Aritmética eficiente em curvas elípticas no modelo de Huff

Edson Floriano S. Junior[1], **Diego F. Aranha**[1]

[1]Universidade de Brasília (UnB) – Brasília, DF – Brasil
florianoefsj@gmail.com,dfaranha@unb.br

Resumo. *Neste trabalho, serão examinadas fórmulas explícitas para aritmética em curvas elípticas no modelo de Huff, com o objetivo de aprimorar o desempenho de sistemas criptográficos baseados nessas curvas. Propõem-se novas fórmulas para o cálculo de operações compostas que aceleram o laço principal da operação de multiplicação por escalar em até 11,8%. Demonstra-se também que o emprego das fórmulas mais rápidas com custos diferentes para cada operação consiste em uma abordagem mais eficiente para multiplicação de ponto do que utilizar fórmulas unificadas, conservando a segurança contra ataques de canal lateral.*

Abstract. *In this work, we examine explicit formulas for performing arithmetic in elliptic curves in the Huff model, with the objective of improving the performance of cryptographic systems based on such curves. We propose novel formulas for computing composite operations which speed up the main loop of the scalar multiplication operation in up to 11.8%. We also show that employing the fastest formulas with different costs for each operation consists in a more efficient approach for scalar multiplication than using unified formulas, while still conserving security against side-channel attacks.*

1. Introdução

As recentes notícias de casos de espionagem por parte de entidades governamentais mostram claramente o quão vulneráveis estão nossos meios de comunicação. Entidades externas são incentivadas política e economicamente a tomarem conhecimento de informações sigilosas e até a realizarem interferências ativas, manipulando informações vitais à continuidade de um negócio ou à soberania de uma nação. Por esse motivo, a necessidade de se implantar serviços de *confidencialidade* e *autenticação* continua crescente.

Criptografia assimétrica é um candidato natural para esta finalidade, por permitir soluções razoáveis para o problema da distribuição de chaves simétricas e fornecer assinaturas digitais irretratáveis, garantindo tanto *autenticação de origem* quanto de *integridade dos dados* [4]. No caso das assinaturas digitais, segundo [13], podem ser utilizados algoritmos baseados na dificuldade de fatoração de inteiros grandes, como o esquema de assinatura RSA [11], ou ainda, no Problema do Logaritmo Discreto (DLP), como o *Digital Signature Algorithm* (DSA) [10]. Existem ainda, os que se baseiam no DLP quando instanciado sobre *curvas elípticas* (ECDLP) como o ECDSA [5]. A grande vantagem de se utilizar curvas elípticas é depender de um problema subjacente de natureza exponencial, permitindo ganhos de desempenho e custo de armazenamento advindos do tamanho menor dos parâmetros e chaves criptográficas envolvidas.

Apesar da dificuldade intrínseca do problema subjacente em que se baseia a criptografia assimétrica, existem outras estratégias eficientes de ataque. Uma possibilidade é contornar o sistema criptográfico em si e explorar vulnerabilidades características de

sua *implementação*, monitorando o processamento em curso da primitiva criptográfica, visando obter *bits* da chave. Com vistas de proteger os sistemas deste tipo de ataque, um ramo de pesquisas bem difundido é o que busca a chamada *proteção contra vazamento de informação por canal lateral* [8], que consiste em padronizar o processamento das operações fundamentais do protocolo, de maneira a não fornecer informações a um atacante intrusivo. No contexto de curvas elípticas, isto implica em utilizar algoritmos *regulares* para a operação fundamental de multiplicação de ponto por escalar. Alternativas nesse sentido consistem em adotar curvas elípticas com fórmulas unificadas para duplicação e adição de pontos, como por exemplo os modelos de Edwards [2] e Huff [6].

Este artigo tem por objetivo apresentar novas fórmulas para aritmética em curvas de Huff que, em combinação com um algoritmo regular de multiplicação de pontos, conservam a proteção de canal lateral com uma melhora de desempenho de até 11,8% em relação a uma implementação com fórmulas unificadas que utiliza a recodificação w-NAF [12]. A seção 2 apresenta os pré-requisitos necessários para uma boa compreensão deste artigo, seguida pela seção 3 onde é apresentado o modelo de Huff para curvas elípticas e as fórmulas já exploradas por [6] em seu trabalho. A seção 4 traz efetivamente a contribuição deste artigo , ao passo que na seção 5 são discutidos os resultados e, finalmente, as conclusões na seção 6.

2. Preliminares

Para o devido entendimento do conteúdo deste artigo, recomenda-se ao leitor que esteja familiarizado com os conceitos de operações módulo m, grupo e corpo [3] e curvas elípticas [14]. Além destes conceitos, acrescentam-se os seguintes.

2.1. Curva de Weierstrass reduzida

Definição 1. Duas curvas E_1 e E_2 definidas sobre K, são ditas *isomorfas sobre* K se existe uma mudança admissível de variáveis que transforma a equação E_1 na equação E_2.

Através de uma mudança admissível de variáveis [4], uma *curva elíptica* E sobre um *corpo K* de característica diferente de 2 e 3, representada por $E(K)$ pode ser reduzida a uma equação da forma:

$$E(K) : y^2 = x^3 + ax + b \tag{1}$$

Esta equação é chamada *equação de Weierstrass*, a qual se acrescenta por conveniência um ponto extra, ∞, o ponto no infinito e onde $a, b \in K$ e $\Delta \neq 0$, sendo Δ o *discriminante* de *E* definido como $\Delta = -16(4a^3 + 27b^2)$.

2.2. Coordenadas projetivas

Seja K um corpo, e sejam c e d inteiros positivos. Segundo [4], pode-se definir uma relação de equivalência $\sim$ no conjunto $K^3\backslash\{0, 0, 0\}$ de triplas não-nulas sobre K por:

$$(X_1, Y_1, Z_1) \sim (X_2, Y_2, Z_2)$$

se $X_1 = \lambda^c X_2,\ Y_1 = \lambda^d Y_2,\ \ Z_1 = \lambda Z_2$ para algum $\lambda \in K^*$. A classe de equivalência que contém $(X, Y, Z) \in K^3\backslash\{0, 0, 0\}$ é

$$(X : Y : Z) = \{(\lambda^c X, \lambda^d Y, \lambda Z) : \lambda \in K^*\}. \tag{2}$$

$(X : Y : Z)$ é chamado de *ponto projetivo*, e (X, Y, Z) é chamado de representante da classe $(X : Y : Z)$. O conjunto de todos os pontos projetivos é denotado por $\mathbb{P}(K)$. Note-se que se $(X', Y', Z') \in (X : Y : Z)$ então $(X : Y : Z) = (X' : Y' : Z')$, ou seja, qualquer elemento de uma classe de equivalência pode servir como seu representante. Em particular, se $Z \neq 0$, então $(X/Z^c : Y/Z^d : 1)$ é o único representante com coordenada $Z = 1$. Isso implica que temos uma correspondência 1-1 entre o conjunto de pontos projetivos $\mathbb{P}(K)^* = \{(X : Y : Z) : X, Y, Z \in K, Z \neq 0\}$ e o conjunto de *pontos afins* $\mathbb{A}(K) = \{(x, y) : x, y \in K\}$. O conjunto de pontos $\mathbb{P}(K)^0 = \{(X : Y : Z) : X, Y, Z \in K, Z = 0\}$ é chamado de *linha no infinito* visto que seus pontos não correspondem a qualquer dos pontos afins. Neste trabalho serão utilizadas as coordenadas projetivas *homogêneas*, em que $c = d = 1$ e, portanto, ao ser mencionado um ponto $P' = (X : Y : Z)$ o seu correspondente afim $P = (x, y)$ pode ser obtido pela simples multiplicação das coordenadas X e Y pelo inverso de Z em K.

3. Modelo de Huff para curvas elípticas

Definição 2. Seja K um corpo de característica $\neq 2$. Considere o conjunto de pontos $(x, y) \in E(K)$ que satisfazem a equação

$$ax(y^2 - 1) = by(x^2 - 1), \tag{3}$$

ou em sua forma projetiva, $(X : Y : Z) \in E(K)$ satisfazendo a equação

$$aX(Y^2 - Z^2) = bY(X^2 - Z^2), \tag{4}$$

onde $a, b \in K^*$ e $a^2 \neq b^2$. Esta forma é chamada de *modelo de Huff* de uma curva elíptica.

A definição 1 que trata sobre isomorfismo de curvas implica que se duas curvas E_1 e E_2 são isomorfas sobre K, então os grupos $E_1(K)$ e $E_2(K)$ de seus pontos também são isomorfos [4], (cabe ressaltar que a recíproca nem sempre é verdadeira). Tendo estes conceitos em mente, conclui-se que toda curva elíptica sobre corpos finitos não-binários com um sub-grupo isomorfo a $\mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ pode ser transformada na forma de Huff [6].

3.1. Lei de Grupo numa curva de Huff

Valendo-se da notação geométrica, o trabalho [6] propõe uma lei de grupo para uma curva descrita no modelo de Huff. A linha tangente em $(0 : 0 : 1)$ é $aX = bY$, a qual intersecta a curva com multiplicidade 3, então $\mathcal{O} = (0 : 0 : 1)$ é um ponto de inflexão de E. A curva E, juntamente com a operação denotada por $\oplus$, formam, portanto, um grupo $(E, \oplus)$ com elemento neutro $\mathcal{O}$ e, cuja lei $\oplus$ tem a seguinte propriedade: para qualquer linha intersectando a curva E em 3 pontos (incluindo multiplicidades), temos que $P_1 \oplus P_2 \oplus P_3 = \mathcal{O}$. Em particular, o inverso do ponto $P_1 = (X_1 : Y_1 : Z_1)$ é $\ominus P_1 = (X_1 : Y_1 : -Z_1)$, ou seja, em coordenadas afins, o inverso de $P = (x, y)$ é o ponto $-P = (-x, -y)$, o que corresponde a reflexão do ponto em relação à origem $(0, 0)$ (que corresponde a $\mathcal{O}$). Então a soma de dois pontos P_1 e P_2 será $P_1 + P_2 = \ominus P_3$.

Um fato relevante a se ressaltar é que neste modelo, *elemento neutro* e *ponto no infinito* são coisas totalmente distintas. Por definição, o ponto no infinito (∞) como sendo seu próprio inverso. Portanto, os três pontos no infinito (isto é, com coordenada $Z = 0$), a

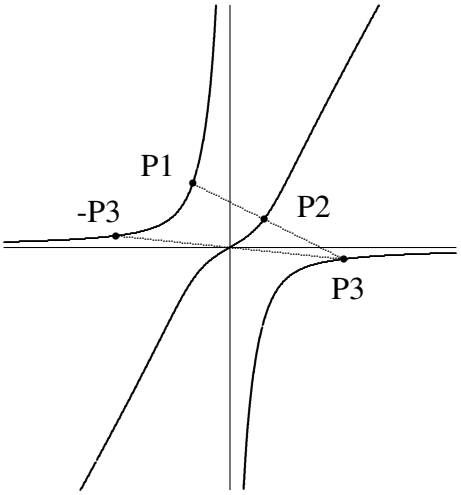

Figura 1. Curva de Huff: $11x(y^2-1) = 19y(x^2-1)$ **sobre** $\mathbb{R}$

saber: $(1:0:0)$, $(0:1:0)$ e $(a,b,0)$, são exatamente os três pontos primitivos de torção 2 de E. Isto quer dizer que a soma de quaisquer dois destes pontos é igual ao terceiro. Mais genericamente, $(X_1:Y_1:Z_1) \oplus (1:0:0)$ é o inverso do ponto de intersecção da linha horizontal passando através de $(X_1:Y_1:Z_1)$ com E. Quando $Z_1 \neq 0$, temos:

$$(X_1:Y_1:Z_1) \oplus (1:0:0) = (Z_1^2 : -X_1Y_1 : X_1Z_1), \tag{5}$$

e analogamente,

$$(X_1:Y_1:Z_1) \oplus (0:1:0) = (-X_1Y_1 : Z_1^2 : Y_1Z_1). \tag{6}$$

Como $(a:b:0) = (1:0:0) \oplus (0:1:0)$, quando $Z_1 \neq 0$, teremos $(X_1:Y_1:Z_1) \oplus (a:b:0) = (Z_1^2 : -X_1Y_1 : X_1Z_1) \oplus (0:1:0)$ e, portanto,

$$(X_1:Y_1:Z_1) \oplus (a:b:0) = \begin{cases} (a:b:0)\ se\ (X_1:Y_1:Z_1) = (0:0:1) \\ (Y_1Z_1 : X_1Z_1 : -X_1Y_1) \text{ caso contrário.} \end{cases} \tag{7}$$

Adicionar $(a:b:0)$ a qualquer dos quatro pontos $(\pm 1:\pm 1:0)$ transforma-o em seu inverso. Segue então que estes pontos são solução para a equação $2P = (a:b:0)$ e são pontos primitivos de torção 4. Estes oito pontos frisados acima, isto é, $\mathcal{O}$, os três pontos no infinito, e $(\pm 1:\pm 1:0)$, formam um subgrupo isomorfo a $\mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$.

A correspondência entre pontos da equação (4) e a equação de Weierstrass $V^2W = U(U+a^2W)(U+b^2W)$ é dada por:

$$\begin{aligned} \Upsilon \ &: \ \mathbb{P}^2(K) \to \mathbb{P}^2(K) : \\ &(X:Y:Z) \mapsto (U:V:W) = (ab(bX-aY) : ab(b^2-a^2)Z : -aX+bY) \end{aligned} \tag{8}$$

$$\begin{aligned} \Upsilon^{-1} \ &: \ \mathbb{P}^2(K) \to \mathbb{P}^2(K) : \\ &(U:V:W) \mapsto (X:Y:Z) = (b(U+a^2W) : a(U+b^2W) : V). \end{aligned} \tag{9}$$

Oberve que o ponto no infinito $(0:1:0)$ na curva de Weierstrass é mapeado para $(0:0:1)$ na curva de Huff pela função Υ^{-1}.

3.2. Fórmulas afins

O trabalho [6] traz ainda fórmulas para a lei de grupo. Excluindo os pontos no infinito e utilizando a forma $ax(y^2-1) = by(x^2-1)$, seja $y = \lambda x + \mu$ a linha secante passando através de dois pontos diferentes $P_1 = (x_1, y_1)$ e $P_2 = (x_2, y_2)$. Esta linha intersecta a curva em um terceito ponto $\ominus P_3 = (-x_3, -y_3)$. Substituindo a equação da reta na equação da curva, temos

$$ax((\lambda x+\mu)^2-1) = b(\lambda x+\mu)(x^2-1) \Rightarrow \lambda(a\lambda-b)x^3 + \mu(2a\lambda-b)x^2 + \ldots = 0$$

e quando definida, obtém-se

$$\begin{cases} x_3 = x_1 + x_2 + \frac{\mu(2a\lambda-b)}{\lambda(a\lambda-b)} \\ y_3 = \lambda x_3 - \mu, \end{cases} \tag{10}$$

onde $\lambda = \frac{(y_1-y_2)}{x_1-x_2}$ e $\mu = y_1 - \lambda x_1$. Após algumas manipulações, obtém-se

$$\begin{cases} x_3 = \frac{(x_1+x_2)(1+y_1y_2)}{(1+x_1x_2)(1-y_1y_2)} \\ y_3 = \frac{(y_1+y_2)(1+x_1x_2)}{(1-x_1x_2)(1+y_1y_2)}. \end{cases} \tag{11}$$

As fórmulas (11) estão definidas sempre que $x_1x_2 \neq \pm 1$ e $y_1y_2 \neq \pm 1$ e, ao contrário das fórmulas (10), podem ser usadas tanto para somar quanto para duplicar pontos.

3.3. Fórmulas projetivas

As formulas apresentadas anteriormente envolvem inversões em um corpo K, o que representa alto custo de desempenho computacional. Para evitar esta operação, são utilizadas coordenadas projetivas. As operações de multiplicação e quadrado serão denotadas por m e s, respectivamente, no corpo K. Substituindo x e y nas fórmulas (11) pelas suas correspondentes coordenadas projetivas (X/Z) e (Y/Z) e eliminando os denominadores, temos como resultado as seguintes fórmulas:

$$\begin{cases} X_3 = (X_1Z_2 + X_2Z_1)(Y_1Y_2 + Z_1Z_2)^2(Z_1Z_2 - X_1X_2) \\ Y_3 = (Y_1Z_2 + Y_2Z_1)(X_1X_2 + Z_1Z_2)^2(Z_1Z_2 - Y_1Y_2) \\ Z_3 = (Z_1^2Z_2^2 - X_1^2X_2^2)(Z_1^2Z_2^2 - Y_1^2Y_2^2) \end{cases} \tag{12}$$

que podem ser computadas com custo $\underline{12m}$. O custo pode ainda ser reduzido para $\underline{11m}$ no caso em que um dos pontos está em coordenadas afins, ou seja, $Z_2 = 1$.

3.4. Duplicação dedicada

Quando se deseja executar um grande número de duplicações, arranjos podem ser feitos de modo a obter fórmulas específicas para melhor desempenho, como mostrado abaixo:

$$\begin{cases} X_3 = (2X_1Z_1)(Y_1^2 + Z_1^2)^2(Z_1^2 - X_1^2) \\ Y_3 = (2Y_1Z_1)(X_1^2 + Z_1^2)^2(Z_1^2 - Y_1^2) \\ Z_3 = (Z_1^4 - X_1^4)(Z_1^4 - Y_1^4) \end{cases} \tag{13}$$

que podem ser computadas com $\underline{10m + 1s}$. Quando a relação do custo de duplicação e multiplicação no corpo é $s < \frac{3}{4}m$, o cálculo de $2P$ pode ser computado com $\underline{7m + 5s}$ da seguinte forma:

$$\mathbf{s_1} = X_1^2,\ \mathbf{s_2} = Y_1^2,\ \mathbf{s_2} = Z_1^3,\ \mathbf{A} = (s_3 + s_1),\ \mathbf{B} = (s_3 - s_1),\ \mathbf{C} = (s_3 + s_2),\ \mathbf{D} = (s_3 - s_2),$$
$$\mathbf{s_4} = (X_1 + Z_1)^2 - A,\ \mathbf{s_5} = (Y_1 + Z_1)^2 - C,\ \mathbf{m_1} = s_4C,\ \mathbf{m_2} = s_5A,\ \mathbf{m_3} = CD,\ \mathbf{m_4} = AD,$$
$$\mathbf{X_3} = m_1m_3,\ \mathbf{Y_3} = m_2m_4,\ \mathbf{Z_3} = m_3m_4.$$

3.5. Movendo a origem

Segundo [6] pode-se escolher $\mathcal{O}' = (0 : 1 : 0)$ como elemento neutro, fazendo uma pequena mudança na lei de grupo. Denotando-se por $\oplus'$ a adição correspondente de pontos, então $P_1 \oplus' P_2 = (P_1 \ominus \mathcal{O}') \oplus (P_2 \ominus \mathcal{O}') \oplus \mathcal{O}' = P_1 \oplus P_2 \oplus \mathcal{O}'$. Então temos:

$$\begin{cases} X_3 = (X_1Z_2 + X_2Z_1)(Y_1Y_2 + Z_1Z_2)(Y_1Z_2 + Y_2Z_1) \\ Y_3 = (X_1X_2 - Z_1Z_2)(Z_1^2Z_2^2 - Y_1^2Y_2^2) \\ Z_3 = (Y_1Z_2 + Y_2Z_1)(X_1X_2 + Z_1Z_2)(Y_1Y_2 - Z_1Z_2), \end{cases} \tag{14}$$

que pode ser calculada com $\underline{11m}$. Para o caso em que $Z_2 = 1$ o número de multiplicações cai para $\underline{10m}$. Para duplicação dedicada, a execução requer $\underline{6m + 5s}$. Deve-se notar também que o oposto $\ominus' P_1$ de um ponto P_1 permanece inalterado pois $\ominus' P_1 = \ominus(P_1 \ominus \mathcal{O}') \oplus \mathcal{O}' = \ominus P = (X_1 : Y_1 : -Z_1)$.

3.6. Multiplicação de ponto por escalar

As fórmulas dadas acima são utilizadas para calcular kP, onde k é um inteiro e P um ponto na curva elíptica E, definida sobre K. Esta operação é chamada multiplicação por escalar e domina a maior parte do tempo de execução de um sistema criptográfico baseado em curvas elípticas. O Algoritmo 1 calcula kP e processa os *bits* de k da esquerda para a direita. Como pode ser visto no laço principal do algoritmo, ao processar um *bit* $k_i = 1$ de um inteiro k, serão executadas as operações $Q \leftarrow 2Q$ e $Q \leftarrow Q + P$ consecutivamente, ao passo que se o *bit* for $k_i = 0$ apenas $Q \leftarrow 2Q$ é executada. Esta diferença no processamento de *bits* 0 ou 1 é a base para os chamados ataques de canal lateral, em que um adversário que monitora o processamento do computador utiliza informações de tempo de execução de porções do algoritmo para diferenciar iterações que executam apenas duplicações ou duplicações e adições de ponto, o que permite ao atacante inferir com alta probabilidade a parcela da chave para aquela iteração [8].

Algoritmo 1 Método binário de multiplicação de pontos da esquerda para a direita.

Entrada: $k = (k_{l-1}, \ldots, k_1, k_0)_2, P \in E(\mathbb{F}_q)$
Saida: kP.

1: $Q \leftarrow \infty$.
2: **for** $i \leftarrow t - 1$ **downto** 0 **do**
3: $\quad Q \leftarrow 2Q$.
4: $\quad$ **if** $k_i = 1$ **then** $Q \leftarrow Q + P$
5: **end for**
6: **return** (Q).

Duas abordagens são as mais comuns para se implementar a operação de multiplicação de ponto por escalar em tempo constante, de maneira a subverter ataques de canal lateral: utilizar fórmulas unificadas ou um algoritmo de multiplicação regular. Ambas as abordagens são compatíveis com algoritmos de multiplicação de ponto por escalar que processam o escalar com uma janela de comprimento w. A utilização das fórmulas unificadas (12) fornece a vantagem de que as operações de duplicação e adição de pontos, executadas no laço principal do algoritmo de multiplicação, passam a ter estritamente o mesmo custo computacional. Do ponto de vista de desempenho, fórmulas unificadas possibilitam a utilização do algoritmo de multiplicação de ponto com recodificação mais eficiente do escalar e menor densidade possível, o que termina por reduzir o número de

adições e favorecer o desempenho. O Algoritmo 2 é um exemplo de algoritmo eficiente para multiplicação de ponto que pode fazer uso de fórmulas unificadas para tornar o tempo de execução constante. A recodificação w-NAF [9] produz escalares com no máximo um dígito a mais que o escalar original e densidade aproximadamente $\frac{1}{w+1}$. Por outro lado, algoritmos regulares para multiplicação de ponto recodificam o escalar de maneira previsível, com algum custo computacional pelo aumento da densidade e número de adições resultante. A recodificação [7], adaptada da recodificação w-NAF, é um exemplo de algoritmo regular com densidade $\frac{1}{w-1}$, portanto maior que o Algoritmo 2.

Algoritmo 2 Método w-NAF para multiplicação de ponto.

Entrada: $k \in \mathbb{Z}, P \in E(\mathbb{F}_p)$.
Saída: $kP \in E(\mathbb{F}_p)$.

1: Calcular a representação $NAF_w(k) = \sum_{i=0}^{l-1} k_i 2^i$
2: Calcular $P_i = iP$, para $u \in \{1, 3, 5, \ldots, 2^{w-1} - 1\}$
3: $Q \leftarrow \infty$
4: **for** $i \leftarrow l - 1$ **downto** 0 **do**
5: $\quad Q \leftarrow 2Q$
6: $\quad$ **if** $k_i \neq 0$ **then**
7: $\quad\quad$ **if** $u_i > 0$ **then** $Q \leftarrow Q + P_{k_i}$; **else** $Q \leftarrow Q - P_{k_i}$
8: $\quad$ **end if**
9: **end for**
10: **return** Q

4. Contribuições

A seguir, são apresentadas as fórmulas e sequências de operações desenvolvidas, visando a aceleração de operações compostas para utilização em algoritmos de multiplicação regular. Como visto no Algoritmo 1, ao processar um bit $k_i = 1$ de um inteiro k, serão executadas as operações $Q \leftarrow 2Q$ e $Q \leftarrow Q + P$ consecutivamente. Pode-se então obter uma significativa melhora no tempo de execução destas operações transformando-as em uma só, do tipo $Q \leftarrow 2Q + P$, aplicando as coordenadas obtidas das fórmulas (13) nas formulas (12). Considerando $P_1 = Q$ e $P_2 = P$, obtém-se as fórmulas:

$$\begin{cases} X_3 = (Z_2C + X_2A)(Z_2A - X_2C)(Z_2B + Y_2D)^2 \\ Y_3 = (Z_2D + Y_2B)(Z_2B - Y_2D)(Z_2A + X_2C)^2 \\ Z_3 = (Z_2A + X_2C)(Z_2A - X_2C)(Z_2B - Y_2D)(Z_2B + Y_2D), \end{cases} \tag{15}$$

onde $A = (Z_1^2 + X_1^2)(Z_1^2 - Y_1^2), B = (Z_1^2 + Y_1^2)(Z_1^2 - X_1^2), C = (2X_1Z_1)(Z_1^2 + Y_1^2), D = (2Y_1Z_1)(Z_1^2 + X_1^2)$, que pode ser calculado com $\underline{17m + 5s}$ como segue:

$$\mathbf{s_1} = X_1^2,\ \mathbf{s_2} = Y_1^2\ \mathbf{s_3} = Z_1^2,\ \mathbf{T_1} = (s_3 + s_1),\ \mathbf{T_2} = (s_3 - s_1),\ \mathbf{T_3} = (s_3 + s_2),\ \mathbf{T_4} = (s_3 - s_2),$$
$$\mathbf{s_4} = (X_1 + Z_1)^2 - T_1,\ \mathbf{s_5} = (Y_1 + Z_1)^2 - T_3,\ \mathbf{A} = T_4T_1,\ \mathbf{B} = T_3T_2,\ \mathbf{C} = s_4T_3,\ \mathbf{D} = s_5T_1,$$
$$\mathbf{m_5} = Z_2A,\ \mathbf{m_6} = X_2C,\ \mathbf{m_7} = Z_2B,\ \mathbf{m_8} = Y_2D,\ \mathbf{T_1} = (m_5 - m_6),\ \mathbf{T_2} = (m_5 + m_6),$$
$$\mathbf{T_3} = (m_7 - m_8),\ \mathbf{T_4} = (m_7 + m_8),\ \mathbf{m_9} = (A + C)(X_2 + Z_2) - T_2,$$
$$\mathbf{m_{10}} = (B + D)(Y_2 + Z_2) - T_4,\ \mathbf{m_{11}} = T_1T_4,\ \mathbf{m_{12}} = T_2T_3,\ \mathbf{m_{13}} = m_9T_4,\ \mathbf{m_{14}} = m_{10}T_2,$$
$$\mathbf{X_3} = m_{11}m_{13},\quad \mathbf{Y_3} = m_{12}m_{14},\quad \mathbf{Z_3} = m_{11}m_{12}.$$

No caso em que P_2 é dado em coordenadas afins ($Z_2 = 1$), este cálculo se resume a $\underline{15m + 5s}$, eliminando-se as multiplicações m_5 e m_7 acima. Utilizando $\mathcal{O} = (0 : 1 : 0)$

como elemento neutro, as fórmulas definidas em 3.5 e procedendo de maneira análoga ao procedimento exposto acima, obtém-se as seguintes fórmulas para $P_3 = 2P_1 + P_2$:

$$\begin{cases} X_3 = (AZ_2 + BX_2)(CZ_2 - DY_2)(CY_2 - DZ_2) \\ Y_3 = (AX_2 - BZ_2)(CZ_2 - DY_2)(CZ_2 + DY_2) \\ Z_3 = -(AX_2 + BZ_2)(CY_2 - DZ_2)(CZ_2 + DY_2), \end{cases} \tag{16}$$

onde $A = (2X_1Z_1)(Z_1^2 + Y_1^2), B = (Z_1^2 + X_1^2)(Y_1^2 - Z_1^2), C = (2Y_1Z_1)(Z_1^2 + X_1^2), D = = (Z_1^2 + Y_1^2)(X_1^2 - Z_1^2)$, que requerem $\underline{16m + 5s}$ da seguinte forma:

$$\mathbf{s_1} = X_1^2,\ \mathbf{s_2} = Y_1^2,\ \mathbf{s_3} = Z_1^2,\ \mathbf{T_1} = (s_1 + s_3),\ \mathbf{T_2} = (s_1 - s_3),\ \mathbf{T_3} = (s_2 + s_3),\ \mathbf{T_4} = (s_2 - s_3),$$
$$\mathbf{s_4} = (X_1 + Z_1)^2 - T_1,\ \mathbf{s_5} = (Y_1 + Z_1)^2 - T_3,\ \mathbf{A} = s_4T_3,\ \mathbf{B} = T_1T_4,\ \mathbf{C} = s_5T_1,\ \mathbf{D} = T_3T_2$$
$$\mathbf{m_5} = AX_2,\ \mathbf{m_6} = BZ_2,\ \mathbf{m_7} = CZ_2,\ \mathbf{m_8} = DY_2,\ \mathbf{T_1} = (m_5 - m_6),$$
$$\mathbf{T_2} = (m_5 + m_5),\ \ \mathbf{T_3} = (m_7 - m_8),\ \ \mathbf{T_4} = (m_7 + m_8),$$
$$\mathbf{m_9} = (A + B)(X_2 + Z_2) - T_2,\ \ \mathbf{m_{10}} = (C + D)(Y_2 + Z_2) - T_4,$$
$$\mathbf{X_3} = m_9m_{10}T_3,\ \ \mathbf{Y_3} = T_1T_3T_4,\ \ \mathbf{Z_3} = -m_{10}T_2T_4.$$

Novamente, quando $Z_2 = 1$, P_3 pode ser calculado com $\underline{14m + 5s}$ como segue:

$$\mathbf{s_1} = X_1^2,\ \mathbf{s_2} = Y_1^2,\ \mathbf{s_3} = Z_1^2,\ \mathbf{T_1} = (s_1 + s_3),\ \mathbf{T_2} = (s_1 - s_3),\ \mathbf{T_3} = (s_2 + s_3),\ \mathbf{T_4} = (s_2 - s_3),$$
$$\mathbf{s_4} = (X_1 + Z_1)^2 - T_1,\ \mathbf{s_5} = (Y_1 + Z_1)^2 - T_3,\ \mathbf{A} = s_4T_3,\ \mathbf{B} = T_1T_4,\ \mathbf{C} = s_5T_1,\ \mathbf{D} = T_3T_2,$$
$$\mathbf{m_5} = AX_2,\ \ \mathbf{m_6} = BX_2,\ \ \mathbf{m_7} = CY_2,\ \ \mathbf{m_8} = DY_2,\ \mathbf{T_1} = (A + m_6),\ \mathbf{T_2} = (C - m_8),$$
$$\mathbf{T_3} = (m_7 - D),\ \mathbf{T_4} = (m_5 - B),\ \ \mathbf{T_5} = (C + m_8),\ \ \mathbf{T_6} = (m_5 + B),$$
$$\mathbf{X_3} = T_1T_2T_3,\ \ \mathbf{Y_3} = T_2T_4T_5,\ \ \mathbf{Z_3} = -T_3T_5T_6.$$

Em situações em que se necessite calcular vários pontos de um grupo, pode ser útil um conjunto de fórmulas que forneça como saída ambos $2P_1 + P_2$ e $2P_1 - P_2$ sem um grande custo adicional. Com esta finalidade, são expostas abaixo fórmulas que calculam $2P_1 \pm P_2$ com custo adicional de $\underline{4m}$ em relação ao conjunto de fórmulas (16) utilizando $\mathcal{O} = (0 : 1 : 0)$ como elemento neutro e $Z_2 = 1$.

$$2P_1 + P_2 = \begin{cases} X_3 = (A + BX_2)(C - DY_2)(CY_2 - D) \\ Y_3 = (AX_2 - B)(C - DY_2)(C + DY_2) \\ Z_3 = -(AX_2 + B)(CY_2 - D)(C + DY_2); e \end{cases}$$

$$2P_1 - P_2 = \begin{cases} X_4 = (A + BX_2)(C + DY_2)(CY_2 + D) \\ Y_4 = (AX_2 - B)(C - DY_2)(C + DY_2) \\ Z_4 = -(AX_2 + B)(CY_2 - D)(C + DY_2), \end{cases} \tag{17}$$

onde $A = (2X_1Z_1)(Z_1^2 + Y_1^2), B = (Z_1^2 + X_1^2)(Y_1^2 - Z_1^2), C = (2Y_1Z_1)(Z_1^2 + X_1^2), D = = (Z_1^2 + Y_1^2)(X_1^2 - Z_1^2)$, que requerem $\underline{18m + 5s}$ da seguinte forma:

$$\mathbf{s_1} = X_1^2,\ \mathbf{s_2} = Y_1^2,\ \mathbf{s_3} = Z_1^2,\ \mathbf{T_1} = (s_1 + s_3),\ \mathbf{T_2} = (s_1 - s_3),\ \mathbf{T_3} = (s_2 + s_3),\ \mathbf{T_4} = (s_2 - s_3),$$
$$\mathbf{s_4} = (X_1 + Z_1)^2 - T_1,\ \mathbf{s_5} = (Y_1 + Z_1)^2 - T_3,\ \mathbf{A} = s_4T_3\ \mathbf{B} = T_1T_4\ \mathbf{C} = s_5T_1\ \mathbf{D} = T_3T_2,$$
$$\mathbf{m_5} = AX_2,\ \mathbf{m_6} = BX_2,\ \mathbf{m_7} = CY_2,\ \mathbf{m_8} = DY_2,\ \mathbf{T_1} = (A + m_6),\ \mathbf{T_2} = (C - m_8),$$
$$\mathbf{T_3} = (m_7 - D),\ \mathbf{T_4} = (m_5 - B),\ \mathbf{T_5} = (C + m_8),\ \mathbf{T_6} = (m_5 + B),\ \mathbf{m_9} = T_2T_4,\ \mathbf{m_{10}} = T_5T_6,$$
$$\mathbf{X_3} = T_1T_2T_3,\ \mathbf{Y_3} = m_9T_5,\ \mathbf{Z_3} = -T_3m_{10},\ \mathbf{X_4} = T_5T_7T_8,\ \mathbf{Y_4} = m_{10}T_2,\ \mathbf{Z_4} = m_9T_8.$$

5. Resultados

Considerando as duas abordagens para proteção do canal lateral descritas na Seção 3.6, é possível estimar o impacto das fórmulas e determinar qual a mais eficiente em ambos

os casos. A precomputação inicial executada por ambos os algoritmos exige o cálculo de diversos múltiplos do ponto P, que pode ser realizado utilizando a abordagem convencional com uma duplicação e adições sucessivas; ou as fórmulas para duplicação combinada com adição e subtração de pontos (DAS) a partir do cálculo do poto $3P$, obtido a partir da fórmula de duplicação e adição de pontos (DA). Por exemplo, os pontos $5P$ e $7P$ podem ser obtidos a partir da fórmula $2 \cdot (3P) \pm P$. A simples utilização do algoritmo w-NAF para multiplicação de um ponto por um escalar com l *bits* requer l duplicações de ponto (D) e $\frac{l}{w+1}$ adições de ponto (A), todas efetuadas com fórmulas unificadas (U). A utilização de um algoritmo regular conserva o custo de precomputação, mas permite a utilização das fórmulas mais eficientes para duplicação e adição de pontos. A Tabela 1 apresenta o custo computacional das duas abordagens para multiplicação por escalar. Em ambos os casos, o processamento da chave da esquerda para a direita permite utilizar adições em coordenadas misturadas.

Tabela 1. Custo computacional de duas abordagens para multiplicação de ponto por um escalar de comprimento l protegida contra vazamento de informação por canal lateral. O custo é dividido nas etapas de precomputacão inicial e laço principal utilizando o escalar recodificado.

Algoritmo	**Custo computacional**
Precomputação convencional	$D + (2^{w-2} - 1)A$
Precomputação composta	$DA + (2^{w-3} - 1)DAS$
Recodificação w-NAF	$lU + \frac{l}{w+1}U$
Recodificação regular	$(l - \frac{l}{w-1})D + \frac{l}{w-1}DA$

Para o nível de segurança de 128 *bits* e considerando os custos $U = 11m$, $D = 6m + 5s$, $A = 10m$, $DA = 14m + 5s$ e $DAS = 18m + 5s$, a Tabela 2 estima os custos computacionais das duas abordagens. Considerando três casos reportados na literatura como resultados de implementação de aritmética em um corpo primo, temos $s = m$ [1], $s = 0.85m$ [4] e $s = \frac{2}{3}m$ [1], onde o último é obtido quando a curva é instanciada sobre uma extensão quadrática $\mathbb{F}_{p^2}$, pode-se comparar o custo computacional correspondente à melhor escolha de w nas duas abordagens. Primeiramente, a precomputação utilizando as fórmulas compostas é mais eficiente que a precomputação original apenas para o caso $w = 4, s = \frac{2}{3}m$. Felizmente, os ganhos são mais representativos na execução do laço principal. Para $s = m$, a abordagem regular é 1,1% menos eficiente que w-NAF. Para $s = 0.85m$, a abordagem regular é 4,5% mais eficiente que w-NAF. Entretanto, para $s = \frac{2}{3}m$, a abordagem regular implementada com fórmulas compostas é 11,8% mais eficiente que w-NAF. Desta forma, as fórmulas compostas desenvolvidas oferecem ganho de desempenho quando a curva é instanciada sobre uma extensão quadrática.

Tabela 2. Custo computacional de duas abordagens para multiplicação de ponto por escalar no nível de segurança de 128 *bits* para duas escolhas de w e protegida contra vazamento de informação por canal lateral.

Algoritmo	$w = 4$	$w = 5$
Precomputação convencional	$36m + 5s$	$76m + 5s$
Precomputação composta	$32m + 10s$	$68m + 20s$
Recodificação w-NAF	$3388m$	$3289m$
Recodificação regular	$2224m + 1280s$	$2048m + 1280s$

6. Conclusão

A operação de multiplicação de ponto por escalar é crítica para o desempenho e segurança de sistemas criptográficos baseados no logaritmo discreto em curvas elípticas. Neste trabalho, foi apresentado um conjunto de fórmulas para operações compostas em curvas elípticas que possuem potencial para acelerar a aritmética em curvas elípticas representadas no modelo de Huff e, consequentemente, a operação de multiplicação de ponto por escalar sem, no entanto, comprometer a segurança contra ataques de canal lateral. Considerando diversas razões entre implementações da multiplicação e quadrado em corpos finitos, estimamos o ganho de desempenho no laço principal do algoritmo como no máximo 11,8%, exatamente quando a curva elíptica é definifa sobre a extensão quadrática de um corpo primo. Curiosamente, observamos que a utilização de fórmulas unificadas não consiste na abordagem mais eficiente para implementação segura contra ataques de canal lateral de sistemas criptográficos baseados em curvas elípticas.

Referências

[1] D. F. Aranha, K. Karabina, P. Longa, C. H. Gebotys, and J. López. Faster Explicit Formulas for Computing Pairings over Ordinary Curves. In *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 48–68. Springer, 2010.

[2] Daniel J. Bernstein and Tanja Lange. Faster addition and doubling on elliptic curves. In *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 29–50. Springer, 2007.

[3] H. H. Domingues and G. Iezzi. *Álgebra Moderna*. Atual, 1995.

[4] D. Hankerson, A. J. Menezes, and S. Vanstone. *Guide to Eliptic Curve Cryptography*. Springer-Verlag, 2004.

[5] D. Johnson, A. Menezes, and S. A. Vanstone. The Elliptic Curve Digital Signature Algorithm (ECDSA). *Int. J. Inf. Sec.*, 1(1):36–63, 2001.

[6] M. Joye, M. Tibouchi, and D. Vergnaud. Huff's model for elliptic curves. In *ANTS-IX*, volume 6197 of *LNCS*, pages 234–250. Springer, 2010.

[7] M. Joye and M. Tunstall. Exponent recoding and regular exponentiation algorithms. In *AFRICACRYPT 2009*, volume 5580 of *LNCS*, pages 334–349. Springer, 2009.

[8] Paul C. Kocher. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In *CRYPTO 1996*, volume 1109 of *LNCS*, pages 104–113. Springer, 1996.

[9] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Criptography*. CRC, 1996.

[10] NIST. Digital Signature Standard. FIPS Publication 186, 1994.

[11] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[12] J. A. Solinas. Efficient Arithmetic on Koblitz Curves. *Designs, Codes and Cryptography*, 19(2-3):195–249, 2000.

[13] D. R. Stinson. *Cryptography - Theory and pratice*. CRC, 2006.

[14] L. C. Washington. *Elliptic curves: number theory and cryptography*. CRC, 2008.