)$. A partir desse momento, quando um usuário A deseja enviar uma mensagem cifrada para B (usando a chave pública P_B), deve seguir o seguinte procedimento [8][19]:

A: Escolhe, aleatoriamente, um inteiro k

A: Calcula, a partir de k, G, P_B e P_M , **um par** de pontos P_{C1} e P_{C2} :

$P_{C1} = (kG)$

$P_{C2} = (P_M + kP_B)$

A: Transmite para B o **par** de pontos cifrados P_C :

$P_C = [P_{C1}, P_{C2}]$

Do outro lado, quando B recebe a mensagem (par de pontos P_C), recupera P_M, a partir do segundo ponto (P_{C2}), da seguinte forma:

B: Embora B não conheça k, sabendo que: $kP_B = k\, n_B G$ (pois $P_B = n_B G$):

$P_{C1}\, n_B = (kG)\, n_B = kP_B \quad \Rightarrow \quad kP_B$ "sai" de P_{C1}

B: Para extrair P_M de P_{C2}, basta calcular $P_{C2} - kP_B$, assim:

$P_{C2} - P_{C1}\, n_B = (P_M + kP_B) - kP_B = (P_M + [(kG)\, n_B]) - [(kG)\, n_B]) = \boldsymbol{P_M}$

Esquemas similares para obtenção de texto cifrado, assim como assinatura digital em curvas elípticas, também podem ser encontrados em [3], [8], [14], [18] e [19]. A seguir, descrevemos um exemplo simplificado de uma troca de mensagem cifrada, a partir dos passos descritos anteriormente:

5.1. Exemplo de Criptografia com Curvas Elípticas [19]

Considerando uma curva elíptica $E(F_{751})$ com a equação $y^2 = x^3 - x + 188 \pmod{751}$ ($a = -1$ e $b = 188$), e ponto gerador $G = (0, 376)$. Consequentemente, os múltiplos kG do ponto gerador G são ($1 \leq k \leq 751$):

$G = (0, 376)$	$2G = (1, 376)$	$3G = (750, 375)$	$4G = (2, 373)$
$5G = (188, 657)$	$6G = (6, 390)$	$7G = (667, 571)$	$8G = (121, 39)$
$9G = (582, 736)$	$10G = (57, 332)$	... ...	$761G = (565, 312)$
$762G = (328, 569)$	$763G = (677, 185)$	$764G = (196, 681)$	$765G = (417, 320)$
$766G = (3, 370)$	$767G = (1, 377)$	$768G = (0, 375)$	$769G = O$ (ponto no infinito)

Quando A decide mandar uma mensagem M para B, ele deve saber, a partir de alguma estratégia [8], a chave pública do destinatário. Se o destinatário desse exemplo escolheu o valor $n_B = 85$ como chave privada, teremos que a chave pública correspondente à mesma é o ponto $P_B = n_B G = 85(0, 376) \Rightarrow P_B = (671,558)$. O procedimento de criptografia de P_M passa, então, pelas etapas:

A: Mapeia a mensagem M no ponto $P_M = (433, 253) \in E(Fq)$

A: Escolhe, digamos, $k = 113$

A: Calcula o par de pontos P_C :

$P_C = [(kG), (P_M + kP_B)]$

$P_C = [113\ (0, 376), (443, 253) + 113\ (671, 558)]$

$P_C = [(34, 633), (443, 253) + (47, 416)]$

$P_C = [(34, 633), (217, 606)]$

A: Transmite para B o par de pontos cifrados $P_C = [(34, 633), (217, 606)]$

B: Recebe o par de pontos cifrados $P_C = [(34, 633), (217, 606)$

B: Calcula, a partir de sua chave privada $n_B = 85$:

$(P_M + kP_B) - [n_B (kG)] = (217, 606) - [85(34, 633)]$

$(P_M + kP_B) - [n_B (kG)] = (217, 606) - [(47, 416)]$

$(P_M + kP_B) - [n_B (kG)] = (217, 606) + [(47, -416)]$ (ver equação (3.7))

$(P_M + kP_B) - [n_B (kG)] = (217, 606) + [(47, 335)]$ ($-416 = 335 \bmod 751$)

$(P_M + kP_B) - [n_B (kG)] = (443, 253)$

B: Mapeia ponto $P_M = (433, 253)$ novamente na mensagem *M*

6. Considerações Finais

Podemos observar que o uso da criptografia com chave publica baseada em curvas elípticas é uma excelente opção, não somente em termos de nível de segurança, como também em todos os principais pontos relativos à eficiência de operação. Dadas suas características, essa técnica pode ser utilizada, principalmente, em sistemas "embutidos" ou sistemas com restrições físicas de espaço e/ou capacidade de processamento. Estudos recentes [14] apontam para o uso dessa técnica na implementação de cartões inteligentes ("smart cards"), entre outros tipos de aplicação.

7. Bibliografia

[1] V. Miller. *Uses of Elliptic Curves in Cryptography*. Advances in Cryptography, Crypto 85, Springs Verlag LNCS 218, 417-426, 1986

[2] M. Koblitz. *Elliptic Curve Cryptosystems*, Math Comp., 48, 203-209, 1987

[3] Gabriel Belingueres. *Introducción A Los Criptosistemas de Curva Elíptica*. http://www.toptutoriales.com/matematicas/criptografia/criptografia7.htm

[4] www.certicom.com – *An Introduction to Information Security*– A Certicom Whitepaper – March, 1997

[5] www.certicom.com - *Current Public-Key Cryptographic Systems* – A Certicom Whitepaper – April, 1997

[6] www.certicom.com – *Remarks on the Security of the Elliptic Curve Cryptosystem* – A Certicom Whitepaper – September, 1997

[7] Gadiel Seroussi. *Elliptic Curve Cryptography*, ITW 1999, Metsovo, Greece, June, 27th – July, 1st. Information Theory and Networking Workshop, 1999

[8] William Stallings. *Cryptography and Network Security – Principles and Practice*. 2nd Edition – Prentice Hall

[9] Reinhard Rauscher, Frank Bohnsack. *Results of na Elliptic-Curve-Approach for Use in Cryptosystems*. EUROMICRO Conference, 1999. Proceedings. 25th, 09/08/1999 -09/10/1999, 1999 Location: Milan , Italy pp 415-422 vol.2 1999

[10] J.J. Botes and W.T. Penzhorn. *Public-Key Cryptosystems Based on Elliptic Curves*. Communications and Signal Processing, 1993., Proceedings of the 1993 IEEE South African Symposium on , 6 Aug 1993

[11] Eduardo Wolski. *Sistemas Criptográficos Baseados em Curvas Elípticas* http::\www.unb.br – Professor do Departamento de Engenharia Elétrica – Universidade de Brasília - UNB

[12] M.O.Rabin. *Digitalized Signatures and Public-Key Functions as Intractable as Factorization*. MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[13] T.ElGamal. *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*. IEEE Transactions on Information Theory, Vol 31, pp. 469-472, 1985

[14] Elsayed Mohammed, A. E. Emarah and Kh. El-Shennawy. *Elliptic Curve Cryptosystems on Smart Cards*. 2001 IEEE 35th International Carnahan Conference on , Oct 2001 pp 213 -222

[15] Diffie, W.,L Hellman, M. "*New Directions on Cryptography*". IEEE Transactions on Information Theory, 1976

[16] www.certicom.com - "*ECC On Line Tutorial*"

[17] www.cryptoman.com/elliptic.htm - "*Elliptic Curve Cryptography FAQ v1.12*" – 22nd December, 1997.

[18] Neal Koblitz (Koblitz Et Al)., Alfred Menezes, Scott Vanstone - "*The State of Elliptic Curve Cryptography*" – http::/modular.fas.harvard.edu/ - 2000.

[19] Prof. Dr. Jean-Yves Chouinard - "*Notes on Elliptic Curve Cryptography*" – Design of Secure Computer Systems - http::/www.site.uottawa.ca/ - September, 24th, 2002.