Análise de Seleção de Parâmetros em Criptografia Baseada em Curvas Elípticas

Rosemberg André da Silva

Dissertação de Mestrado Profissional

Instituto de Computação
Universidade Estadual de Campinas

Análise de Seleção de Parâmetros em Criptografia Baseada em Curvas Elípticas

Rosemberg André da Silva

Julho de 2006

Banca Examinadora:

- Ricardo Dahab
  DTC/IC/UNICAMP (Orientador)
- Marco Aurélio do Amaral Henriques
  DCA/FEEC/UNICAMP
- Julio C. López Hernández
  DTC/IC/UNICAMP
- Paulo Licio de Geus
  DSC/IC/UNICAMP (suplente)

Análise de Seleção de Parâmetros em Criptografia Baseada em Curvas Elípticas

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Rosemberg André da Silva e aprovada pela Banca Examinadora.

Campinas, 28 de Julho de 2006.

Ricardo Dahab
DTC/IC/UNICAMP (Orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do Mestrado Profissional em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Júlia Milani Rodrigues – CRB8a / 2116

Silva, Rosemberg André
Si38a Análise de seleção de parâmetros em criptografia baseada em curvas elípticas / Rosemberg André Silva -- Campinas, [S.P. :s.n.], 2006.

Orientador : Ricardo Dahab
Trabalho final (mestrado profissional) - Universidade Estadual de Campinas, Instituto de Computação.

1. Criptografia. 2. Curvas elípticas. 3. Proteção de dados. I. Dahab, Ricardo. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Título em inglês: Parameter selection analysis on elliptic curve cryptography.

Palavras-chave em inglês (Keywords): 1. Cryptography. 2. Elliptic curves. 3. Data protection.

Área de concentração: Engenharia de Software

Titulação: Mestre em Ciência da Computação

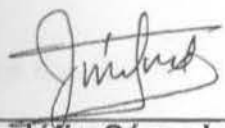
Banca examinadora: Prof. Dr. Ricardo Dahab (IC-UNICAMP)
Prof. Dr. Marco Aurélio do Amaral Henriques (FEEC-UNICAMP)
Prof. Dr. Julio Cesar López Hernández (IC-UNICAMP)
Prof. Dr. Paulo Licio de Geus (IC-UNICAMP)

Data da defesa: 28/07/2006

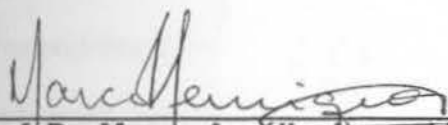
Programa de Pós-Graduação: Mestrado em Ciência da Computação

TERMO DE APROVAÇÃO

Trabalho Final Escrito defendido e aprovado em 28 de julho de 2006, pela Banca Examinadora composta pelos Professores Doutores:



Prof. Dr. Júlio César Lopez Hernándes
IC - UNICAMP



Prof. Dr. Marco Aurélio Amaral Henriquez
FEEC - UNICAMP

Prof. Dr. Ricardo Dahab
IC - UNICAMP

Substitua pela folha com a assinatura da banca

Resumo

A escolha dos parâmetros sobre os quais uma dada implementação de Criptografia sobre Curvas Elípticas baseia-se tem influência direta sobre o desempenho das operações associadas bem como sobre seu grau de segurança. Este trabalho visa analisar a forma como os padrões mais usados na atulalidade lidam com este processo de seleção, mostrando as implicações que tais escolhas acarretam.

Abstract

The choice of parameters associated with a given implementation of ECC (Elliptic Curve Cryptography) has direct impact on its performance and security level. This dissertation aims to compare the most common standards used now-a-days, taking into account their selection criteria and their implications on performance and security.

Agradecimentos

À minha família, por seu amor e apoio incondicionais.
A meu orientador Ricardo Dahab, pela paciência e ânimo na condução deste trabalho.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

1.1 Histórico sobe Criptografia

Criptografia é o ramo da Teoria da Informação que lida com questões relativas à segurança na transmissão e proteção de dados, fazendo uso do mapeamento do conteúdo original da informação num conteúdo cifrado (criptograma) e vice-versa.

Os primeiros registros de seu uso têm alguns milhares de anos: hieróglifos codificados datando de aproximadamente 4500 A.C. Textos judaicos aplicando substituição de letras do alfabeto por outras (600 A.C., método Atbash) ilustram a utilização da criptografia com fins religiosos. Idéia semelhante foi aplicada pelos romanos, que utilizavam adição com redução de módulo para mapear caracteres da mensagem original em caracteres da mensagem cifrada. No século XVI este método foi melhorado por Vigenère, que propôs o uso de blocos de caracteres cada qual utilizando um deslocamento diferente.

A partir do século XIX começam a surgir abordagens mais formais à criptografia e à cripto-análise (quebra de mensagens cifradas), como, por exemplo, através dos trabalhos de Charles Babbage e Friedrich Kasiski. O uso de métodos matemáticos vai sofisticando-se através das duas primeiras guerras mundiais. Os trabalhos de William Friedman e Marian Rejewski destacam-se neste período, marcado pelo uso de máquinas mecânicas e eletromecânicas (como o Enigma, de fabricação alemã) na troca de mensagens cifradas.

Até meados da década de 1970 a transmissão de mensagem fazia uso exclusivamente de chaves privadas para cifrar e decifrar conteúdos. A cifragem e a decifragem eram feitas apenas por quem possuísse tais chaves (ou quem conseguisse quebrar o esquema criptográfico utilizado). Em 1976 uma mudança radical de paradigma ocorreu quando Diffie e Hellman [26] propuseram o uso de chaves públicas. Um ano depois foi criado o RSA, primeiro sistema de criptografia de chaves públicas, aplicando as idéias de Ron Rivest, Adi Shamir e Len Adleman [57]. O problema matemático que garante a segurança de seus protocolos é a fatoração de números inteiros.

Em 1985, Neal Koblitz [29] e Victor Miller [45] propuseram independetemente a Criptografia sobre Curvas Elípticas (ECC). A idéia era prover as mesmas funcionalidades que os esquemas de RSA, mas substituindo o problema matemático. Ao invés de fatoração de números inteiros, passou-se a empregar o problema de logaritmo discreto sobre curvas elípticas (ECDLP). Pelo fato deste problema ser de mais difícil resolução, exigindo o uso de algoritmos que têm tempo de execução exponencial, as chaves utilizadas podem ser menores que as de RSA para o mesmo nível de segurança. Uma chave de 160 bits baseada em curvas elípticas tem aproximadamente o mesmo nível de segurança que uma chave de 1024 bits de RSA.

1.2 Objetivos de Segurança da Criptografia

Num mundo em que transações comerciais e financeiras feitas por meio eletrônico são efetuadas com freqüência crescente, faz-se necessário proteger o conteúdo das informações trocadas entre diferentes entidades e garantir que as partes envolvidas na comunicação possam ser identificadas e autenticadas.

Suponhamos um cenário simplificado em que duas partes A e B trocam mensagens entre si sobre um canal não-seguro. Seja uma terceira parte E tentando ler ou modificar tais mensagens, e eventualmente fazer-se passar por A ou B. Os sistemas criptográficos tentam proteger a comunicação entre A e B através das seguintes garantias:

- Sigilo: apenas as partes autorizadas A e B podem ter acesso aos conteúdos originais das mensagens trocadas. Muito embora E tenha acesso ao conteúdo cifrado (criptogramas), o esforço computacional requerido para extrair o conteúdo original a partir deles deve ser tão grande ou caro que não compense ser executado.

- Integridade de Dados: deve-se assegurar que o conteúdo original das mensagens trocadas entre A e B não tenha sido alterado por E. Caso alguma alteração tenha ocorrido, ela deve ser detectável por A ou B.

- Autenticação da Origem: caso A tenha transmitido a mensagem, o sistema deve prover meios para que B possa assegurar-se de que realmente foi A quem a enviou.

- Autenticação de Entidade: a identidade das partes envolvidas na troca de mensagens deve ser verificável de alguma forma, ou seja, A deve ter alguma garantia de que está comunicando-se com B, e vice-versa.

- Irrevogabilidade: uma vez que uma das partes tenha transmitido uma mensagem, o sistema deve garantir que ela possa ser vinculada a seu transmissor. Assim, este não conseguirá defender com sucesso a afirmação de que não é a fonte de tal mensagem.

1.3 Criptografia com Chaves Públicas

Tomando como critério a forma como as chaves de criptografia são empregadas, os sistemas criptográficos podem ser divididos em duas grandes categorias: simétricos e assimétricos.

Os primeiros são tais que as entidades envolvidas na comunicação devem receber chaves secretas e autenticadas a serem usadas nas operações de criptografia. Para cifragem com chaves simétricas pode-se fazer uso, por exemplo, de métodos como DES, RC4 e AES. Adicionalmente, pode-se utilizar MAC para autenticação de mensagem e HMAC para verificação de sua integridade. Apesar de serem mais eficientes que os sistemas assimétricos, possuem desvantagens sérias com relação a distribuição e gerenciamento de chaves.

Para que as chaves simétricas possam ser distribuídas, é necessário que se faça uso de um canal de comunicação secreto e autenticado. Algumas soluções adotadas consistem em utilizar canal físico confiável (como entrega pessoal) ou de um provedor de serviços que estabeleça chaves secretas com todas as entidades de uma rede e posteriormente use tais chaves para distribuição de uma segunda chave secreta, que será usada entre as partes que queiram se comunicar. Soluções como esta última requerem a existência de uma autoridade central confiável, e é pouco prática em aplicações como correio eletrônico através da internet.

Ao problema de distribuição soma-se o de gerenciamento de chaves. Cada entidade de uma rede de comunicação utilizando este tipo de criptografia precisa, potencialmente, de chaves diferentes para todos os seus pares. O emprego de um servidor central responsável pela administração das chaves contribui para diminuir os impactos deste problema, dado que as entidades não mais precisarão armazenar e gerenciar tais chaves. A irrevogabilidade não é trivialmente resolvida pelos esquemas simétricos, dado que, pelo fato de transmissor e receptor usarem as mesmas chaves, não ser direto determinar qual entidade gerou uma determinada mensagem.

Os sistemas assimétricos, por sua vez, fazem uso de pares de chaves: cada entidade tem associada a si uma chave pública e uma chave privada relacionadas matematicamente. O canal de distribuição de chaves não precisa ser secreto, mas apenas autenticado. As chaves públicas são usadas na cifragem, e as privadas na decifragem. Além disso, o mecanismo de assinatura é de mais fácil implementação pelo fato de cada entidade ter seu próprio par de chaves. Assim sendo, a assinatura pode ser associada diretamente a este par: assina-se utilizando a chave privada, e verifica-se a assinatura por meio da chave pública. Os três maiores problemas observados em sistemas simétricos (distribuição de chave, irrevogabilidade e assinatura) são, portanto, resolvidos de forma relativamente simples pelos sistemas assimétricos.

Dado um par de chaves assimétricas associado a um usuário, a segurança dos protocolos de criptografia reside no fato de que, para que alguém consiga derivar a chave privada a partir da pública, um problema matemático intratável tenha que ser resolvido. Desta forma, RSA baseia-se na fatoração de inteiros; ElGamal/DSA no logaritmo discreto; Criptografia sobre curvas elípticas, no problema de logaritmo discreto sobre grupos gerados pela curva.

1.3.1 RSA

Este sistema é resultado do trabalho de Rivest, Shamir, Adleman [57] e provê esquemas para geração de chave, cifragem, decifragem e assinatura.

- **Geração de chaves**: a partir de um parâmetro de segurança l , deriva uma chave pública (n, e) e uma chave privada d tais que $ed \equiv 1 \pmod{n}$.
- **Cifragem**: dados a chave pública (n, e) e o texto $m \in [0, n-1]$, obtem-se a mensagem cifrada $c = m^e \bmod n$.
- **Decifragem**: dada a chave privada d , a chave pública (n, e) e a mensagem cifrada c , obtem-se $c^d = (m^e)^d = m^{ed} = m \pmod{n}$.
- **Assinatura**: dadas a chave privada d , a chave pública (n, e) e a mensagem m , calcula-se $s = (\mathrm{Hash}(m))^d \bmod n$.
- **Verificação da assinatura**: dadas a chave pública (n, e) , a mensagem m e a assintura s , é verificado se $s^e \bmod n$ coincide com $\mathrm{Hash}(m)$.

A segurança destes esquemas reside no pressuposto de que, dados (n, e), é tão difícil obter d quanto resolver o problema de fatoração de número inteiro.

1.3.2 Logaritmo Discreto

Este sistema é baseado nos trabalhos de Diffie e Hellman [26] sobre protocolo para obtençao de chaves, e de ElGamal [15] sobre cifragem, decifragem e assinatura.

- **Geração de chaves**: dados os parâmetros de domínio (p, q, g) , onde p é um número primo, $g \in [1, p-1]$ com ordem q , obtem-se as chaves pública y e privada x tais que $y = g^x \bmod p$.

- **Cifragem**: dados os parâmetros de domínio (p, q, g) , a chave pública y e a mensagem $m \in [0, p-1]$, obtém-se $c_1 = g^k \bmod p$ e $c_2 = m.y^k \bmod p$, para $k \in [1, q-1]$ arbitrariamente escolhido.

- **Decifragem**: dados os parâmetros de domínio (p, q, g) , a chave privada x e o texto cifrado (c_1, c_2) , obtém-se $c_2.c_1^{-x} = m.y^k.g^{-kx} = m.g^{kx}.g^{-kx} = m.g^0 = m \,(\bmod p)$.

- **Assinatura**: dados os parâmetros de domínio (p, q, g) , a chave privada x e a mensagem m , obtém-se $r = (g^k \bmod p) \bmod q$ e $s = k^{-1}(\mathrm{Hash}(m) + xr) \bmod q$ para $k \in [1, q-1]$.

- **Verificação da assinatura**: Dados os parâmetros de domínio (p, q, g) , a chave pública y , a mensagem m e a assinatura (r, s) , verifica-se que $(g^{u_1} y^{u_2} \bmod p) \bmod q$ coincide com r, sendo que $u_1 = \mathrm{Hash}(m).s^{-1} \bmod q$, e $u_2 = r.s^{-1} \bmod q$.

A segurança dos esquemas acima reside no fato de que, dados (p, q, g) e y, é difícil calcular $x \in [1, q-1]$ tal que $y = g^x \bmod p$. Tal cálculo corresponde ao problema do logaritmo discreto.

1.3.3 Criptografia sobre Curvas Elípticas (ECC)

Este sistema é resultado dos trabalhos independentes de Koblitz [29] e Miller [45].

- **Geração de chaves**: dados os parâmetros de domínio (p, E, P, n) , onde P é um ponto sobre a curva elíptica E definida no corpo F_p, tal que o subgrupo gerado por P tenha ordem n , obtem-se a chave pública Q e a chave privada $d \in [1, n-1]$ tais que $Q = dP$.

- **Cifragem**: dados os parâmetros de domínio (p, E, P, n) , a chave pública Q e a mensagem m, obtém-se $C_1 = kP$ e $C_2 = M + kP$, onde $k \in [1, n-1]$ e M corresponde a uma representação de m em $E(F_p)$. Este foi um dos primeiros métodos empregados. Outros mais sofisticados são utilizados atualmente, conforme pode ser visto nas especificações citadas por esta dissertação.

- **Decifragem**: dados os parâmetros de domínio (p, E, P, n) , a chave privada d e o texto cifrado (C_1, C_2), obtem-se $M = C_2 - dC_1$ e extrai-se m a partir de M. Este foi um dos primeiros métodos empregados. Outros mais sofisticados são utilizados atualmente, conforme pode ser visto nas especificações citadas por esta dissertação.

- **Assinatura**: dados os parâmetros de domínio, uma chave privada d e uma mensagem m, o processo de assinatura resulta na obtenção dois elementos (r, s) do corpo finito associado aos parâmetros tais que $s = k^{-1}(e + dr) \bmod n$, onde n representa o número de elementos do corpo finito, $k \in [1, n-1]$ e $e = \text{Hash}(m)$.

- **Verificação de Assinatura**: para verificar (r, s), faz-se uso da chave pública Q do transmissor e do ponto-base P listado nos parâmetros de domínio. Calcula-se $A = es^{-1}P + rs^{-1}Q$. A coordenada x de A deve coincidir com r para que a assinatura seja válida.

A segurança deste sistema reside no fato de não se conhecerem algoritmos sub-exponenciais que resolvam o problema do logaritmo discreto sobre curvas elípticas, onde, dados os parâmetros de domínio (p, E, P, n) e $Q \in E(F_p)$, deve-se obter d tal que $Q = dP$. Não se conhece nenhum algoritmo que resolva este problema em tempo sub-exponencial. A dificuldade em se resolver este problema permite que chaves menores em ECC consigam o mesmo grau de segurança que chaves maiores em RSA. Como citado na seção 1.1, uma chave de 160 bits em ECC resulta no mesmo grau de segurança que uma de 1024 bits em RSA. Sistemas que tenham limitações de banda e memória podem ser bastante beneficiados com o uso de chaves menores. A tabela 1.1, construída a partir de dados extraídos da especificação SEC 2 [9], ilustra a vantagem de ECC sobre os demais sistemas criptográficos (RSA e logaritmo discreto) tomando tamanho de chaves como parâmetro.

A especificação X9.62 ilustra através da tabela 1.2 o tempo necessário para que o melhor algoritmo de propósito geral conhecido (Pollard ρ) resolva o problema de logaritmo discreto sobre curvas elípticas, supondo que uma máquina de 1 MIPS consiga executar 40000 adições sobre uma curva elíptica por segundo.

1.4 Objetivo da Dissertação

Esta dissertação faz uma compilação de importantes informações sobre o estado da arte na aplicação de curvas elípticas a criptografia, ressaltando como a escolha de parâmetros de domínio afeta as características de desempenho e segurança dos sistemas de criptografia baseados em tais curvas. Foco especial será dado às escolhas de corpos e curvas, utilizando-se resultados de estudos do meio acadêmico e das especificações e padrões adotados pela indústria, de forma a endereçar os ataques mais comumente encontrados.

1.5 Organização Deste Documento

Este documento foi dividido em 8 capítulos divididos da seguinte forma:

- Capítulo 1 - Introdução: Expõe o objetivo deste trabalho e a forma como ele está estruturado.
- Capítulo 2 - Definições de Curvas Elípticas: São dadas referências às definicões de grupos, corpos, curvas elípticas, representações, ECDL. Aritmética sobre Curvas Elípticas: Adição, multiplicação, transformacões lineares, mapas.
- Capítulo 3 - Escolha de Parâmetros para Criptografia sobre Curvas Elípticas: Escolha do corpo, curva e parâmetros da curva, graus de liberdade na escolha (consórcio SECG).
- Capítulo 4 - Teorema de Hasse para Curvas Elípticas: Ordem de uma curva elíptica.
- Capítulo 5 - Curvas Elípticas e Hiper-elípticas: Problema do logaritmo discreto sobre os dois tipos duas curvas.
- Capítulo 6 - Compromisso entre Segurança e Desempenho: Como a escolha de parâmetros pode afetar estes aspectos, número de pontos da curva, ataques.
- Capítulo 7 - Padrões em Criptografia sobre Curvas Elípticas : Padrões consolidados - P1363, ANSI X9.62, ANSI X9.63, WAP WTLS, FIPS 186.2.
- Capítulo 8 - Conclusões : consolidação das escolhas de parâmetros para ECC, com apreciações sobre desempenho e segurança.

Neste capítulo foi exposta a finalidade deste trabalho e a maneira em que foi dividido. No capítulo seguinte serão apresentados conceitos necessários à compreensão de criptografia sobre curvas elípticas.

Tabela 1.1: Comparações entre tamanhos de chaves

Nível de Segurança (bits)	Chave Simétrica (bits)	Chave ECC (bits)	Chave DSA/RSA(bits)
56	56	112	512
80	80	160	1024
112	112	224	2048
128	128	256	3072
192	192	384	7680
256	256	521	15360

Tabela 1.2: Complexidade do ECDLP

Tamanho em bits da coordenada n	Complexidade do Algoritmo $(\sqrt{\frac{\pi n}{4}})$	Computação (MIPS ano)
162	2^{80}	1.1×10^{12}
224	2^{111}	2.3×10^{21}
256	2^{127}	1.5×10^{26}
384	2^{191}	2.9×10^{45}
521	2^{259}	8.4×10^{65}

Capítulo 2

Curvas Elípticas: Definições e Operações

Os esquemas de criptografia sobre curvas elípticas analisados neste documento envolvem operações aritméticas numa curva elíptica ou no corpo finito sobre o qual está definida. O presente capítulo tem por objetivo fazer uma introdução dos conceitos necessários ao entendimento e aplicação de tais operações.

2.1 Grupos

Grupo G corresponde a um conjunto C e uma operação binária **op** definida sobre os elementos deste conjunto de tal forma que, dados $a, b, c \in C$:

- a operação **op** seja fechada sobre o conjunto C: $a \,\mathbf{op}\, b \in C$
- a operação **op** seja associativa: $a \,\mathbf{op}\, (b \,\mathbf{op}\, c) = (a \,\mathbf{op}\, b) \,\mathbf{op}\, c$
- exista um elemento neutro (identidade aditiva) $\mathcal{O} \in C$ tal que $a \,\mathbf{op}\, \mathcal{O} = \mathcal{O} \,\mathbf{op}\, a = a$
- para cada elemento $a \in C$ exista um elemento inverso $\text{inv}(a)$ tal que $a \,\mathbf{op}\, \text{inv}(a) = \text{inv}(a) \,\mathbf{op}\, a = \mathcal{O}$

Além disso, o grupo será considerado abeliano, se:

- a operação **op** for comutativa: $a \,\mathbf{op}\, b = b \,\mathbf{op}\, a$

2.2 Corpos

Um corpo consiste de um conjunto $\mathbb{F}$ com operações de adição (denotada por $+$) e multiplicação (denotada por $\cdot$) de tal forma que:

- $(\mathbb{F}, +)$ seja um grupo abeliano com elemento identidade denotado por 0;
- $(\mathbb{F} \setminus \{0\}, .)$ seja um grupo abeliano com elemento identidade denotado por 1;
- Distributividade seja válida: $(a + b).c = a.c + b.c$ para todos $a, b, c \in \mathbb{F}$.

Se o conjunto $\mathbb{F}$ definido acima for finito corpo é dito finito.

Além disso, as operações de subtração e divisão podem ser definidas utilizando os inversos aditivo e multiplicativo, respectivamente.

Desta forma, dados $a, b \in \mathbb{F}$:

- subtração (denotada por -): $a - b = a + (-b)$, onde $-b$ é o inverso aditivo de b;
- divisão (denotada por /): $a/b = a.b^{-1}$, onde b^{-1} é o inverso multiplicativo de $b \neq 0$.

O seguinte resultado da teoria de corpos finitos estabelece que o número de elementos do conjunto a partir do qual o corpo é definido denomina-se ordem do corpo.

Teorema 1 Existe um corpo finito $\mathbb{F}$ de ordem q , se, e somente se, q for uma potência de número primo (isto é, $q = p^m$, com p primo, chamado característica de $\mathbb{F}$, e $m \geq 1 \in \mathbb{Z}$).

2.2.1 Corpos Primos

Para a definição de corpo finito dada acima, se $m = 1$ o corpo é dito primo. O conjunto de inteiros dados por $\{0, 1, ..., p - 1\}$ juntamente com as operações de adição e multiplicação módulo p constituem o corpo primo $\mathbb{F}_p$. Além disso, p é chamado módulo de $\mathbb{F}_p$, e para fazer redução módulo p de qualquer inteiro a é necessário tomar o resto da divisão inteira de a por p.

Importante: exceto quando explicitamente dito o contrário, os corpos primos mencionados neste documento terão característica maior que 3. Isto deve-se ao fato de que as equações de curva elípticas sobre tais corpos serem diferentes das dos demais primos. Como os corpos de interesse em criptografia normalmente são de característica muito maior que três, a restrição será seguida.

2.2.2 Corpos Binários

Também chamados finitos de característica 2, são denotados por $\mathbb{F}_{2^m}$. Seus elementos podem ser considerados como polinômios binários de grau menor ou igual a $m-1$ com coeficientes em $\mathbb{F}_2$:

$$\mathbb{F}_{2^m} = \{a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_2x^2 + a_1x + a_0\} \, \text{com} \, a_i \in \{0,1\}$$

A operação de adição dos elementos deste corpo consiste na adição usual de polinômios, sendo que a aritmética aplicada aos coeficientes é de módulo 2. A operação de multiplicação, no entanto, requer redução polinomial de módulo. Para tanto, faz-se uso de um polinômio binário irredutível $f(x)$ de grau m. A redução de um dado polinômio $a(x)$ consiste em tomar-se o resto da divisão polinomial de $a(x)$ por $f(x)$. Por razões de desempenho, quanto menor for o número de termos não-nulos de $f(x)$ tanto mais eficiente será a operação de redução. Trinômios (e, na ausência destes para um dado m, pentanômios) são as melhores escolhas conhecidas.

Nota: corpos da forma $\mathbb{F}_{p^m}$, com $p \neq 2$ não serão considerados neste documento, posto que não são utilizados pelos padrões considerados no estudo.

2.2.2.1 Base Polinomial

Os elementos de $\mathbb{F}_{2^m}$ podem ser vistos como vetores binários de dimensão m sobre $\mathbb{F}_2$. Uma base polinomial é denotada por $\{x^{m-1}, x^{m-2}, ..., x^2, x, 1\}$. Cada elemento de $\mathbb{F}_{2^m}$, portanto, corresponde a um polinômio de grau menor que m : $a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1x + a_0$, com coeficientes $a_i \in \{0,1\}$. Tal elemento pode ser denotado como o vetor de bits $(a_{m-1}...a_1a_0)$.

Adição: A adição de dois elementos $(a_{m-1}...a_1a_0)$ e $(b_{m-1}...b_1b_0)$ resulta em $(c_{m-1}...c_1c_0)$, onde $c_i = a_i \oplus b_i$ e $\oplus$ denota a operação ou-exclusivo.

Multiplicação: Seja o polinômio irredutível $f(x) = x^m + f_{m-1}x^{m-1} + ... + f_1x + f_0$, com $f_i \in \mathbb{F}_2$ para $i = 0, ..., m-1$ e $f_0 = 1$. O produto de dois elementos $(a_{m-1}...a_1a_0)$ e $(b_{m-1}...b_1b_0)$ resulta em $(r_{m-1}...r_1r_0)$, onde o polinômio $r_{m-1}x^{m-1}+...+r_1x+x_0$ corresponde ao resto da divisão de $(a_{m-1}x^{m-1}+...+a_1x+a_0).(b_{m-1}x^{m-1}+...+b_1x+b_0)$ pelo polinômio $f(x)$ sobre $\mathbb{F}_2$.

2.2.2.2 Base Normal

Tem a forma $\{\alpha, \alpha^2, \alpha^{2^2}, ..., \alpha^{2^{m-1}}\}$ para um dado $\alpha \in \mathbb{F}_{2^m}$. Tais bases sempre existem para $m \geq 1$. Este tipo de base é bastante útil em implementações em hardware. A

operação de elevar um dado elemento do corpo ao quadrado consiste simplesmente em fazer um deslocamento circular de bits. A implementação de multiplicadores seriais de bits, como descrito por Massey e Omura [51] é grandemente simplificada.

A medida de complexidade em hardware deste multiplicador é uma função da base escolhida. Seu valor é mínimo para a base normal ótima, cuja caracterização pode ser encontrada em Onyszchuck [52]. Tais bases são denominadas bases gaussianas ótimas e existem sempre que m não for divisível por 8. Seu tipo é dado por um inteiro positivo que indica quão simples resulta a operação de multiplicação sobre tal base.

2.3 Curvas Elípticas.

2.3.1 Equações da Curva

Corpo Primo $\mathbb{F}_p$

Através de mudanças admissíveis de variável, a equação da curva elíptica pode ser posta da forma

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

onde $a, b \in F_p$, e tem discriminante dado por $\Delta = -16(4a^3 + 27b^2)$. Para que a curva não seja singular, tal discriminante deve ser diferente de 0 (mod p).

Corpo Binário $\mathbb{F}_{2^m}$

Através de mudanças admissíveis de variável, a equação assume a forma

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

onde $a, b \in \mathbb{F}_{2^m}$, e tem discriminante dado por $\Delta = b$. Para que a curva não seja singular, tal discriminante deve ser diferente de 0.

2.3.2 Representação de Pontos

As equações dadas para as curvas elípticas no item anterior foram baseadas em coordenadas afins. Caso as operações de inversão sobre os corpos finitos onde as equações foram definidas sejam significativamente mais caras que as operações de multiplicação, pode ser vantajoso o uso de outros sistema de coordenadas que minimize o número de tais operações.

Coordenadas Projetivas: Seja K um corpo e c, d inteiros positivos. Pode-se definir uma relação de equivalência $\sim$ no conjunto $K^3\backslash\{(0,0,0)\}$ de trios definidos 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 contendo $(X, Y, Z) \in K^3\backslash\{(0,0,0)\}$ é:

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

$(X : Y : Z)$ é chamado ponto projetivo e (X, Y, Z) é chamado de representante de $(X : Y : Z)$. O conjunto de todos os pontos é denominado $P(K)$. Se $(X', Y', Z') \in (X : Y : Z)$ então $(X' : Y' : Z') = (X : Y : Z)$, significando que qualquer elemento de uma equivalência pode servir como seu representante.

Como exemplo, $Z \neq 0$, $(X/Z^c, Y/Z^d, 1)$ é o único representante com $Z = 1$ do ponto projetivo $(X : Y : Z)$. Tem-se, desta forma, uma representação 1-1 entre o cojunto de pontos projetivos $P(K)^* = \{(X : Y : Z) : X, Y, Z \in K,\ Z \neq 0\}$ e o conjunto de pontos afins $A(K) = \{(x, y) : x, y \in K\}$.

Substituindo-se x por X/Z^c e y por Y/Z^d nas equações em coordenadas afins das curvas elípticas e eliminando-se os denominadores, tem-se como resultado equações da curva elíptica na forma projetiva.

Coordenadas Projetivas com Corpo Primo:

- Coordenadas projetivas padrão: $c = 1$, $d = 1$, $Z \neq 0$, correspondendo ao ponto afim $(X/Z, Y/Z)$.
- Coordenadas jacobianas projetivas: $c = 2$, $d = 3$, $Z \neq 0$, correspondendo ao ponto afim $(X/Z^2, Y/Z^3)$.
- Coordenadas Chudnovski: o ponto jacobiano $(X : Y : Z)$ é representado como $(X : Y : Z : Z^2 : Z^3)$.

Os diferentes tipos de coordenadas podem ser combinados nos algoritmos de adição e multiplicação em curvas elípticas objetivando melhoria de desempenho. Para os algoritmos listados em [43], capítulo 3, temos para a curva $y^2 = x^3 - 3x + b$ (A =coord. afins; P =coord. projetiva padrão; J =jacobiana; C =Chudnoviski; I =operação de inversão no corpo; M= operação de multiplicação no corpo; Q= operação de elevar ao quadrado no corpo) os custos mostrado nas tabelas 2.1, 2.2 e 2.3 .

Coordenadas Projetivas com Corpo Binário:

- Coordenadas projetivas padrão: $c = 1$, $d = 1$, $Z \neq 0$, correspondendo ao ponto afim $(X/Z, Y/Z)$.
- Coordenadas jacobianas projetivas: $c = 2$, $d = 3$, $Z \neq 0$, correspondendo ao ponto afim $(X/Z^2, Y/Z^3)$.
- Coordenadas López-Dahab: $c = 1$, $d = 2$, $Z \neq 0$, correspondendo ao ponto afim $(X/Z, Y/Z^2)$.

Considerando a curva $y^2 + xy = x^3 + ax^2 + b$, temos os resultados comparativos (M= multiplicação sobre o corpo; D= divisão sobre o corpo) na tebela 2.4.

2.4 Adição de Pontos

Seja E uma curva elíptica definida sobre o corpo F_q. Os pontos da curva e a operação de adição formam um grupo abeliano, com o ponto ∞ servindo como elemento identidade.

Uma propriedade interessante em relação à operação de adição para curvas elípticas definidas sobre o corpo dos reais é que, geometricamente, ela consiste em traçar uma corda pelos dois pontos sendo adicionados e refletir, em relação ao eixo x , o terceiro ponto de intersecção entre tal corda e a curva elíptica. Para somar um ponto a si próprio, traça-se uma tangente pelo ponto e reflete-se a segunda intersecção de tal tangente com a curva em relação ao eixo x para obter o resultado. Como exemplo, a figura 2.1 ilustra tal propriedade para a curva de equação $y^2 = x^3 - 8x + 6$.

2.4.1 Corpo Primo $\mathbb{F}_p$

Seja E uma curva elíptica definida sobre o corpo $\mathbb{F}_p$, onde p é primo, dada por $y^2 = x^3 + ax + b$. Sejam dois pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$, com $P \neq \pm Q$. Então, $P + Q = (x_3, y_3)$, onde

$x_3 = (\frac{y_2 - y_1}{x_2 - x_1})^2 - x_1 - x_2$ e $y_3 = (\frac{y_2 - y_1}{x_2 - x_1})(x_1 - x_3) - y_1$

Como exemplo, a figura 2.2 ilustra uma adição de pontos sobre a curva de equação $y^2 = x^3 + 6x + 13$ definida sobre F_{23}. O aspecto geométrico da adição visto sobre o corpo de reais não mais se verifica aqui.

Para o caso em que $P = Q$, $P + Q = P + P = 2P$. Se, $P \neq -P$, temos:

$x_3 = (\frac{3x_1 + a}{2y_1}) - 2x_1$ e $y_3 = (\frac{3x_1^2 + a}{2y_1})(x_1 - x_3) - y_1$

Para este tipo de corpo, também temos: $-P = (x_1, -y_1)$

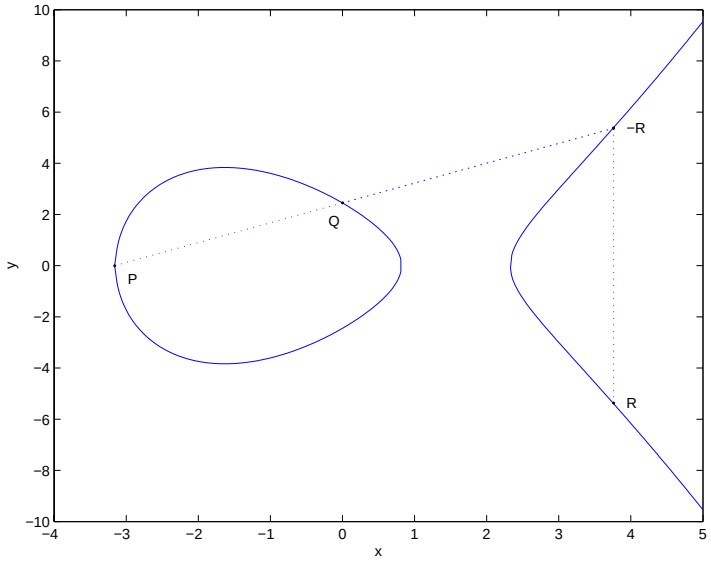


Figura 2.1: Adição de pontos em curva definida sobre o corpo dos números reais

2.4.2 Corpo Binário $\mathbb{F}_{2^m}$

Seja E uma curva elíptica definida sobre o corpo $\mathbb{F}_{2^m}$, com $m \geq 1$ inteiro, dada por $y^2 + xy = x^3 + ax^2 + b$. Sejam dois pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$, com $P \neq \pm Q$. Então, considerando $\lambda = (y_1 + y_2)/(x_1 + x_2)$, $P + Q = (x_3, y_3)$, onde

$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$ e $y_3 = \lambda(x_1 + x_3) + x_3 + y_1$

Como exemplo, a figura 2.3 ilustra uma adição de pontos sobre a curva de equação $y^2 + xy = x^3 + g^{13}x^2 + g^{10}$ definida sobre F_{2^4}. O aspecto geométrico da adição visto sobre o corpo de reais também não se verifica aqui. Os valores utilizados na construção da figura 2.3 encontram-se listados na tabela 2.5

Para o caso em que $P = Q$, $P + Q = P + P = 2P$. Se $P \neq -P$, temos:

$x_3 = \lambda^2 + \lambda + a$ e $y_3 = x_1^2 + \lambda x_3 + x_3$, com $\lambda = x_1 + y_1/x_1$

Para este tipo de corpo, temos: $-P = (x_1, x_1 + y_1)$

2.5 Multiplicação de Pontos

Seja um ponto P sobre uma curva elíptica E , e m um número inteiro. A operação de multiplicação de m por P consistem em obter o resultado da soma de m termos:

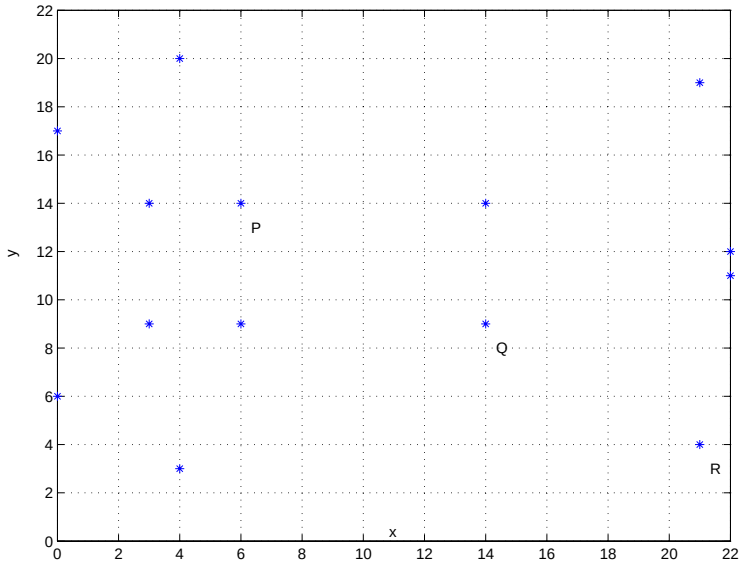


Figura 2.2: Adição de pontos em curva definida sobre um corpo finito primo

$[m]P = P + P + ... + P$.

O menor valor positivo de m para o qual a multiplicação resulta em ∞ é denominado ordem de P.

2.6 ECDL - Elliptic Curve Discrete Logarithm

O problema do logaritmo discreto sobre curvas elípticas (ECDL) é o problema em cuja dificuldade de resolução baseia-se a segurança dos protocolos de criptografia que usam este tipo de curva. Ele consiste em, dada a curva E e dois pontos P, Q sobre ela relacionados da forma $P = lQ$, encontrar o valor l (com l menor que a ordem de P).

2.7 Compressão de Pontos

Entre os parâmetros de domínio utilizados nas implementações de ECC constam os coeficientes da curva elíptica, suficientes para defini-la completamente. Quando da manipulação de pontos da curva nos diferentes protocolos suportados por ECC, pode-se fazer uso de uma representação compacta dos pontos. Isto pode resultar numa economia considerável de banda/memória, dado que o comprimento do ponto em tal representação é bastante diminuído. Para tanto, faz-se uso do fato de que a equação da curva elíptica

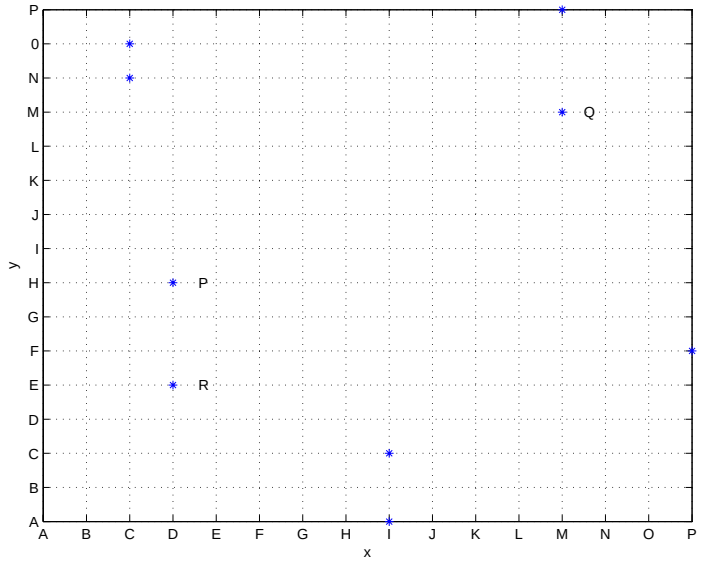


Figura 2.3: Adição de pontos em curva definida sobre um corpo finito binário

é fornecida como parte dos parâmetros de domínio. Como temos em mãos o valor da coordenada-x do ponto, faz-se necessário tão somente indicar qual dos dois valores da coordenada-y que satisfazem a equação está associado o ponto.

A especificação SEC 2 [9] utiliza compressão de ponto de acordo com o seguinte algoritmo:

ALGORITMO: Compressão de Pontos

ENTRADA: Ponto P , corpo F_q , coeficientes da curva
SAÍDA: String M de octetos representando o ponto
1. Se $P = 0$, $M \leftarrow 00_{16}$
2. Se $P \neq 0$
2.1. Converter x_p num string X de comprimento $\lceil (\log q)/8 \rceil$
2.2. Se o corpo for primo, $\tilde{y_p} \leftarrow y_p \pmod 2$
2.3. Se o corpo for binário e $x_p = 0$, $\tilde{y_p} \leftarrow 0$
2.4. Se o corpo for binário ($\mathbb{F}_{2^m}$) e $x_p \neq 0$, calcular $z = z_{m-1}x^{m-1} + ... + z_1 x + z_0$ tal que $z = y_p x_p^{-1}$ e fazer $\tilde{y_p} \leftarrow z_0$
2.5. Se $\tilde{y_p} = 0$ fazer string $Y = 02_{16}$, senão $Y = 03_{16}$
2.6. Concatenar X a Y e atribuir a M
3. Sair com M.

2.8 Isomorfismos

Dadas as equações de duas curva elípticas E_1 e E_2 definidas sobre o corpo primo $\mathbb{F}_p$:

$$E_1 : y^2 = x^3 + ax + b$$

$$E_2 : y^2 = x^3 + \bar{a}x + \bar{b}$$

as cuvas são ditas isomórficas sobre $\mathbb{F}_p$ se existir $u \in \mathbb{F}_p$, com $u \neq 0$, tal que a mudança de variável $(x, y) \rightarrow (u^2x, u^3y)$ transforme a equação E_1 em E_2.

Dadas as equações de duas curva elípticas E_3 e E_4 definidas sobre o corpo primo $\mathbb{F}_{2^m}$:

$$E_3 : y^2 + xy = x^3 + ax^2 + b$$

$$E_4 : y^2 + xy = x^3 + \bar{a}x^2 + \bar{b}$$

as cuvas são ditas isomórficas sobre $\mathbb{F}_{2^m}$ se $b = \bar{b}$ e $\mathrm{Tr}(a)$=$\mathrm{Tr}(\bar{a})$, onde $\mathrm{Tr}(x)$=$x + x^2 + x^{2^2} + ... + x^{2^{m-1}}$, com $x \in \mathbb{F}_{2^m}$. Nestas condições, existe $s \in \mathbb{F}_{2^m}$, com $\bar{a} = s^2 + s + a$ tal que a mudança de variável $(x, y) \rightarrow (x, y + sx)$ transforme a equação E_3 em E_4.

A relação de isomorfismo é uma relação de equivalência no conjunto de curvas elípticas definidas sobre um corpo K. Se duas curvas E_1 e E_2 sobre K forem isomórficas, os grupos correspondentes aos pontos gerados por tais curvas também o serão. É possivel, desta forma, constuir um mapeamento reversível dos elementos de um grupo nos do outro.

Neste capítulo foi mostrada a definição de curva elíptica, bem como as operações que sobre ela podem ser efetuadas visando utilização no escopo de criptografia. A seguir será discutido como fazer a escolha de parâmetros para definir a aplicação de tais curvas à criptografia.

Tabela 2.1: Duplicação de Ponto

$2A \rightarrow A$	$1I, 2M, 2Q$
$2P \rightarrow P$	$7M, 3Q$
$2J \rightarrow J$	$4M, 4Q$
$2C \rightarrow C$	$5M, 4Q$

Tabela 2.2: Adição

$A + A \rightarrow A$	$1I, 2M, 1Q$
$P + P \rightarrow P$	$12M, 2Q$
$J + J \rightarrow J$	$12M, 4Q$
$C + C \rightarrow C$	$11M, 3Q$

Tabela 2.3: Adição / Coordenadas Mistas

$J + A \rightarrow J$	$8M, 3Q$
$J + C \rightarrow J$	$11M, 3Q$
$C + A \rightarrow C$	$8M, 3Q$

Tabela 2.4: Coordenadas Projetivas - Custo de Operações

Coordenadas	Adição	Duplicação de um Ponto
Afim	$1M, 1D$	$1M, 1D$
Projetiva Padrão	$13M$	$7M$
Jacobiana	$14M$	$5M$
López-Dahab	$14M$	$4M$

Tabela 2.5: Corpo $\mathbb{F}_{2^4}$

coordenada-x	representação	valor
A	1	(0 0 0 1)
B	g	(0 0 1 0)
C	g^2	(0 1 0 0)
D	g^3	(1 0 0 0)
E	g^4	(0 0 1 1)
F	g^5	(0 1 1 0)
G	g^6	(1 1 0 0)
H	g^7	(1 0 1 1)
I	g^8	(0 1 0 1)
J	g^9	(1 0 1 0)
K	g^{10}	(0 1 1 1)
L	g^{11}	(1 1 1 0)
M	g^{12}	(1 1 1 1)
N	g^{13}	(1 1 0 1)
O	g^{14}	(1 0 0 1)
P	0	(0 0 0 0)

Capítulo 3

Escolha de Parâmetros para Criptografia sobre Curvas Elípticas

Os diferentes padrões de criptografia sobre curvas elípticas definem conjuntos de parâmetros que as implementações que alegam conformidade possam utilizar. Em geral, tais conjuntos são denominados "parâmetros de domínio" e definem, entre outras coisas, o corpo e a curva elíptica a serem utilizados na implementação de protocolos. Neste capítulo será mostrado como obter desepenho e segurança através da seleção criteriosa de parâmetros.

As opções de corpo analisadas aqui são $\mathbb{F}_p$ e $\mathbb{F}_{2^m}$, onde p é um número primo e m é um número inteiro. Quanto às curvas elípticas, há tipos que permitem implementações eficientes de operações (como as de Koblitz) e tipos que precisam ser evitados, por resultarem em implementações sabidamente inseguras (como as singulares). Os efeitos sobre segurança e desempenho que tais parâmetros acarretam são discutidos em detalhes a seguir.

Um grande grau na liberdade de escolhas de tais parâmetros, desde que cercado com os devidos cuidados, certamente resulta em implementações mais seguras. Entretanto, pode haver problemas com interoperabilidade e desempenho entre diferentes implementações. Visando tratar tal problema, o consórcio SECG (através de suas especificações SEC 1 [8] e SEC 2 [9]) propôs algumas restrições nas escolhas de parâmetros, tentando garantir tanto um nível razoável de segurança, quanto desempenho e compatibilidade entre as implementações disponíveis no mercado.

3.1 Escolha do Corpo

Nesta seção serão mostrados corpos que permitem implementações eficientes de operações sobre curvas elípticas e corpos resistentes a ataques conhecidos. Os que são frágeis

do ponto de vista de segurança também são listados como escolhas a serem evitadas.

3.1.1 Corpos Primos ($\mathbb{F}_p$)

Opções de Bom Desempenho

- Números Primos de Mersenne Há algoritmos de teste de primalidade bastante otimizados para classes especiais de números, tais como os de Mersenne, que são números da forma $2^s - 1$, com $s \in \mathbb{Z}$. Tais números são muito úteis por também permitirem aritmética eficiente nos corpos neles baseados.

Como visto em [42], um dado número de Mersenne $2^s - 1$ é primo, se, e somente se:

(i) s é primo e

(ii) a série definida por $u_0 = 4$, $u_{k+1} = (u_k^2 - 2) \bmod n$ para $k \geq 0$ satisfizer a condição $u_{s-2} = 0$.

Os números de Mersenne permitem operação de redução de módulo de forma bastante eficiente, como no algoritmo seguinte, fazendo uso de deslocamentos, adições e multiplicações de precisão simples:

ALGORITMO: Redução de Módulo $m = b^t - c$

ENTRADA: base b, inteiro positivo x, e um módulo $m = b^t - c$, onde c tem l dígitos na base b, com $l < t$.
SAÍDA: $r = x \bmod m$.
1. $q_0 \leftarrow \lfloor x/b^t \rfloor$, $r_0 \leftarrow x - q_0 b^t$, $r \leftarrow r_0$, $i \leftarrow 0$
2. Enquanto $q_i > 0$ faça :
2.1 $q_{i+1} \leftarrow \lfloor q_i c/b^t \rfloor$, $r_{i+1} \leftarrow q_i c - q_{i+1} b^t$
2.2 $i \leftarrow i + 1$, $r \leftarrow r + r_i$
3. Enquanto $r \geq m$ faça $r \leftarrow r - m$
4. Retornar (r)

- Números Primos da Forma NIST A especificação FIPS 186-2 recomenda a implementação de curvas elípticas sobre corpos baseados nos seguintes números primos, que são uma generalização dos números de Mersenne:

$p_{192} = 2^{192} - 2^{64} - 1$
$p_{224} = 2^{224} - 2^{96} + 1$
$p_{256} = 2^{256} - 2^{224} + 2^{196} + 2^{96} - 1$
$p_{384} = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$
$p_{521} = 2^{521} - 1$

Os números primos NIST, como mostrado acima, são escritos como soma ou subtração de um pequeno número de potências de 2. Com exceção de p_{521}, as potências usadas nas expressões são todas múltiplas de 32. Isto resulta em operações de multiplicação com redução de módulo bastante rápidas em máquinas de 32 bits.

A especificação FIPS 186-2 provê regras para operações em cada um dos corpos gerados a partir dos números acima na forma de exemplos. A maneira usual de se multiplicarem dois inteiros (mod p) consiste em calcular o produto e reduzi-lo (mod p). Portanto, tem-se o seguinte problema: dado um número inteiro $A < p^2$, calcular $B \leftarrow A \bmod p$.

Em geral, deve-se obter B como o resto de uma divisão inteira. Se p for um número generalizado de Mersenne, entretanto, B pode ser expresso como uma soma ou diferença (mod p) de um pequeno número de termos. Para calcular esta expressão, deve-se obter a soma ou diferença inteira e reduzi-la modulo p. Esta última redução pode ser obtida através da adição ou subtração repetida de p.

A seguir tem-se o exemplo de como tal operação é feita para o corpo $\mathbb{F}_{192}$. Para os demais corpos a especificação FIPS 186-2[7] pode ser consultada para obtenção do procedimento.

O módulo em questão e dado por $p_{192} = 2^{192} - 2^{64} - 1$. Cada inteiro $A < p_{192}^2$ pode ser escrito como

$A = A_5 2^{320} + A_4 2^{256} + A_3 2^{192} + A_2 2^{128} + A_1 2^{64} + A_0$

onde cada A_i é um inteiro de 64 bits. A expressão para B é dada por:

$B \leftarrow T + S_1 + S_2 + S_3 \bmod p_{192}$

onde os termos de 192 bits são dados por

$T = A_2 2^{128} + A_1 2^{64} + A_0$

$S_1 = A_3 2^{64} + A_3$

$S_2 = A_4 2^{128} + A_4 2^{64}$

$S_3 = A_5 2^{128} + A_5 2^{64} + A_5$

Essas igualdades podem ser obtidas utilizando-se as seguintes congruências:

$2^{192} \equiv 2^{64} + 1 \pmod{p}$

$2^{256} \equiv 2^{128} + 1 \pmod{p}$

$2^{230} \equiv 2^{128} + 2^{64} + 1 \pmod{p}$

Em 2005, a Agência NSA dos Estados Unidos da América publicou uma série de recomendações na chamada "Suite B", abrangendo AES, hashing, assinaturas digitais e gerenciamento de chaves. Para proteção de documentos classificados como SECRET, a NSA recomenda o uso de corpos primos com módulo de 256 bits. Por outro lado, para proteção de documentos classificados como TOP SECRET, a recomendação da agência é que se utilizem corpos primos com módulo de 384 bits.

3.1.2 Corpos Binários ($\mathbb{F}_{2^m}$)

Opções de Bom Desempenho

- Bases Normais Na seção 2.2.2 foram mostradas bases normais, que permitem operações de multiplicação bastante eficientes (particularmente as gaussianas, que são ótimas). Além disso, o quadrado de um elemento pode ser obtido fazendo deslocamento circular (o que é relativamente barato para hardware). Os padrões NIST e SECG incluem as bases normais gaussianas entre suas recomendações.

- Bases polinomiais utilizando trinômios ou pentanômios Permitem redução de módulo de forma mais eficiente que outros polinômios com maiores número de termos dado o menor número de operações envolvidas no processo. A especificação ANSI X9.62[2] provê tabelas com tais polinômios para m variando até 1999.

Opções de Boa Segurança

- $\mathbb{F}_{2^m}$, onde m é um número primo Estes tipos de corpos binários são mandatórios para padrões como o ANSI X9.62 [2]. Eles inviabilizam o uso de ataques baseados no resultados de Weil para solução do problema ECDLP sobre curvas hiper-elípticas, como mostrado na seção 6.2.

Corpos Finitos Não-recomendados

- $\mathbb{F}_{2^m}$, onde m é um número composto Gaudry, Hess e Smart [28], baseados no trabalho de Frey [17], propuseram um solução do problema de logaritmo discreto sobre curva elíptica (ECDLP) pela aplicação da método de Weil mapeando ECDLP num problema sobre curva hiper-elíptica. Menezes e Qu [39] mostraram que este método não é viável para corpos $\mathbb{F}_{2^m}$ quando m é um número primo. Jacobson, Menezes e Stein [40], Maurer, Menezes and Teske [41] listam vários exemplos de parâmetros de domínio sobre $\mathbb{F}_{2^m}$ em que este tipo de ataque pode ser feito de forma viável. Baseando-se em tais resultados, padrões como X9.62, por exemplo, proíbem o uso de m composto visando evitar esta fragilidade. Este ataque é mostrado em maiores detalhes em 6.2.

De acordo com [44], há fortes evidências de que os corpos $\mathbb{F}_{2^m}$, com $m \in [185, 600]$ e divisível por 5 são fracos para uso em ECC. Ser fraco significa que para tais corpos o problema de ECDLP pode ser resolvido de forma bem mais eficiente que o método Pollard ρ.

Além destes, um outro forte candidato é $F_{2^{161}}$. Neste caso, para 2^{94} das 2^{162} classes de isomorfismo de curvas elípticas, a aplicação do método de redução GHS (Gaudry, Hess, Smart) resulta numa curva hiper-elíptica sobre $F_{2^{23}}$, onde o problema de HCDLP (logaritmo discreto sobre curva hiper-elíptica) é de fácil resolução. Desta forma, se um problema ECDLP arbitrariamente escolhido sobre $F_{2^{161}}$ puder ser eficientemente mapeado em uma curva elíptica que pertença a uma das 2^{94} curvas citadas acima, teríamos uma demonstração conclusiva da inadequação de $F_{2^{161}}$ para ECC.

3.2 Escolha da Curva

3.2.1 Opções de Bom Desempenho

- Curvas de Koblitz Este é um tipo especial de curva sobre F_2 conhecida como curva anômala binária, definida da seguinte forma $E(F_2) : y^2+xy = x^3+ax^2+1$, com $a \in \{0, 1\}$. O co-fator deste tipo de curva é 2, se $a = 1$ e 4, se $a = 0$. Multiplicação de pontos da curva elíptica por um escalar é bastante eficiente para este tipo de curva, como mostrado abaixo. Não é necessária duplicação de ponto como passo intermediário.

Dada uma curva de Koblitz E , define-se como mapa de Frobenius a transformação $\tau : E(\mathbb{F}_{2^m}) \rightarrow E(\mathbb{F}_{2^m})$ definida por $\begin{cases} \tau(x, y) = (x^2, y^2) \\ \tau(\infty) = \infty \end{cases}$

Para representação na base normal, a operação acima corresponde a deslocamento circular para a direita na representação de x e y, que tem baixo custo computacional.

Dados m e a, sejam as seguintes definições:

· Seja $c > 5$ um número inteiro.

· $\mu = (-1)^{1-a}$

· para $i = 0$ e $i = 1$ seja a seqüência $s_i(m)$ definida por

$$s_i(0) = 0, s_i(1) = 1 - i$$

$$s_i(m) = \mu s_i(m - 1) - 2s_i(m - 2) + (-1)^i$$

· Defina a seqüência $V(m)$ como

$$V(0) = 2, V(1) = \mu$$

$$V(m) = \mu V(m - 1) - 2V(m - 2)$$

A partir disto, pode-se calcular a multiplicação escalar nP na curva de Koblitz $E(\mathbb{F}_{2^m})$ como no algoritmo abaixo. Dado que é feito o uso de representação não-adjacente (TNAF), o número aproximado de adições e subtrações é, em média, de $m/3$, com probabilidade de pelo menos $1 - 2^{5-c}$.

ALGORITMO: Multiplicação de Ponto por Escalar

ENTRADA: escalar n , ponto P
SAÍDA: ponto Q resultado da multiplicação de P por n 1. Para $i = 0$ e $i = 1$ faça
1.1 $n\prime = \lfloor n/2^{a-C+(m-9)/2} \rfloor$
1.2 $g\prime \leftarrow s_i(m).n\prime$
1.3 $h\prime \leftarrow \lfloor g\prime/2^m \rfloor$
1.4 $j\prime \leftarrow V(m).h\prime$
1.5 $l\prime \leftarrow$ arredondamento$((g\prime + j\prime)/2^{(m+5)/2})$
1.6 $\lambda_i \leftarrow l\prime/2^C$
1.7 $f_i \leftarrow$ arredondamento(λ_i)
1.8 $\eta_i \leftarrow \lambda_i - f_i$
1.9 $h_i \leftarrow 0$
2. $\eta \leftarrow 2\eta_0 + \mu\eta_1$
3. Se $\eta \geq 1$
3.1 então
3.1.1 Se $\eta_0 - 3\mu\eta_1 < -1$
3.1.2 então $h_1 \leftarrow \mu$
3.1.3 senão $h_0 \leftarrow 1$
3.2 senão se $\eta_0 + 4\mu\eta_1 \geq 2$
3.2.1 então $h_1 \leftarrow \mu$
4. Se $\eta < -1$
4.1 então
4.1.2 Se $\eta_0 - 3\mu\eta_1 \geq 1$
4.1.2.1 então $h_1 \leftarrow -\mu$
4.1.2.2 senão $h_0 \leftarrow -1$
4.2 senão
4.2.1 se $\eta_0 + 4\mu\eta_1 < -2$
4.2.1.1 então $h_1 \leftarrow -\mu$
5. $q_0 \leftarrow f_0 + h_0$
6. $q_1 \leftarrow f_1 + h_1$
7. $r_0 \leftarrow n - (s_0 + \mu s_1)q_0 - 2s_1q_1$
8. $r_1 \leftarrow s_1q_0 - s_0q_1$
9. $Q \leftarrow 0$
10. $P_0 \leftarrow P$
11. Enquanto $r_0 \neq 0$ ou $r_1 \neq 0$
11.1 Se r_0 for ímpar então
11.1.1 $u \leftarrow 2 - (r_0 - 2r_1 \bmod 4)$

11.1.2 $r_0 \leftarrow r_0 - u$
11.1.3 Se $u = 1$ então $Q \leftarrow Q + P_0$
11.1.4 Se $u = -1$ então $Q \leftarrow Q - P_0$
11.2 $P_0 \leftarrow \tau P_0$
11.3 $(r_0, r_1) \leftarrow (r_1 + \mu r_0/2, -r_0/2)$
12. Sair Q

Em 1998, Gallant, Lambert e Vanstone [21], e Wiener e Zuccherato [64] mostraram que os melhores algoritmos conhecidos para solução do ECDLP podem ser acelerados por um fator de $\sqrt{2}$. Também mostraram que, uma dada curva elíptica $E(F_{2^s})$ com coeficientes $a, b \in F_{2^e}$, então o melhor algoritmo para solução de ECDLP em $E(F_{2^{st}})$ pode ser acelerado por um fator de $\sqrt{2t}$. A curva elíptica binária de Koblitz dada por $E : y^2 + xy = x^3 + x^2 + 1$, definida sobre F_2, tem o número de pontos dado por $\#E(F_{2^{163}}) = 2n$, onde n é um número primo de 162 bits. O problema de ECDLP$(F_{2^{163}})$ pode ser resolvido com cerca de 2^{77} operações de curvas elípticas. Isto representa 1/16 das 2^{81} operações que seriam necessárias para resolver o mesmo problema sobre uma curva aleatoriamente gerada com ordem similar. O ganho em desempenho, neste caso, tem como preço uma perda em segurança.

- Curvas CM O método de multiplicação complexa (CM) permite que se escolha a ordem do grupo formado por uma curva elíptica antes que a equação desta seja derivada explicitamente. Para curvas elípticas sobre $\mathbb{F}_q$, o método CM é também conhecido como método de Atkin-Morain. Para $\mathbb{F}_{2^m}$, ele é conhecido por método de Lay-Zimmer. A tabela 6.3 pode ser usada como auxílio na obtenção de curvas robustas a alguns ataques conhecidos.

As principais idéias em que se baseia este método foram derivadas do teste de primalidade proposto por Atkin [12]. O método faz uso de aritmética sobre corpos complexos quadráticos $K = \mathbb{Q}(\sqrt{-\mathbb{D}})$, onde o número $-D$ é um discriminante fundamental, servindo como entrada básica para a obtenção da curva. O número de classe de K deve ser baixo para que o método seja eficiente. Até o momento não foram encontradas fragilidades decorrentes desta escolha.

O número de pontos na curva elíptica é dado por $m = p + 1 - t$, onde t é o traço de Frobenius. Além disso, $t = \alpha + \overline{\alpha}$, onde α tem norma p em K.

Dado que a equação diofantina $4p = x^2 + Dy^2$ deve ser satisfeita para que a curva elíptica sobre $\mathbb{F}_p$ tenha multiplicação complexa da ordem do discriminante $-D$, temos $\alpha = \pm(x + \sqrt{-D}y)/2$, como visto em [56], capítulo VIII. Para solução da equação diofantina, que é equivalente a $p = u^2 + dv^2$, pode ser usado o algoritmo de Cornacchia, como mostrado a seguir:

ALGORITMO: Solução da Equação Diofantina

ENTRADA: número inteiro d, número primo p
SAÍDA: solução para $p = u^2 + dv^2$, caso exista uma.
1. Seja $p/2 < x_0 < p$ uma solução para $x^2 \equiv -d (\text{mod } p)$
2. $q_0 \leftarrow \lfloor p/x_0 \rfloor$
3. $x_1 = p \bmod q_0$
4. $k \leftarrow 0$
5. Até que $x_k^2 < p \leq x_{k-1}^2$ faça
5.1 $q_{k+1} \leftarrow \lfloor x_k/x_{k+1} \rfloor$
5.2 $x_{k+2} \leftarrow x_k \bmod x_{k+1}$
5.3 $k \leftarrow k + 1$
6. $u \leftarrow x_k$,$v \leftarrow \sqrt{(p - x_k^2)/d}$
7. Se $v \in \mathbb{Z}$ retornar (u, v) senão retornar 'não tem solução'

- Curvas com Homeomorfismos Convenientes Gallant, Lambert e Vanstone [22] mostraram que curvas elípticas com endomorfismos eficientemente calculados podem ser utilizadas para acelerar em até 50% o cálculo de multiplicação de pontos. As curvas de Koblitz descritas anteriormente são um exemplo disto. Outras incluídas nesta categoria:

$E_1 : \ y^2 = x^3 + ax$ definida sobre corpo primo $\mathbb{F}_p$, onde $p \equiv 1 \, (\text{mod} 4)$.

Seja $\alpha \in \mathbb{F}_p$ um elemento de ordem 4. Então, o mapeamento $\phi : E_1 \rightarrow E_1$ definido por $(x, y) \rightarrow (-x, \alpha y)$ e $\mathcal{O} \rightarrow \mathcal{O}$ é um homeomorfismo definido sobre $\mathbb{F}_p$. Se $P \in E_1(\mathbb{F}_p)$ for um ponto de ordem prima n , então ϕ funciona sobre $\langle P \rangle$ como se fosse um mapa de multiplicação $[\lambda]$, isto é, $\phi(Q) = \lambda Q$ para todo $Q \in \langle P \rangle$, onde λ é um inteiro satisfazendo $\lambda^2 \equiv -1 \, (\text{mod} n)$. Desta forma, $\phi(Q)$ é calculado utilizando-se uma única multiplicação sobre $\mathbb{F}_p$.

$E_2 : \ y^2 = x^3 + b$, definida sobre corpo primo $\mathbb{F}_p$, onde $p \equiv 1 \, (\text{mod} 3)$. Este tipo é utilizado pelo protocolo WTLS. Seja $\beta \in \mathbb{F}_p$ um elemento de ordem 3. Seja $\phi : E_2 \rightarrow E_2$ definido por $(x, y) \rightarrow (\beta x, y)$ e $\mathcal{O} \rightarrow \mathcal{O}$ um homeomorfismo definido sobre $\mathbb{F}_p$. Se $P \in E_2(\mathbb{F}_p)$ for um ponto de ordem prima n , então ϕ funciona sobre $\langle P \rangle$ como se fosse um mapa de multiplicação $[\lambda]$, isto é, $\phi(Q) = \lambda Q$ para todo $Q \in \langle P \rangle$, onde λ é um inteiro satisfazendo $\lambda^2 - \lambda \equiv -1 \, (\text{mod} n)$. Desta forma, $\phi(Q)$ é calculado utilizando-se uma única multiplicação sobre $\mathbb{F}_p$.

De forma geral, se o homeomorfismo ϕ puder agir sobre $\langle P \rangle$ como um mapa multiplicativo $[\lambda]$, para calcular kP basta encontrarmos k_1 e $k_2 \in [0, \sqrt{n}]$ tais que $k = k_1 + k_2\lambda \equiv (mod \, n)$.

Desta forma, $kP = (k_1 + k_2\lambda)P = k_1P + k_2\lambda P = k_1P + k_2\phi(P)$. Basta utilizar o método de múltiplas exponenciações simultâneas para obter kP. O algoritmo é mostrado na seqüência.

ALGORITMO: Multiplicações Simultâneas

ENTRADA: $w, q, t, k = (k_{t-1}, k_{t-2}, ..., k_0), l = (l_{t-1}, l_{t-2}, ..., l_0), P, Q$
SAÍDA: $kP + lQ$
1. Calcular $iP + jQ$ para todo $i, j \in [0, 2^w - 1]$.
2. Escrever $k = (k_{d-1}, k_{d-2}, ..., k_0)$ e $l = (l_{d-1}, l_{d-2}, ..., l_0)$ onde k_i e l_i são strings de comprimento w e $d = \lceil t/w \rceil$.
3. $R \leftarrow \mathcal{O}$
4. Para i variando de $d - 1$ até 0, faça
4.1 $R \leftarrow 2^w R$
4.2 $R \leftarrow R + (k_iP + l_iQ)$
5. Retornar R.

3.2.2 Opções de Boa Segurança

- Restrições para o número de pontos da Curva $\#E(F_q)$:

Os valores que $\#E(F_q)$ pode assumir devem obedecer algumas restrições para que a curva seja resistente aos ataques de Pohlig-Hellman e Pollard ρ , descritos em 6.2. $\#E(F_q)$ deve ser divisível por um número primo n bastante grande. Para aumentar a resistência contra o ataque de Pohlig-Hellman, $\#E(F_q)$ deve ser primo ou quase primo, ou seja, $\#E(F_q) = hn$ onde n é primo e $h \in \{1, 2, 3, 4\}$.

Para tornar a curva resistente a ataques de pequenos subgrupos, a condição é mais restritiva ainda: $\#E(F_q)$ deve ser primo.

3.2.3 Curvas a Serem Evitadas

- Curvas supersingulares Dado um corpo primo $\mathbb{F}_p$, define-se como supersingular uma curva elíptica $E(\mathbb{F}_p)$ tal que $\#E(\mathbb{F}_p) = p + 1$. Esse tipo de curva é explicitamente proibido em vários padrões por ser particularmente frágil a ataques MOV, como visto em 6.2. Há métodos eficientes de reduzir o problema ECDLP ao problema de logaritmo discreto sobre corpo finito para este tipo de curva.

- Curvas Anômalas sobre F_q Dado um corpo primo F_q, define-se como anômala uma curva elíptica $E(\mathbb{F}_p)$ tal que $\#E(\mathbb{F}_p) = p$. A exemplo das curvas supersingulares, este tipo também deve ser evitado. O ataque a curvas anômalas descrito em 6.2 permite resolver

o problema ECDLP através de algoritmos bastante eficientes, como os mostrados nos trabalhos de Semaev [55], Smart [59] e Satoh [11].

3.3 Recomendações do Consórcio SECG

3.3.1 SECG

A SECG (acrônimo de Standards for Efficient Cryptography Group) é um consórcio de indústrias fundado em 1998 com o objetivo de desenvolver padrões comerciais que facilitem a adoção eficiente de criptografia, garantindo interoperabilidade entre uma gama variada de plataformas de computação. Seus membros incluem companhias do setor de tecnologia e órgãos governamentais da área de segurança da informação.

3.3.2 Especificação SEC 2

O propósito desta especificação é listar exemplos de parâmetros de domínio para curvas elípticas nos níveis requeridos de segurança pela especificação SEC 1, bem como pelos padrões ANSI X9.62, ANSI X9.63 e IEEE P1363. A especificação recomenda fortemente que as implementações de ECC selecionem parâmetros a partir do conjunto listado por ela visando uma maior interoperabilidade entre sistemas que adotem soluções de criptografia baseadas em curvas elípticas.

Implementações que afirmarem estar em conformidade com esta especificação devem fazer uso de algum subconjunto dos parâmetros por esta recomendados em seus esquemas de criptografia construídos sobre curvas elípticas. Supõe-se que, tipicamente, as implementações que estiverem em conformidade com a SEC 2 optarão o mesmo com relação à SEC 1. Futuramente, o consórcio colocará à disposição dos implementadores um sistema que possa verificar conformidade. Atualmente está disponível um servidor que pode ser utilizado na verificação de ECDH (Geração de chaves utilizando a versão Diffie-Hellman sobre curvas elípticas) e ECDSA (Assinatura digital utilizando curvas elípticas).

3.3.2.1 Parâmetros de Domínio para $\mathbb{F}_p$

A SEC 2 lista um conjunto de parâmetros a serem utilizados quando o corpo sobre o qual as curvas elípticas são definidas é $\mathbb{F}_p$.

De acordo com convenção da especificação SEC 1, os parâmetros de domínio para curvas elípticas sobre $\mathbb{F}_p$ consistem num sêxtupla dada por: $T = (p, a, b, G, n, h)$, onde:

- p é um número inteiro primo que especifica o corpo finito $\mathbb{F}_p$;
- $a, b \in \mathbb{F}_p$, que definem uma curva elíptica $E(\mathbb{F}_p) : y^2 = x^3 + ax + b \pmod{p}$;

- $G = (x_g, y_g)$ é o ponto-base sobre $E(\mathbb{F}_p)$,
- n é um número primo correspondente à ordem do ponto-base G;
- h é um número inteiro correspondente ao co-fator $h = \#E(\mathbb{F}_p)/n$.

Os parâmetros acima são representados como um conjunto de octetos, de acordo com a especificação SEC 1. Tal documento ainda restringe a escolha de p de tal forma $\log_2 p \in \{112, 128, 160, 224, 256, 384, 521\}$. Isto tem por objetivo encorajar a interoperabilidade entre as implementações que seguem as especificações do consórcio SECG. Além disso, tais valores são um indicativo do grau de segurança associado aos parâmetros de domínio, dado que: se $\log_2 p = 2t$, então têm-se t bits de segurança. Isto significa que para se resolver o problema do logaritmo discreto seriam necessárias aproximadamente 2^t operações.

Os números primos p sobre os quais o corpo $\mathbb{F}_p$ é definido são escolhidos de forma a facilitar implementações eficientes como as descritas em [7]. Caso se mostre necessário deixar a escolha de p ser feita de forma aleatória devido a demanda comercial, a especificação SEC 2 deverá ser alterada para cobrir tal requisito. Na versão mais recente, os parâmetros de domínio associados a cada valor de p consistem de dois conjuntos distintos: um para as curvas de Koblitz e outro para parâmetros que tenham sido seguramente escolhidos aleatoriamente. Para segurança tipo exportação e tipo extremamente alto, no entanto, não há opção de curvas de Koblitz.

Parâmetros associados às curvas de Koblitz permitem implementações muito eficientes. Tais curvas são mais conhecidas como do tipo anômalo sobre $\mathbb{F}_{2^m}$, com $a, b \in \{0, 1\}$ [32]. Aqui, há uma generalização para $\mathbb{F}_p$ para curvas que possuem endomorfismo calculado de forma eficiente [20]. Os parâmetros associados às curvas de Koblitz foram escolhidos de tal forma que a curva resultante tivesse uma ordem correspondente a um número primo.

O segundo conjunto de parâmetros, por sua vez, oferece algumas características bastante conservadoras. Eles são escolhidos aleatoriamente através de uma semente usando SHA-1, de acordo com a especificação ANSI X9.62 [2] . Isto assegura que tais parâmetros não podem ter sido predeterminados; portanto, são pouco vulneráveis a ataques do tipo propósito especial. Além disso, é um forte indicativo de que armadilhas não foram colocadas durante sua geração. A semente utilizada na geração aleatória pode opcionalmente ser fornecida, de forma que o usuário possa fazer a verificação de que a seleção foi aleatória.

Nesta especificação, os parâmetros aleatórios são tais que a curva elíptica resultante tenha ordem prima ou que a multiplicação escalar de pontos possa ser acelerada pelo método de Montgomery [46]. Isto é conseguido quando repetidamente os parâmetros são aleatoriamente gerados e, a seguir, feita a verificação de contagem dos pontos até que os parâmetros adequados sejam encontrados. Tipicamente, os parâmetros são escolhidos de

tal forma que $a = p - 3$, dado que isso resulta em implementação eficiente. Para um dado p, aproximadamente metade das classes de isomorfismo de curvas elípticas sobre $\mathbb{F}_p$ contêm uma curva com $a = p - 3$. A especificação SEC 1 fornece mais instruções quanto à seleção de parâmetros de curva sobre $\mathbb{F}_p$.

Os parâmetros recomendados sobre $\mathbb{F}_p$ têm denominações mnemônicas que ajudam a identificá-los facilmente. Cada nome inicia-se com *sec* (para denotar Standards for Efficient Cryptography), seguido de p (para indicar que o corpo é $\mathbb{F}_p$), acompanhado por um número que indica o tamanho de p em bits. Na seqüência, tem-se k (para indicar que a curva elíptica é uma curva de Koblitz) ou r (para indicar que os parâmetros foram aleatoriamente escolhidos). Por último, tem-se um número de seqüência.

A tabela 3.1 resume as propriedades dos parâmetros recomendados sobre $\mathbb{F}_p$. Os seguintes campos estão presentes:

- parâmetros: indica o nome pelo qual o conjunto de parâmetros é referenciado na especificação;
- seção: identifica em que parte da especificação SEC 1 localiza-se a descrição do conjunto;
- segurança: indica o número de bits de segurança que o conjunto em questão oferece;
- tamanho: corresponde ao tamanho em bits da ordem do corpo $\mathbb{F}_p$;
- RSA/DSA: dá o tamanho aproximado em bits de uma implementação RSA/DSA que ofereça um grau de segurança equivalente ao conjunto de parâmetros ECC. (Maiores detalhes sobre segurança em ECC podem ser obtidos na especificação SEC 1 [8]);
- Koblitz / Aleatória: indica se curva elíptica é uma curva de Koblitz, ou se seus parâmetros foram escolhidos de forma seguramente aleatória.

A tabela 3.2 ilustra como as recomendações da SEC 2 [9] encontram-se com respeito aos padrões: {ANSI X9.62, ANSI x9.63, FSML, IEEP1363, IPsec, NIST, WAP}.

Legenda:

-: não conformidade

c: conformidade

r: parâmetros recomendados

Exemplo: O conjunto de parâmetros secp192k1 é mostrado abaixo, correspondendo a

$T = (p, a, b, G, n, h)$.
$\mathbb{F}_p$ é definido por:
$p = 2^{192} - 2^{32} - 2^{12} - 2^8 - 2^7 - 2^6 - 2^3 - 1$
Curva $E : y^2 = x^3 + ax + b$ definida por: $a = 0$, $b = 3$
Ponto-base G comprimido:
G = 03DB4FF10EC057E9AE26B07D0280B7F4341DA5D1B1EAE06C7D
e não-comprimido:
G = 04DB4FF10EC057E9AE26B07D0280B7F4341DA5D1B1EAE06C7D
9B2F2F6D9C5628A7844163D015BE86344082AA88D95E2F9D
Ordem de G:
n = FFFFFFFFFFFFFFFFFFFFFFFE26F2FC170F69466A74DEFD8D
Co-fator
h = 01

3.3.2.2 Parâmetros de Domínio para $\mathbb{F}_{2^m}$

A SEC 2 lista um conjunto de parâmetros a serem utilizados quando o corpo sobre o qual as curvas elípticas são definidas é $\mathbb{F}_{2^m}$.

De acordo com convenção sugerida pela especificação SEC 1, os parâmetros de domínio para curvas elípticas sobre este tipo de corpo são dados pelo sétuplo $T = (m, f(x), a, b, G, n, h)$, onde:

- m é um número inteiro que especifica o corpo finito $\mathbb{F}_{2^m}$;
- $f(x)$ é um polinômio irredutível de grau m;
- $a, b \in \mathbb{F}_{2^m}$ especificam a curva elíptica $E(\mathbb{F}_{2^m}) : y^2 + xy = x^3 + ax^2 + b$;
- $G = (x_g, y_g)$ é o ponto-base sobre a curva $E(\mathbb{F}_{2^m})$;
- n é o número primo que indica o ordem de G
- h é o co-fator $\#E(\mathbb{F}_{2^m})/n$

Seguindo recomendação da especificação SEC 1, $m \in \{113, 131, 163, 193, 233, 239, 409, 571\}$. Além disso, os polinômios utilizados para redução devem ser os listados na tabela 3.3. Estas restrições têm por objetivo encorajar interoperabilidade entre as diferentes implementações para os níveis de segurança mais comuns. A exemplo do que já havia sido feito para o corpo $\mathbb{F}_p$, para $\mathbb{F}_{2^m}$ também são definidos dois conjuntos de parâmetros para cada

nível de segurança: um associado às curvas de Koblitz e outro a curvas cujos parâmetros tenham sido escolhidos de forma aleatória. Para níveis de segurança do tipo exportação ou com extra-segurança, apenas o segundo tipo de conjunto está disponível.

Parâmetros associados às curvas de Koblitz resultam em implementações bastante eficientes. Tais curvas são do tipo binário anômalo, com $a, b \in \{0, 1\}$.

Quanto ao tipo de curva escolhida de forma aleatória, é utilizada uma semente obtida de SHA-1, de acordo com o especificado em ANSI X9.62. Os parâmetros são escolhidos através do processo repetido de escolha aleatória de uma semente e contagem de pontos na curva correspondente pelo algoritmo de Schoof até que um conjunto adequado seja encontrado.

Portanto, ou temos o parâmetro a escolhido aleatoriamente ou $a = 1$. É importante lembrar que aproximadamente metade das classes de isomorfismos de curvas elípticas sobre $\mathbb{F}_{2^m}$ contém curvas com $a = 1$. SEC 1 dá maiores detalhes sobre seleção de parâmetros de curvas elípticas sobre $\mathbb{F}_{2^m}$.

A nomenclatura adotada para os conjuntos de parâmetros é tal que:

- o nome inicia-se com *sec* (denotando Standards for Efficient Cryptography);
- seguido por t, para indicar $\mathbb{F}_{2^m}$;
- na seqüência, tem-se um número correspondente a m, que define o corpo sobre o qual a curva é definida;
- a seguir tem-se ou k para indicar que os parâmetros estão associados a uma curva de Koblitz ou r para indicar que a escolha foi feita de forma aleatória.

A tabela 3.4 resume as propriedades dos parâmetros recomendados sobre $\mathbb{F}_{2^m}$. Os seguintes campos estão presentes:

- parâmetros: indica o nome pelo qual o conjunto de parâmetros é referenciado na especificação;
- seção: identifica em que parte da especificação SEC 1 localiza-se a descrição do conjunto;
- segurança: indica o número de bits de segurança que o conjunto em questão oferece;
- tamanho: corresponde ao tamanho em bits da ordem do corpo $\mathbb{F}_{2^m}$;
- RSA/DSA: dá o tamanho aproximado em bits de uma implementação RSA/DSA que ofereça um grau de segurança equivalente ao conjunto de parâmetros ECC. (Maiores detalhes sobre segurança em ECC podem ser obtidos na especificação SEC 1 [8]);

- Koblitz / Aleatória: indica se curva elíptica é uma curva de Koblitz, ou se seus parâmetros foram escolhidos de forma seguramente aleatória.

A tabela 3.5 ilustra como as recomendações da SEC 2 encontram-se com respeito aos padrões: {ANSI X9.62, ANSI x9.63, FSML, IEEP1363, IPsec, NIST, WAP}.

Legenda:

-: não conformidade

c: conformidade

r: parâmetros recomendados

Exemplo:

sect571r1 correspondente aos parâmetros são dados por $T = (m, f(x), a, b, G, n, h)$

O corpo $F_{2^{571}}$ tem representação associada ao polinômio $f(x) = x^{571} + x^{10} + x^5 + x^2 + 1$

A curva $E : y^2 + xy = x^3 + ax + b$ sobre $\mathbb{F}_{2^m}$é definida por a, b dados por:

$a = 1$

b =02F40E7E2221F295DE297117B7F3D62F5C6A97FFCB8CEFF1CD6BA8CE
4A9A18AD84FFABBD8EFA59332BE7AD6756A66E294AFD185A78FF12AA
520E4DE739BACA0C7FFEFF7F2955727A

aleatoriamente escolhidos, com semente
S = 2AA058F73A0E33AB486B0F610410C53A7F132310, como especificado em ANSI X9.62 [2] na representação em base normal, e convertido para representação em base polinomial.

Ponto-base G comprimido:

G = 030303001D34B856296C16C0D40D3CD7750A93D1D2955FA80A
A5F40FC8DB7B2ABDBDE53950F4C0D293CDD711A35B67FB1499AE6003
8614F1394ABFA3B4C850D927E1E7769C8EEC2D19

e não-comprimido:

G = 040303001D34B856296C16C0D40D3CD7750A93D1D2955FA80A
A5F40FC8DB7B2ABDBDE53950F4C0D293CDD711A35B67FB1499AE6003
8614F1394ABFA3B4C850D927E1E7769C8EEC2D19037BF27342DA639B
6DCCFFFE B73D69D78C6C27A6009CBBCA1980F8533921E8A684423E43
BAB08A576291AF8F461BB2A8B3531D2F0485C19B16E2F1516E23DD3C
1A4827AF1B8AC15B

ordem de G:

n = 03FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFE661CE18FF55987308059B186823851EC7DD9CA1
161DE93D5174D66E8382E9BB2FE84E47

co-fator:
h = 02

Neste capítulo foram mostradas opções de corpos finitos e curvas que resultam em implementações eficientes e seguras. Foram mostradas também escolhas a ser evitadas, pois sabidamente resultam em implementações frágeis ou de fraco desempenho. Vários destes parâmetros estão relacionados à ordem da curva elíptica. No próximo capítulo serão mostrados diferentes métodos através dos quais tal grandeza pode ser calculada.

Tabela 3.1: Parâmetros para Corpos Primos

Parâmetros	Seção	Segurança	Tamanho	RSA/DSA	Koblitz/Aleatória
secp112r1	2.2.1	56	112	512	a
secp112r2	2.2.2	56	112	512	a
secp128r1	2.3.1	64	128	704	a
secp128r2	2.3.2	64	128	704	a
secp160k1	2.4.1	80	160	1024	k
secp160r1	2.4.2	80	160	1024	a
secp160r2	2.4.3	80	160	1024	a
secp192k1	2.5.1	96	192	1536	k
secp192r1	2.5.2	96	192	1536	a
secp224k1	2.6.1	112	224	2048	k
secp224r1	2.6.2	112	224	2048	a
secp256k1	2.7.1	128	256	3072	k
secp256r1	2.7.2	128	256	3072	a
secp384r1	2.8.1	192	384	7680	a
secp521r1	2.9.1	256	521	15360	a

Tabela 3.2: Compatibilidade entre Padrões

Parâmetros	X9.62	X9.63	FSML	P1363	IPSec	NIST	WTLS
secp112r1	-	-	-	c	c	-	r
secp112r2	-	-	-	c	c	-	c
secp128r1	-	-	-	c	c	-	c
secp128r2	-	-	-	c	c	-	c
secp160k1	c	r	c	c	c	-	c
secp160r1	c	c	c	c	c	-	r
secp160r2	c	r	c	c	c	-	c
secp192k1	c	r	c	c	c	-	c
secp192r1	r	r	c	c	c	r	c
secp224k1	c	r	c	c	c	-	c
secp224r1	c	r	c	c	c	r	c
secp256k1	c	r	c	c	c	-	c
secp256r1	r	r	c	c	c	r	c
secp384r1	c	r	c	c	c	r	c
secp521r1	c	r	c	c	c	r	c

Tabela 3.3: Polinômios Redutores (SEC 1)

Corpo	Polinômios de Redução
$F_{2^{113}}$	$f(x) = x^{113} + x^9 + 1$
$F_{2^{131}}$	$f(x) = x^{131} + x^8 + x^3 + x^2 + 1$
$F_{2^{163}}$	$f(x) = x^{163} + x^7 + x^6 + x^3 + 1$
$F_{2^{193}}$	$f(x) = x^{193} + x^{15} + 1$
$F_{2^{233}}$	$f(x) = x^{233} + x^{74} + 1$
$F_{2^{239}}$	$f(x) = x^{239} + x^{36} + 1$ou $f(x) = x^{239} + x^{158} + 1$
$F_{2^{283}}$	$f(x) = x^{238} + x^{12} + x^7 + x^5 + 1$
$F_{2^{409}}$	$f(x) = x^{409} + x^{87} + 1$
$F_{2^{571}}$	$f(x) = x^{571} + x^{10} + x^5 + x^2 + 1$

Tabela 3.4: Parâmetros para Corpos Binários (SEC 1)

Parâmetros	Seção	Segurança	Tamanho	RSA/DSA	koblitz/aleatório
sect113r1	3.2.1	56	113	512	a
sect113r2	3.2.2	56	113	512	a
sect131r1	3.3.1	64	131	704	a
sect131r2	3.3.2	64	131	704	a
sect163k1	3.4.1	80	163	1024	k
sect163r1	3.4.2	80	163	1024	a
sect163r2	3.4.3	80	163	1024	a
sect193r1	3.5.1	96	193	1536	a
sect193r2	3.5.2	96	193	1536	a
sect233k1	3.6.1	112	233	2240	k
sect233r1	3.6.2	112	233	2240	a
sect239k1	3.7.1	115	239	2304	k
sect283k1	3.8.1	128	283	3456	k
sect283r1	3.8.2	128	283	3456	a
sect409k1	3.9.1	192	409	7680	k
sect409r1	3.9.2	192	409	7680	a
sect571k1	3.10.1	256	571	15360	k
sect571r1	3.10.2	256	571	15360	a

Tabela 3.5: Compatibilidade entre padrões

Parâmetros	X9.62	X9.63	FSML	P1363	IPSec	NIST	WTLS
sect113r1	-	-	-	c	c	-	r
sect113r2	-	-	-	c	c	-	c
sect131r1	-	-	-	c	c	-	c
sect131r2	-	-	-	c	c	-	c
sect163k1	c	r	r	c	r	r	r
sect163r1	c	c	r	c	r	-	c
sect163r2	c	r	r	c	c	r	c
sect193r1	c	r	c	c	c	-	c
sect193r2	c	r	c	c	c	-	c
sect233k1	c	r	c	c	c	r	c
sect233r1	c	r	c	c	c	r	c
sect239k1	c	c	c	c	c	-	c
sect283k1	c	r	r	c	r	r	c
sect283r1	c	r	r	c	r	r	c
sect409k1	c	r	c	c	c	r	c
sect409r1	c	r	c	c	c	r	c
sect571k1	c	r	c	c	c	r	c
sect571r1	c	r	c	c	c	r	c

Capítulo 4

Determinação da Ordem de Curvas Elípticas

Este capítulo exibe diferentes métodos para determinação da ordem de curvas elípticas, dado ser esta grandeza de grande importância na seleção de parâmetros de domínio seguros em sistemas criptográficos baseados nestas curvas.

O conjunto de pontos pontos sobre uma curva elíptica $E(F_q)$ definida sobre um corpo finito F_q , juntamente com a operação binária de adição de pontos, formam um grupo abeliano. A ordem n de tal grupo é dada pelo número de pontos da curva. Para que tal curva seja aceitável para efeitos de criptografia, sua ordem precisa satisfazer algumas restrições que dificultem a solução do problema de ECDLP, resultando em protocolos menos susceptíveis a ataques.

Algumas destas restrições são:

- $n = s.r$ onde r é um número primo, s é um número inteiro, e $s \ll r$. Isto significa que há um sub-grupo de ordem prima de valor r . Quanto mais alto for r , mais forte será a curva do ponto de vista de criptografia.

- $n = q$, primo, não é aceitável, dado que isto implicaria numa curva anômala, e, portanto, fraca para aplicações em criptografia (com exceção das curvas de Koblitz).

- n não pode ser um divisor de $q^l - 1$, $1 \leq l \leq M$ pois resultaria numa curva supersingular, e, portanto frágil a ataques MOV, como mostrado em 6.2. A grandeza M dá uma medida do grau de segurança em relação a este tipo de ataque, e é conhecido como limiar de MOV. Em geral, um valor de 20 é suficiente para assegurar um grau razoável de segurança.

Este capítulo lida com formas de se determinar a ordem do grupo formado por uma curva elíptica, que é um problema não-trivial, se levarmos em consideração que os números

envolvidos são grandes (no mínimo, com 160 bits de comprimento, para que se tenha um nível aceitável de segurança).

No capítulo anterior foram listados tipos de curvas com boas propriedades para uso em criptografia:

- Curvas CM: este tipo de curva é construído a partir de um valor dado com entrada para a ordem do grupo gerado pela curva elíptica. A ordem, portanto, já é conhecida.
- Curvas de Koblitz: para o corpo $\mathbb{F}_{2^m}$ vimos que há dois valores para o cofator: $\{2, 4\}$, o que simplifica bastante o cálculo da ordem da curva. Também será analisado o caso de característica diferente de 2.
- Curvas aleatórias: para este tipo de curva, é necessário cálculo da ordem e verificação da existência de sub-grupo cuja ordem seja dada por número primo, como mecionado acima.

4.1 Teorema de Hasse

Dado um corpo finito F_q e uma curva elíptica definida sobre ele $E(F_q)$, o número de pontos sobre tal curva é dado por

$$\#E(F_q) = q + 1 - t$$

onde t corresponde ao traço de Frobenius em q. O mapa de Frobenius de potência q, representado por $\phi : E \rightarrow E$, é definido como:

$$\begin{cases} \tau(x, y) = (x^q, y^q) \\ \tau(\infty) = \infty \end{cases}$$

t e ϕ relacionam-se da seguinte forma: $\phi^2 - [t]\phi + [q] = [0]$.

O teorema de Hasse afirma que t satisfaz a seguinte relação: $|t| \leq 2\sqrt{q}$. Ela estabelece um intervalo de comprimento $4\sqrt{q}$ em torno do valor $q+1$ para valores que t pode assumir. Uma demonstração deste teorema pode ser encontrado em [58]. Esta relação facilita a criação de algoritmos de cálculo do número de pontos de $E(F_q)$ ao restringir os valores que a ordem da curva pode assumir.

4.2 Cálculo da Ordem do Grupo

O teorema de Hasse e as restrições listadas no início deste capítulo podem ser utilizados na determinação do valor exato da ordem do grupo gerado por uma curva elíptica.

Suponhamos que um dado número m tenha sido dado como sendo $\#E(F_q)$, para uma curva elíptica definida sobre o corpo finito F_q. As seguintes verificações podem ser feitas para que seja assegurado que tal valor é de fato a ordem do grupo associado à curva. Qualquer uma delas que falhe é um indicativo de que o número não é a ordem do grupo, ou que a curva não é adequada para criptografia.

- verificar se $q + 1 - 2\sqrt{q} \leq m \leq q + 1 + 2\sqrt{q}$;
- selecionar aleatoriamente um ponto P na curva E e verificar se $[m]P = \infty$. Caso o algoritmo que forneceu o número para teste forneça mais de um valor para ser verificado, a condição acima deve ser avaliada em cada um deles. A utlilização de vários outros pontos além de P aumenta a probabilidade de que o valor que passar pelos testes seja de fato a ordem do grupo. É necessária alguma garantia de que entre os valores passados pelo algoritmo para verificação esteja aquele que de fato é a ordem do grupo.
- verificar que a fatoração $m = s.r$, é possível, com $s \ll r$. É necessário que r seja primo. Caso $[s]P = \infty$, deve-se escolher um novo ponto para execução do teste. A condição $r > 4\sqrt{q}$ garante que m é a ordem do grupo e que não há nenhum outro múltiplo de r no intervalo de Hasse.

4.3 Curvas de Koblitz

As curvas de Koblitz têm uma forma bastante simples para cálculo do número de pontos, como mostrado abaixo.

Sejam E uma curva elíptica definida sobre F_q , c_1 seu traço de Frobenius, $Z(E;T) = \exp(\sum \frac{N_n}{n} T^n)$, $n \geq 1$ (função zeta) e $N_n = \#E(F_{q^n})$. Koblitz [30] mostra que

$$Z(E;T) = \frac{P(T)}{(1-T)(1-qT)}$$

onde

$$P(T) = 1 - c_1 T + qT^2 = (1-\alpha)(1-\overline{\alpha})$$

com discriminante não-positivo, e α com magnitude $\sqrt{q}$.

A partir do resultado acima, $N_n = \#E(F_{q^n}) = q^n + 1 - \alpha^n - (\overline{\alpha})^n$

4.4 Método de Shanks e Mestre

Este método é descrito em [13] e baseia-se na técnica BSGS ("Baby Step / Giant Step"), a qual consiste no cálculo de duas séries de produto de escalar por pontos na curva elíptica

e na busca por termos coincidentes em tais séries. O método tem por objetivo determinar a ordem da curva elíptica, tendo complexidade da ordem de $O(q^{1/4+\epsilon})$. Para valores muito altos de q este algoritmo é inadequado. Nas próximas seções serão vistas opções de menor complexidade.

Do teorema de Hasse, temos que $\#E(F_q) = q + 1 - t$, onde $|t| \leq 2\sqrt{q}$. Seja $t' = t + \lfloor 2\sqrt{q} \rfloor \in [0, 4\sqrt{q}]$ a ser usado no algoritmo abaixo.

ALGORITMO: Cálculo do número de pontos de uma curva elíptica

ENTRADA: F_q, $E(F_q)$
SAÍDA: $\#E(F_q)$
1. $Q \leftarrow (q+1)P$ para ponto P sobre a curva aleatoriamente escolhido.
2. $Q_1 \leftarrow Q + [\lfloor 2\sqrt{q} \rfloor]P$
3. $m \leftarrow \lceil 2.\sqrt[4]{q} \rceil$
4. Para $j = 0, ..., m-1$ calcular e armazenar $[j]P$. Isto corresponde aos passos pequenos.
5. Para $i = 0, ..., m-1$ calcular $Q_1 - [i]([m]P)$ até que o resultado coincida com algum $[j]P$ calculado acima. Isto corresponde aos passos grandes. Quando a coincidência de valores é obtida, temos $t' = im + j$. Dado que $t' = t + \lfloor 2\sqrt{q} \rfloor$ também, a obtenção de t é direta.
6. Retornar $M = q + 1 - t$

4.5 Algoritmo de Schoof

Este algoritmo para obtenção de $\#E(F_q)$ representa uma melhora significativa em relação ao de Shanks-Mestre, que tinha grau de complexidade $O(q^{1/4+\epsilon})$. Seu grau de complexidade é $O(log^8 q)$. Ele baseia-se em:

- teorema de Hasse $\#E(F_q) = q + 1 - t$, $|t| \leq 2\sqrt{q}$
- na determinação de t mod l para l primo satisfazendo $\prod_{i=2,...,max} l_i > 4\sqrt{q}$, onde l_{max} é o menor valor para o qual a desigualdade é verdadeira.
- teorema chinês de restos, que permite obter t a partir de t mod l_i
- mapa de Frobenius ϕ (como mostrado em 4.1), que satisfaz

$$\phi^2(P) - [t]\phi(P) + [q]P = \infty$$

ALGORITMO: Cálculo do número de pontos de uma curva elíptica

ENTRADA: Corpo finito F_q e curva elíptica $E(F_q)$ definida sobre ele.
SAÍDA: número de pontos de $E(F_q)$
1 $M \leftarrow 2$, $l \leftarrow 3$, $S = \{(t \bmod 2, 2)\}$
2 Enquanto $M < 4\sqrt{q}$ faça:
2.1 Para $\tau = 0, ..., (l-1)/2$ faça
2.1.1 Procurar $P \in E[l]$ que satisfaça $\phi^2(P) + [q]P = \pm[\tau]\phi(P)$ e sair deste laço quando encontrá-lo
2.2 $S \leftarrow S \cup \{(\tau, l)\}$ ou $S \leftarrow \cup\{(-\tau, l)\}$, dependendo do sinal para o qual P foi obtido acima.
2.3 $M \leftarrow M \times l$
2.4 $l \leftarrow$primo que sucede l
3 Obter t usando o conjunto S e o algoritmo chinês de restos (ver [56], página 13)
3 Retornar $q + 1 - t$

Embora este algoritmo apresente um ganho de desempenho em relação a Shanks-Mestre, sua complexidade ainda é alta para aplicá-lo a valores elevados de q adequados para uso em criptografia. O algoritmo abaixo apresenta menor grau de complexidade.

4.6 Algoritmo de Schoof-Elkies-Atkin

Este método é uma extensão daquele visto em 4.5 e depende de termos as raízes da equação característica do mapa de Frobenius $\mathcal{F}_l(u) = u^2 - t_l u + q_l = 0$ contidas em F_l (equivalentente a $\Delta_t = t^2 - 4q$ ter ou não raiz quadrada em F_l). Em caso positivo, l corresponde a um número primo de Elkies; caso contrário, correponde a um número primo de Atkin.

Como não dispomos de t para fazer a verificação acima, é necessário que utilizemos o polinômio modular (ver definição em 2) para determinar qual o tipo de número primo temos em mãos.

Caso tenhamos um número primo de Elkies, podemos construir uma curva isógena de grau l com polinômio característico que possa ser usado para identificarmos λ tal que $(x^q, y^q) = [\lambda](x, y)$, onde (x, y) corresponde a um ponto sobre a curva elíptica $E(F_q)$.

Caso tenhamos um número primo de Atkin, podemos obter a informação referente ao subconjunto de valores possíveis para $t \bmod l$. O tamanho de tal subconjunto é determinado pela função de Euler $\phi_{Eul}(r)$, com $r \leq l + 1$.

O algoritmo resultante está mostrado abaixo. Maiores detalhes referentes à implementação podem ser vistos em [47].

ALGORITMO: Cálculo do número de pontos de uma curva elíptica

ENTRADA: Curva elíptica $E(F_q)$
SAÍDA: $\#E(F_q)$
1. $M \leftarrow 1$, $l \leftarrow 2$, $A \leftarrow \{\}$, $E \leftarrow \{\}$
2. Enquanto $M < 4\sqrt{q}$ faça:
2.1. Através do polinômio modular, definir se l é número primo de Atkin ou Elkies
2.2. Se l for número primo de Elkies, faça:
2.2.1. Determinar polinômio característico $f_l(x)$ para isogenia grau l
2.2.2. Obter auto-valor λ mod l para $f_l(x)$
2.2.3. $t \leftarrow \lambda + q/\lambda \pmod{l}$
2.3.4. $E \leftarrow E \cup \{(t, l)\}$
2.3. Senão
2.3.1. Determinar conjunto T tal que $t \pmod{l} \in T$
2.3.2. $A \leftarrow A \cup \{(T, l)\}$
2.4. $M \leftarrow M \times l$
2.5. $l \leftarrow$ número primo que sucede l
3. Recuperar t a partir dos conjuntos A e E
4. Retornar $q + 1 - t$

A parte do algoritmo relativa ao tratamento de números primos de Atkin tem complexidade exponencial. Quanto menor for a quantidade de tais números tratados pelo algoritmo, tanto mais eficiente ele será. Pode ser escolhido apenas um subconjunto destes números (suficiente para que t possa ser recuperado) de forma a não penalizar o algoritmo como um todo.

A parte relativa ao tratamento dos números primos de Elkies tem complexidade $O(\log q)^6$. O algoritmo deve garantir que a maioria dos números primos a serem tratados seja deste tipo para ter um bom desempenho. Filtrar por completo todos os números de Atkin, entretanto, não é uma opção seguida na prática, por apresentar algumas desvantagens.

Este capítulo mostrou algumas formas para determinação da ordem de curvas elípticas. Tal grandeza tem bastante relevância na escolha de parâmetros de domínio em ECC, de forma a ter-se um sistema seguro. Alguns dos ataques mais eficazes contra sistemas ECC são baseados no cálculo do logaritmo discreto sobre grupos formados a partir dos pontos de curvas hiper-elípticas. O capítulo seguinte discorre sobre tais curvas.

Capítulo 5

Curvas Hiper-elípticas

Desde a proposição de criptografia utilizando chaves públicas por Diffie-Hellman, uma variedade de grupos tem sido usada como suporte, tais como corpos finitos e curvas elípticas. Há padrões definidos sobre eles, bem como uma vasta gama de aplicações comerciais. Recentemente, curvas hiper-elípticias de genus baixo têm despertado interesse para utilização em criptografia. O trabalho pioneiro nesta área foi desenvolvido por Koblitz [31], que propõe o uso do Jacobiano de uma curva hiper-elíptica imaginária na troca de chaves.

Um esquema semelhante foi posteriormente proposto por Scheidler, mas sobre curvas reais, utilizando o anel de funções regulares da curva e sua infra-estrutura semelhante a grupo como suporte. Uma modificação deste esquema proposta em [54] não apresentou ganho significativo em desempenho. Jacobson [61] propôs uma abordagem baseada em divisores aritméticos para curva hiper-elíptica real definida sobre um corpo finito. O uso de tais divisores resultou em protocolo aproximadamente 15% mais rápido que o convencional (que usa Jacobiano de curvas híper-elípticas imaginárias).

Trabalhos sobre desempenho e segurança de criptografia sobre curvas hiper-elípticas são abundantes atualmente, objetivando diminuir a diferença ainda existente em relação às curvas elípticas. As curvas hiper-elípticas são bem mais abundantes que as elípticas, e acredita-se que o problema de logaritmo discreto correspondente possa resultar em implementações em criptografia mais seguras que as de curvas elípticas para o mesmo tamanho de chave. Sua adoção por padrões (como HESSL para IEEE em [34], por exemplo) ainda não foi consolidada, no entanto.

As seções seguintes apresentam uma descrição conceitual de curvas hiper-elípticas e dos mecanismos correntemente utililizados para uso em criptografia. As demonstrações das relações listadas abaixo podem ser vistas em [30].

5.1 Definições e Propriedades

Curvas hiper-elípticas são uma classe especial de curvas algébricas e constituem-se numa generalização de curvas elípticas. Há curvas hiper-elípticas para cada valor de genus maior ou igual a 1. Quando o genus é 1, temos em mãos uma curva elíptica.

Seja K um corpo e $\overline{K}$ seu fecho algébrico, define-se como curva hiper-elíptica de genus g sobre K ($g \geq 1$) a equação da forma

$$C : v^2 + h(u)v = f(u) \,, \text{em } K[u, v]$$

onde $h(u) \in K[u]$ é um polinômio de grau no máximo igual a g , $f(u) \in K[u]$ é um polinômio mônico de grau $2g + 1$, e não há pontos de singularidade, ou seja, não há solução $(u, v) \in \overline{K} \times \overline{K}$ que satisfaça a equação $v^2 + h(u)v = f(u)$ e anule as derivadas parciais $2v + h(u) = 0$ e $h'(u)v - f'(u) = 0$.

Dada uma extensão L do corpo K , os pontos racionais de L sobre C , denominados $C(L)$, são o conjunto de pontos $P = (x, y) \in L \times L$ que satisfazem a equação da curva hiper-elíptica juntamente com o ponto no infinito (∞).

Se $P(x, y)$ é um ponto sobre a curva hiper-elíptica, define-se como seu oposto o ponto $\tilde{P} = (x, -y - h(x))$, que também está sobre a curva. Se um ponto e seu oposto coincidem (como, por exemplo, ∞), eles são ditos especiais; caso contrário, são ditos comuns.

As seções seguintes mostram uma série de resultados que visam chegar à definição do jacobiano da curva hiper-elíptica. Ele forma o grupo sobre o qual a maioria de implementações de criptografia sobre este tipo de curva se baseia.

5.2 Funções Polinomiais e Racionais

Define-se como anel de coordenadas de C sobre K , o anel quociente $K[C] = K[u, v]/(v^2 + h(u)v - f(u))$, onde $(v^2 + h(u)v - f(u))$ representa o ideal em $K[u, v]$ gerado pelo polinômio irredutível sobre $\bar{K}$ dado por $v^2 + h(u)v - f(u)$.

De forma semelhante, $\bar{K}[C] = \bar{K}[u, v]/(v^2 + h(u)v - f(u))$ é o anel de coordenadas de C sobre $\bar{K}$. Um elemento de $\bar{K}[C]$ é chamada função polinomial em C.

Para uma dada função polinomial $G(u, v) = a(u) - b(u)v$ em $\bar{K}[C]$, define-se como seu conjugado a função polinomial $\bar{G}(u, v) = a(u) + b(u)(h(u) + v)$.

A norma de G é dada por $N(G) = G\bar{G}$, que tem as seguintes propriedades:

- $N(G)$ é um polinômio em $\bar{K}[u]$
- $N(\bar{G}) = N(G)$
- $N(GH) = N(G)N(H)$, onde H é um polinômio em $\overline{K}[u]$

O corpo de funções $K(C)$ de C sobre K é o corpo de frações $K[C]$. De forma semelhante, o corpo de funções $\bar{K}(C)$ de C sobre K é o corpo de frações de $\bar{K}[C]$. Os elementos de $\bar{K}(C)$ correspondem a funções racionais em C.

Dada a função polinomial $G(u,v) = a(u) - b(u)v$, define-se como grau de G o valor dado por $\text{grau}(G) = \max[2\text{grau}_u(a),\ 2g+1+2\text{grau}_u(b)]$. As seguintes propriedades se aplicam à definição de grau:

- $\text{grau}(G)$=$\text{grau}_u(N(G))$
- $\text{grau}(GH)$ =$\text{grau}(G)$+$\text{grau}(H)$
- $\text{grau}(G)$=$\text{grau}(\bar{G})$

5.3 Polos e Zeros

Seja $R \in \bar{K}(C)^*$ e P um ponto sobre a curva hiper-elíptica C. Se tivermos $R(P) = 0$, então R tem um zero em P. Caso R não seja definido em P , então P é um polo de R , e $R(P) = \infty$.

Sejam $G \in \bar{K}[C]^*$ e $P \in C$. Se $G(P) = 0$, então $\bar{G}(\tilde{P}) = 0$.

Dado $P(x,u)$ sobre a curva hiper-elíptica C , se $G = a(u) - b(u)v \in \overline{K}[C]^*$ tiver um zero em P e x não for raiz para ambos $a(u)$ e $b(u)$, então $\overline{G}(P) = 0 \Longleftrightarrow P$ for um ponto especial.

Se $P(x,y)$ for um ponto especial sobre a curva hiper-elíptica C , $(u-x)$ pode ser escrito da forma $(v-y)^2.S(u,v)$, onde $S(u,v) \in \overline{K}[C]$ não tem nem zero nem polo em P.

Dado $P \in C$, existe uma função $U \in \overline{K}(C)$, com $U(P) = 0$ tal que, para toda função polinomial $G \in \bar{K}[C]^*$ existem um inteiro d e uma função $S \in \overline{K}(C)$ tais que $S(P) \neq 0, \infty$ e $G = U^d S$. Além disso, o número d não depende da escolha de U , que é denominada parâmetro de uniformização para P.

Nas condições listadas acima, a ordem de G em P é definida como $\text{ordem}_p(G) = d$ e implica nos seguintes resultados:

- Dados $G_1, G_2 \in \bar{K}[C]^*$ e $P \in C$, com $\text{ordem}_P(G_1) = r_1$, $\text{ordem}_P(G_2) = r_2$, temos que $\text{ordem}_P(G_1G_2) = \text{ordem}_P(G_1) + \text{ordem}_P(G_2)$
- Supondo que $G_1 \neq -G_2$, se $r_1 \neq r_2$, então $\text{ordem}_P(G_1+G_2) = \min(r_1, r_2)$. Se $r_1 = r_2$, então $\text{ordem}_P(G_1+G_2) \geq \min(r_1, r_2)$.
- $\text{ordem}_P(G) = \text{ordem}_{\bar{P}}(\bar{G})$

Dado $G \in \bar{K}[C]^*$, G tem um número finito de zeros e polos, e

$$\sum_{P \in C} \text{ordem}(P) = 0$$

.

Dados $R = G/H \in \bar{K}(C)^*$ e $P \in C$, $\text{ordem}_P(R) = \text{ordem}_P(G) - \text{ordem}_P(H)$.

5.4 Divisores

Um divisor é definido como uma soma formal de pontos em C

$$\sum_{P \in C} m_P P \, , m_P \in \mathbb{Z}$$

onde há apenas um número finito de m_P são diferentes de zero. Tem-se:

$$\text{Grau}(D) = \sum_{P \in C} m_P$$

e $\text{ordem}_P(D) = m_P$.

O conjunto D de todos os divisores forma um grupo aditivo sob a seguinte regra de adição:

$$\sum_{P \in C} m_P P + \sum_{P \in C} n_P P = \sum_{P \in C} (m_p + n_P) P$$

Dados dois divisores $D_1 = \sum_{P \in C} m_P P$ e $D_2 = \sum_{P \in C} n_P P$, o máximo divisor comum dos dois tem grau zero e é dado por:

$$\text{mdc}(D_1,D_2) = \sum_{P \in C} \min(m_P, n_P) P - (\sum_{P \in C} \min(m_p, n_p)) \infty$$

Dado $R \in \overline{K}(C)^*$, seu divisor é definido como:

$$\text{div}(R) = \sum_{P \in C} (\text{ordem}_P(R)) P$$

Se $R = G/H$, $\text{div}(R) = \text{div}(G) - \text{div}(H)$ e tem grau zero.

O divisor principal é definido da seguinte forma: um divisor D pertencente ao grupo de divisores de grau zero $\mathbf{D^0}$ é chamado divisor principal se $D = \text{div}(R)$ para alguma função racional $R \in \overline{K}(C)^*$. O conjunto de todos os divisores principais é um subgrupo, denotado por $\mathbf{P}$, dos divisores de grau zero. O grupo quociente $\mathbf{J}=\mathbf{D^0}/\mathbf{P}$ é chamado Jacobiano da curva C.

Seja $D = \sum_{P \in C} m_P P$ um divisor. Define-se como suporte do divisor D o conjunto $\text{sup}(D) = \{P \in C \mid m_p \neq 0\}$.

Divisor semi-reduzido é um divisor da forma $D = \sum m_i P_i - (\sum m_i)\infty$, onde cada $m_i \geq 0$ e os P_i são pontos tais que se $P_i \in \sup(D)$ então $\tilde{P}_i \notin \sup(D)$, exceto para $P_i = \tilde{P}_i$ (que implica em $m_i = 1$). Se $\sum m_i \leq g$, onde g é o genus da curva C, então D é um divisor reduzido.

5.5 Sistemas de Criptografia Baseados em Curvas Hiper-Elípticas

De forma semelhante à adaptação que foi feita para curvas elípticas para o cálculo de logaritmo discreto (ECDLP), também para curvas hiper-elípticas tem-se a versão (HCDLP, Logartimo discreto sobre curva hiper-elíptica) que consiste em: dada uma curva hiper-elíptica C definida sobre um corpo finito K , e dados dois divisores reduzidos $D_1, D_2 \in \mathbf{J}(K)$, determinar o inteiro positivo l tal que $D_2 = lD_1$, desde que tal número exista.

Para propósito de criptografia, o corpo e a curva utilizados para implementação devem ser tais que não seja conhecido nenhum algoritimo sub-exponencial para resolução do problema HCDLP. Devem ser evitados, por exemplo, os parâmetros citados em [50], [24], [49] e [14]. Além disso, para que a implementação seja prática e eficiente, as operações de grupo devem ser relativamente fáceis de aplicar. O jacobiano de uma curva hiper-elíptica definida sobre um corpo finito é uma das opções para tal grupo.

O corpo finito K a ser escolhido deve ser tal que operações aritméticas possam ser eficientemente implementadas sobre ele. A curva C a ser escolhida deve ser tal que a ordem do jacobiano $\mathbf{J}(K)$ de C seja divisível por um número primo grande r . Se $K = F_q$, r não pode dividir $q^m - 1$ para pequenos valores de m , visando evitar o ataque de Frey [19].

Este capítulo apresentou o conceito de curvas hiper-elípticas, mostrando algumas reduções do problema de logaritmo discreto que podem comprometer a segurança de sistemas ECC, caso os parâmetros não sejam criteriosamente escolhidos. No próximo capítulo serão discutidos alguns ataques conhecidos contra ECC e as medidas que podem ser adotadas para tornar os sistemas ECC menos vulneráveis a eles.

Capítulo 6

Compromisso entre Segurança e Desempenho

Este capítulo mostra diferentes ataques contra sistemas ECC e a forma como a escolha de parâmetros de domínio pode tornar os sistemas mais robustos a estes.

A segurança dos sistemas de criptografia baseados em curvas elípticas reside no fato de que os melhores algoritmos para resolução do problema de logaritmo discreto (ECDLP) têm complexidade exponencial. Koblitz [38] afirma, por exemplo, em relação à versão de Diffie-Hellman para curvas elípticas (ECDHP), que se ECDLP não puder ser resolvido em tempo sub-exponencial, ECDHP também não poderá.

Para alguns casos especiais, como os mostrados nos diferentes ataques das seções seguintes, é possível reduzir o problema do ECDLP a um outro mais simples, como o de DLP (Problema de Logaritmo Discreto) sobre um corpo finito, que possa ser resolvido em tempo sub-exponencial. Entretanto, uma escolha cuidadosa de parâmetros de domínio pode evitar tais fragilidades, mantendo a complexidade de solução do ECDLP.

Por outro lado, como visto na seção 3, uma seleção criteriosa de corpo finito e curva elítpica podem resultar implementações bastante eficientes. É necessário, no entanto, observar alguns cuidados para que a segurança do sistema de criptografia resultante também seja mantida. Neste capítulo serão discutidos os efeitos que a escolha de parâmetros acarreta quanto a estes dois aspectos.

6.1 Efeito da Escolha de Parâmetros

Nesta seção serão discutidos alguns fatores que devem ser levados em consideração na escolha dos parâmetros, de forma a resultar em sistemas de diferentes graus de segurança e desempenho, conforme os requisitos do sistema criptográfico a ser implementado. Devido à utilização comercial de várias implementações de padrões e a necessidade de se manter

compatibilidade entre elas, é necessário observar as limitações (listadas no capítulo 7) que acabam sendo impostas por tais padrões às escolhas de parâmetros.

Tomando como exemplo ataques que visem a quebra de chaves, os melhores parâmetros são aqueles que deixem como método mais eficiente de ataque a busca exaustiva. Além disso, tais parâmetros devem produzir um número muito grande de chaves entre as quais o adversário deverá fazer a escolha. No que concerne a criptografia sobre curvas elípticas, os parâmetros a serem escolhidos envolvem seleção de corpo finito e curva elíptica. O corpo influencia nas execução de operações aritméticas e no tamanhos das chaves. Por sua vez, o grupo formado pela curva elíptica limita através de seu tamanho o número possível de chaves, e determina o algoritmo de cifragem através do mapeamento determinado pela operação de adição de pontos.

6.1.1 Desempenho

- Corpos binários $\mathbb{F}_{2^m}$ apresentam arimética mais eficiente que corpos primos $\mathbb{F}_p$, principalmente quando a implementação de operações de protocolos é feita em hadware específico (como FPGA descritas em [43], por exemplo). Isto se deve ao fato de que operações em nível de bit associadas ao corpo $\mathbb{F}_{2^m}$ não necessitarem de "vai um", como é o caso de $\mathbb{F}_p$.

- Dentre as diferentes representações possibilitadas para corpos binários, a de base normal resulta em implementações mais simples e eficientes em hardware, como descrito em [52]. A operação de elevação ao quadrado, que é utilizada como auxiliar na multiplicação de pontos, consiste em fazer um simples deslocamento circular de bits.

- Para implementações em software utilizando corpos binários, a representação polinomial resulta em operações de multiplicação de melhor desempenho que as obtidas com base normal, conforme visto em [48].

- Quando se fizer uso de corpos primos, a adoção dos números NIST (listados na especificação FIPS 186-2 [7]) resulta em operações de redução de módulos mais eficientes que as obtidas com números primos arbitrários devido ao fato de que o número de divisões, que são operações custosas, é minimizado.

- As curvas elípticas de Koblitz sobre corpos binários, quando comparadas às curvas aleatoriamente geradas, permitem implementações bastante eficientes de multiplicação de pontos na curva por um escalar, dado que evitam o cálculo de duplicação de pontos. Resultados obtidos sobre corpos binários ($\mathbb{F}_{2^m}$, $m \in \{163, 233, 283\}$)

por Hankerson, Hernandez e Menezes em [37] mostram, em média, que o tempo necessário para efetuar a multiplicação utilizando curvas elípticas de Koblitz corresponde a cerca de 56% daquele gasto com curva aleatória. Tal número não deveria ser tomado como uma regra geral, no entanto. Ele corresponde à implementação feita pelos autores para um conjunto bastante reduzido de parâmetros, mas está em acordo com o que a teoria prevê.

- Curvas com homeomorfismos convenientes, como as mostradas em 3.2.1, e que se assemelhem a mapas de multiplicação, podem acelerar o cálculo de multiplicação de pontos. Tal artificio é usado no padrão WTLS, visto em [10] .

- Quando ao invés de utilizar curvas reconhecidamente com bom desempenho (como as de Koblitz), o implementador optar pela geração de curvas, o método CM representa uma boa escolha, pois resulta numa curva cujo número de pontos ja é conhecido. Desta forma, evita-se calcular a ordem da curva, que é uma operação geralmente custosa, com visto na seção 4.

- Em geral, curvas aleatoriamente geradas por outros métodos que o CM implicam no custo adicional de se determinar a ordem da curva. Quando a ordem obtida é tal que indique que a curva é fraca do ponto de vista de segurança, é necessário gerar iterativamente novas curvas até encontrar uma adequada.

- A utilização de um conjunto de domínio de parâmetros tais como os mostrados na especificação SEC 2 [9] pode resultar numa redução significativa de esforço computacional. Além de poder optar por conjuntos que possuam curva de Koblitz tanto sobre corpos binários quanto sobre corpos primos, o usuário tem a garantia de que tais parâmetros (inclusive a custosa ordem da curva) tenham sido testados para assegurar de um nível adequado de segurança. Este padrão ainda lista curvas aleatoriamente geradas, provendo o usuário com a semente SHA utilizada na geração, caso este opte por fazer verificação.

6.1.2 Segurança

- Os corpos primos $\mathbb{F}_p$ resultam em parâmetros de domínio que fornecem aproximadamente t bits de segurança, onde $2t = \lceil \log_2 p \rceil$. Embora a escolha de p possa ser arbitrária, aconselha-se que fique restrita a um conjunto pequeno (segundo SEC 2 [9], por exemplo, p tal que $\lceil \log_2 p \rceil \in \{160, 192, 224, 256, 384, 521\}$) para garantia de interoperabilidade com outros padrões e implementações existentes. O mínimo valor admissível para p sob o aspecto de segurança corresponde a 160 bits para padrões como X9.62 [2].

- Os corpos binários $\mathbb{F}_{2^m}$ também devem satisfazer o critério de segurança mínimo de 160 bits listado acima. A exemplo do que foi dito para corpos primos, aconselha-se que m seja restrito para aumentar a interoperabilidade das diferentes implementações: SEC 2 recomenda que $m \in \{163, 193, 233, 239, 283, 409, 571\}$. Valores altos para m resultam em implemetações mais seguras, posto que a solução do problema de ECDLP é exponencial sobre o número de pontos do grupo em que está definido. Porém, tais valores resultam em implementações mais lentas, dado que as operações subjacentes aos protocolos de criptografia serão feitas sobre números de maior comprimento em bits.

- Os corpos binários compostos ($\mathbb{F}_{2^m}$, com m não-primo) são susceptíveis a ataques do tipo Descida de Weil, e devem ser evitados.

- As curvas de Koblitz em $\mathbb{F}_{2^m}$, que resultam em operações bastante eficientes de multiplicação de pontos, têm um aspecto desfavorável de segurança quando comparadas às curvas aleatoriamente geradas. Para as primeiras, o método de Pollard paralelizado pode fazer uso de classes de equivalência sob o mapa de Frobenius em conjunto com mapa de negação, como mostrado no capítulo 4 de [43], para obter uma aceleração de $\sqrt{2m}$. Além disso, alguns autores acreditam que, pelo fato destas curvas terem um forma especial, podem mais facilmente ser alvos de ataques específicos, quando comparadas às curvas aleatórias, a exemplo do que ocorre com curvas anômalas ou supersingulares.. Tais tipos de ataque contra as curvas de Koblitz, que encontrem solução do problema ECDLP em tempo sub-exponencial, não são conhecidos ainda. Isto deve ser levado em consideração quando da execução de análise segurança x desempenho para seleção de parâmetros que satisfaçam os requisistos do sistema.

- O ponto-base gerador do grupo formado por pontos da curva elíptica a serem utilizados nos protocolos de criptografia precisa satisfazer as restrições listadas na tabela do item 6.3, visando fornecer robustez aos ataques mais conhecidos.

- As curvas a serem utilizadas precisam satisfazer algumas condições quanto a seu números pontos de forma a serem adequadas a aplicações em criptografia. Para resistir ao ataque MOV, elas devem ser tais que sua ordem não seja divisor de traço de Frobenius; para resistir ao ataque anômalo, a ordem não pode coincidir com a característica do corpo $\mathbb{F}_p$. As curvas listadas em especificações como a SEC 2 já satisfazem tais condições.

6.1.3 Critérios para a escolha dos parâmetros

Como visto nos itens 6.1.1 e 6.1.2, a obtenção de um sistema com o grau mais elevado de segurança disponível normalmente exclui a obtenção do grau mais elevado de desempenho (e vice-versa), quando consideramos o aspecto da escolha de parâmetros. Os requitos de segurança e desempenho do sistema a ser construído é que irão ditar a escolha a ser feita.

Por exemplo, para se ter o máximo grau de desempenho num sistema em que os protocolos de criptografia são implementados em hardware, o ideal seria escolhermos o corpo binário $\mathbb{F}_{2^m}$, com menor m aceitável, e utilizar curvas de Koblitz. Além disso, o uso de parâmetros pré-determinados como os da especificação SEC 2 economiza processamento razoável na validação dos parâmetros (que certamente envolverá cálculo da ordem da curva), assegurando, por tabela, interoperabilidade com outros sistemas que estejam em conformidade com tal padrão.

No outro extremo, em que segurança é o fator primordial, poderíamos utilizar um corpo com o maior número aceitável de elementos, associando a isto a escolha aleatória de curvas. Isto implicaria certamente em operações criptográficas mais lentas, devido ao tamanho em bits do dados manipulados, bem como o acréscimo de esforço computacional decorrente da validação de parâmetros associados à curva. Agregado a isto, tem-se a possibilidade de baixa interoperabilidade com outros sistemas.

Portanto, faz-se necessária uma avaliação criteriosa das reais necessidades do sistema criptográfico sendo construído, para que este não seja penalizado sob os aspectos de segurança e desempenho. Tais necessidades irão fornecer guias para a escolha adequada dos parâmetros de domínio do sistema.

6.2 Ataques

Os ataques conhecidos contra curvas elípticas podem ser classificados de acordo com sua abrangência em gerais e específicos. Os gerais são tais que seu tempo de execução depende primariamente do tamanho dos parâmetros da curva elíptica. Os específicos dependem de alguns fatores particulares da curva elíptica (como número de pontos, por exemplo) para que tenham bom desempenho. A tabela 6.1 lista os ataques a serem vistos nesta seção e suas respectivas classificações.

6.2.1 Pohlig-Hellman

Seja uma curva elíptica E definida sobre um corpo finito F_q e $P \in E(F_q)$ um ponto de ordem n. Dado $Q \in \langle P \rangle$, o problema de logaritmo discreto consistem em encontrar $l \in [0, n-1]$ tal que $Q = lP$.

Seja a decomposição de n em fatores primos dada por

$$n = \prod_{i=1,...,r} p_i^{e_i}$$

O ataque de Pohlig e Hellman reduz o problema de obter l ao problema de obter logaritmos discretos nos subgrupos de $\langle P \rangle$ de ordem prima p_i. Desta forma, a dificuldade de se resolver ECDLP sobre $\langle P \rangle$ equivale a resolução de ECDLP sobre os subgrupos. Para maximizar a resistência do sistema a este tipo de ataque, P deve ser escolhido de tal forma que sua ordem n seja divisível por um número primo grande.

A estratégia aplicada neste ataque consistem em obter $l_i = l \bmod p_i^{e_i}$, $1 \leq i \leq r$ e resolver as congruências $l \equiv l_i \pmod{p_i^{e_i}}$ para $l \in [0, n-1]$. O teorema chinês de restos garante que a solução é única.

O valor de l_i é resultado do cálculo de e_i logaritmos discretos no subgrupo de ordem p_i de $\langle P \rangle$ como mostrado abaixo:

$$l_i = z_0 + z_1 p_i + z_2 p_i^2 + ... + z_{e_i-1} p_i^{e_i-1},\ z_i \in [0, p_i - 1]$$

Cálculo de z_0: $Q_0 = \frac{n}{p_i} Q = l(\frac{n}{p_i} P) = lP_0 = z_0 P_0$. Daí, resolve-se ECDLP em $\langle P_0 \rangle$ para obter z_0.

Cálculo de z_1: $Q_1 = \frac{n}{p_i^2}(Q - z_0 P) = \frac{n}{p_i^2}(l - z_0)P = (l - z_0)(\frac{n}{p_i^2} P) = (z_0 + z_1 P - z_0)(\frac{n}{p_i^2} P) = z_1(\frac{n}{p_i} P) = z_1 P_0$. Daí, resolvendo-se ECDLP para $Q_1 = z_1 P_0$ em $\langle P_0 \rangle$ obtém-se z_1

De forma semelhante, z_t pode ser obtido a partir de ECDLP de Q_t sobre P_0

$$Q_t = \frac{n}{p_i^{t+1}}(Q - z_0 P - z_1 p_i P - z_2 p_i^2 P - ... - z_{t-1} p_i^{t-1} P)$$

6.2.2 Pollard

Pollard [53] dá uma série de métodos para resolução do problema de logaritmo discreto numa gama variada de grupos. Dois destes métodos (ρ e λ) são bastante citados na literatura de criptografia. Apenas o primeiro deles será descrito nesta seção. Tal algoritmo de propósito geral está entre os melhores na solução de ECDLP, e seu grau de complexidade é utilizado como referência para avaliar quão eficientes são os outros métodos.

Seja uma curva elíptica E definida sobre um corpo finito F_q e $P \in E(F_q)$ um ponto de ordem n. Dado $Q \in \langle P \rangle$, o problema de logaritmo discreto consiste em encontrar $l \in [0, n-1]$ tal que $Q = lP$. O método ρ consiste em percorrer o grupo $\langle P \rangle$ sobre o qual o ECDLP é definido à busca de um ciclo (daí a analogia à letra grega que lhe dá o nome), tentando otimizar o uso de espaço em memória. Desta forma, encontram-se pares distintos de números inteiros (c', d') e (c'', d'') tais que:

$$c'P + d'Q = c''P + d''Q$$

Daí,

$$(c' - c'')P = (d'' - d')Q = (d'' - d')lP$$

Resultando

$$l = (c' - c'')(d'' - d')^{-1} \bmod n$$

Uma forma nada otimizada de se efetuar a busca de ciclo consiste em selecionar aleatoriamente os pares de números $(c, d) \in [0, n-1]$ e constuir uma tabela contendo $(c, d, cP + dQ)$ até que $cP + dQ$ seja repetido na tabela, ou seja, que uma colisão ocorra. O número esperado de iterações, bem como de entradas na tabela, até que a colisão ocorra é, em média, $\sqrt{\pi n/2}$.

O método adotado pelo algoritmo Pollard ρ faz uma otimização de espaço, mantendo, no entanto, o número esperado de iterações. Para tanto, é definida a função de iteração $f : \langle P \rangle \rightarrow \langle P \rangle$ tal que, para $X \in \langle P \rangle$ e $c, d \in [0, n-1]$, com $X = cP + dQ$, seja fácil calcular $\overline{X} = f(X)$ e $(\bar{c}, \bar{d})$ com $\overline{X} = \overline{c}P + \overline{d}Q$.

Seja, por exemplo, uma partição aleatória$\{S_1, S_2, ..., S_L\}$ de $\langle P \rangle$, tal que os elementos tenham aproximadamente o mesmo tamanho. Definindo a função de partição como $H(X) = j$, se $X \in S_j$ e considerando $a_j, b_j \in [0, n-1]$, $1 \leq j \leq L$, pode-se definir f da seguinte forma:

$$f(x) = X + a_j P + b_j Q \text{ , onde } j = H(x)$$

Para um dado $X = cP + dQ$, tem-se $f(X) = (c + a_j \bmod n)P + (d + b_j \bmod n)Q$. Qualquer $X_0 \in \langle P \rangle$ resulta numa seqüência $X_i = f(X_{i-1})$, com $i \geq 1$ e $X_i \in P$. Cedo ou tarde, irá haver colisão, dado que $\langle P \rangle$ é finito, e a seqüência entrará em ciclo. Desta forma, há um índice t para o qual $X_t = X_{t+s}$ para algum $s \geq 1$ e $X_i = X_{i-s}$ sempre que $i \geq t + s$. Para encontrar colisão, pode-se utilizar o algoritmo de Floyd [16], que consiste em calcular os pares (X_i, X_{2i}) até que $X_i = X_{2i}$. O número esperado de iterações até ocorrência de uma colisão é de $k = 1.03\sqrt{n}$, com $t \leq k \leq t + s$.

Oorschot [63] propôs um esquema de paralelização do método de Pollard ρ de forma a obter um ganho de desempenho proporcional ao número de máquinas envolvidas no processamento. Em seu trabalho, foi analisado $F_{2^{155}}$. A máquina teórica que sugeriu para resolver o problema de ECDLP levaria cerca de 36 dias para chegar a uma solução. Aumentando-se o valor do menor dos fatores primos pelo qual a ordem da curva é divisível, pode-se melhorar a segurança do sistema a este tipo de abordagem.

6.2.3 MOV (Menezes, Okamoto, Vanstone)

Seja uma curva elíptica E definida sobre um corpo finito F_q e $P \in E(F_q)$ um ponto de ordem n , primo. Seja G um grupo de ordem n. Dado que n é primo, $\langle P \rangle$ e G são cíclicos e isomórficos. Se conseguirmos determinar um isomorfismo $\psi : \langle P \rangle \rightarrow G$, então as instâncias de ECDLP em $\langle P \rangle$ podem ser reduzidas a instâncias de DLP em G . Isto signfica que, dados $P, Q \in \langle P \rangle$, com $Q = lP$, temos $l = log_{\psi(P)}\psi(Q)$. O método proposto a seguir foi descrito por Menezes, Okamoto, Vanstone em [36], e reduz o problema de ECDLP sobre um subgrupo de $E(F_q)$ no problema de DLP sobre a extensão F_{q^k} de F_q. Ele faz uso de emparelhamentos de Weil. O emparelhamento de Tate também podem ser utilizado, sem necessidade da restrição de que $q-1$ seja divisível por n exigida no emparelhamento de Weil.

Seja a curva $E(F_q)$, onde $q = p^m$. Seus pontos e a operação de adição formam um grupo abeliano, com ∞ servindo como identidade. Pelo teorema de Hasse, o grupo tem ordem $\#E(F_q) = q + 1 - t$, com $|t| \leq 2\sqrt{q}$. Além disso, $E(F_q)$ é isomórfica a $\mathbb{Z}_{n1} \oplus \mathbb{Z}_{n2}$, onde $n2 \mid n1$ e $n2 \mid q-1$. E também é curva elíptica sobre qualquer extensão de F_{q^k}. Neste caso, o teorema de Weil permite calcular $\#E(F_{q^k})$ a partir de $\#E(F_q)$. Como $t = q + 1 - \#E(F_q)$, $\#E(F_{q^k}) = q^k + 1 - \alpha^k - \beta^k$, onde α, β são números complexos tais que $1 - tT + qT^2 = (1 - \alpha T)(1 - \beta T)$. Se $\mathrm{mdc}(n, q) = 1$, (torção-n) $E[n] \subset E(F_q)$, se, e somente se:

- $n^2 \mid \#E(F_q)$
- $n \mid q - 1$
- $\phi \in \mathbb{Z}$ ou $\nu(t^2 - 4q/n^2) \subset \mathrm{End}_{F_q}(E)$

Seja n um número inteiro primo relativo a p, e $e_n : E[n] \times E[n] \rightarrow F_q$ um emparelhamento de Weil. Como mostrado no apêndice de [36], Miller desenvolveu um algoritmo de complexidade polinomial para cálculo deste tipo de emparelhamento. Com $E(F_q)$ como definida acima, $P_1, P_2 \in E(\mathbb{F}_p)$ estão na mesma co-partição de $\langle P \rangle$, P de ordem n_1 , se ,e somente se, $e_{n1}(P, P_1) = e_{n_1}(P, P_2)$.

A redução do problema é feita como mostrado a seguir. Seja $E(F_q)$ isomórfica a $\mathbb{Z}_{n1} \oplus \mathbb{Z}_{n2}$, com $n_1 \mid n_2$ e $\mathrm{mdc}(\#E(F_q), q) = 1$. Disso resulta que $E[n_1]$ é isomórfica a $\mathbb{Z}_{n1} \oplus \mathbb{Z}_{n1}$. Dado P de ordem n , com n sendo divisor de n_1 e $R \in E(F_q)$, o ECDLP consite em encontrar $l \in [0, n-1]$ tal que $R = lP$. Temos que $R \in \langle P \rangle \iff nR = \infty$ e $e_n(P, R) = 1$. O algoritmo seguinte mostra como obter informações sobre l resolvendo DLP sobre $\mathbb{F}_p$ quando a ordem de P é máxima.

ALGORITMO: Cálculo do resto da divisão inteira do logaritmo discreto

---

ENTRADA: $P, R \in E(F_q)$, com ordem de P máxima n_1 e $R = lP$
SAÍDA: $l' \equiv l \pmod{n'}$, onde n' é um divisor de n_2
1. Selecionar aleatoriamente $T \in E(F_q)$
2. $\alpha \leftarrow e_{n_1}(P, T)$, $\beta \leftarrow e_{n1}(R, T)$
3. Sair com $l' \leftarrow \log_\alpha \beta$ em F_q

O algoritmo seguinte não requer que a ordem de P seja a máxima e retorna l:

ALGORITMO: Cálculo do logatimo discreto

ENTRADA: $P \in E(F_q) \text{com ordem de } P = n.\ R \in \langle P \rangle$
SAÍDA: l inteiro tal que $R = lP$
1. Determinar maior inteiro k tal que $E[n] \subseteq E(F_{q^k})$
2. Encontrar $Q \in E[n]$ tal que $\alpha = e_n(P, Q)$ e tenha ordem n
3. $\beta \leftarrow e_n(R, Q)$
4. Sair com $l \leftarrow \log_\alpha \beta$ em F_{q^k}

Seja $E(F_q)$ uma curva supersingular de ordem $q + t - 1$ sobre F_q , onde $q = p^m$. As seguintes classes de curva são possíveis para E:

- I - $t = 0$ e $E(F_q)$ isomórfica a $\mathbb{Z}_{q+1}$
- II - $t = 0$ e$E(F_q)$ isomórfica a $\mathbb{Z}_{(q+1)/2} \oplus \mathbb{Z}_2$ (e $q \equiv 3 \pmod 4$)
- III - $t^2 = q$ e m par
- IV - $t^2 = 2q$, $p = 2$ e m ímpar
- V - $t^2 = 3q$ e $p = 3$ e m ímpar
- VI - $t^2 = 4q$ e m par

A tabela 6.2 é utilizada pelo terceiro algoritmo de cálculo do ECDLP para as curvas supersingulares dos tipos mostrados acima:

Algoritmo para cálculo de logaritmo a partir da classe supersingular:

ALGORITMO: Cálculo de logaritmo discreto

ENTRADA: $P \in E(F_q)$, P de ordem n , $R \in \langle P \rangle$, E supersingular
SAÍDA: inteiro l tal que $R = lP$
1. Obter k, c a partir da tabela acima.

2. Escolher Q' aleatoriamente de E . $Q \leftarrow (cn_1/n)Q'$
3. $\alpha \leftarrow e_n(P,Q)$ e $\beta \leftarrow e_n(R,Q)$
4. $l \leftarrow \log_\alpha \beta$ em F_{q^k}
5. Se $lP = R$ então retornar l , senão voltar para passo 2.

As três versões de algoritmos mostradas têm complexidade sub-exponencial. Os parâmetros da curva devem ser diferentes daqueles listados nesta seção para que o sistema seja robusto ao ataque MOV.

6.2.4 Anômalo

Seja o corpo primo $\mathbb{F}_p$ e $E(\mathbb{F}_p)$ uma curva elíptica definida sobre ele. Tal curva é classificada como anômala, se $\#E(\mathbb{F}_p) = p$. O grupo definido pelos pontos da curva e a operação de adição é cíclico, dado que a ordem da curva é prima. Consequentemente, tal grupo é isomórfico ao grupo aditivo de inteiros módulo p , denotado por $\mathbb{F}_p^+$.

O problema do logaritmo discreto sobre $\mathbb{F}_p^+$ consiste em encontrar um número inteiro $l \in [0, p-1]$ tal que $la \equiv b \,(\mathrm{mod}\, p)$, uma vez que tenham sido dados o número primo p , e $a, b \in \mathbb{F}_p$, com $a \neq 0$. Desta forma, pode-se eficientemente calcular $l = ba^{-1} \,\mathrm{mod}\, p$.

Uma vez que se consiga determinar adequadamente um isomorfismo $\psi : E(\mathbb{F}_p) \rightarrow \mathbb{F}_p^+$, como mostrado em [11] e [59] se $E(\mathbb{F}_p)$ for anômala, consegue-se eficientemente reduzir o problema de ECDLP sobre $E(\mathbb{F}_p)$ ao de DLP sobre $\mathbb{F}_p$, que pode ser resolvido em tempo sub-exponencial. Este tipo de curva, portanto, mostra-se inadequado ao uso em protocolos de criptografia.

6.2.5 Logaritmos Múltiplos

Este tipo de ataque consiste em resolver simultaneamente vários problemas de ECDLP sobre o mesmo conjunto de parâmetros de domínio. Um cenário em que isso pode ser visualizado é o de uma rede em que vários usuários compartilhem os mesmos parâmetros de domínio, embora cada um tenha sua própria chave pública.

Pode-se resolver tais problemas iterativamente através de Pollard ρ paralelizado, por exemplo, como mostrado por Kuhn [33]. Quando o primeiro problema é resolvido, pode-se utilizar o resultado para resolver o segundo, e assim sucessivamente. Menezes [35] mostra que, se para resolver o primeiro problema gastou-se um tempo t, os subseqüentes serão resolvidos em um intervalo de tempo decrescente como listado abaixo:

- segundo: $(\sqrt{2} - 1)t$

- terceiro: $(\sqrt{3}-\sqrt{2})t$
- ... i-ésimo: $(\sqrt{i}-\sqrt{i-1})t$

Desta forma, selecionando-se os parâmetros de forma a fazer com que a primeira instância seja difícil de ser resolvida, pode-se tratar adequadamente este tipo de ataque.

6.2.6 Descida de Weil

Seja $E(\mathbb{F}_{2^m})$ uma curva elíptica não-supersingular definida sobre o corpo binário $\mathbb{F}_{2^m}$, com $m = nh$ composto, n primo e h pequeno. Frey [18] propôs o uso da descida de Weil com o intuito de reduzir o problema do logaritmo discreto ECDLP em $\mathbb{F}_{2^m}$ ao problema de DLP sobre o jacobiano de uma curva de genus maior que 1, definida sobre um sub-corpo F_{2^l} , onde l é um divisor de m.

O método consiste em quatro passos fundamentais:

- construir uma restrição de Weil de escalares de E , que corresponda a uma variedade abeliana de dimensão m/l sobre F_{2^l}.
- encontrar uma curva $C(F_{2^l})$ sobre a restrição definida no passo anterior.
- mapear o problema de ECDLP em $E(\mathbb{F}_{2^m})$ no de DLP no jacobiano de C.
- resolver o DLP no jacobiano de C , empregando, por exemplo, o algoritmo de Hafner-McCurley [25] para cálculo de índices.

Uma mudança neste método foi feita por Gaudry, Hess e Smart [28] (ataque GHS) para obtenção de uma curva hiper-elíptica C correspondente à intersecção de $n-1$ hiper-planos sobre a restrição de Weil. Neste artigo, eles ainda dão uma sugestão de um algoritmo bastante eficiente para a redução do ECDLP sobre E no HCDLP sobre $\mathbf{J}(C)$. Para que o cálculo seja eficiente, o genus da curva hiper-elíptica não deve ser maior que 8. Métodos para resolução de HCDLP, tais como o cálculo Enge-Gaudry [23] têm complexidade sub-exponencial.

O ataque GHS falha para as curvas elípticas sobre $\mathbb{F}_{2^m}$, para todos os m primos no intervalo $[160, 600]$, que corresponde ao conjunto de escolhas permitidas pelos padrões listados na seção 7. Para tais valores de m, as curvas hiper-elípticas geradas ou têm genus muito baixo (e, portanto, só fornecem informações triviais sobre o ECDLP) ou têm genus muito alto (e tornam, conseqüentemente, o problema de HCDLP impraticável). Para m compostos, no entanto, o ataque GHS tem-se mostrado bem sucedido para uma pequena fração das curvas elípticas. O uso de outra metodologia para geraçao de curvas

hiper-elípticas na restrição de Weil, entretanto, pode fazer com que tal fração aumente. É aconselhável, portanto, evitar os corpos $\mathbb{F}_{2^m}$ com m composto.

6.2.7 Ataques colaterais

Os ataques listados anteriormente têm por objetivo a solução do problema do ECDLP, supondo que o adversário tenha acesso às chaves públicas e aos parâmetros de domínio. Eventualmente, ele também pode dispor de informações relativas à natureza das mensagens cifradas. Utilizando estes dados e métodos matemáticos, é feita a tentativa de quebrar a segurança do sistema criptográfico, seja resolvendo o problema ECDLP, seja fazendo uso de falhas na implementação do protocolo.

Informações adicionais que ajudem na quebra do sistema podem ser obtidas de outras formas, como através da potência consumida ou de radiação eletromagnética gerada durante a execução do protocolo. Exame do tempo gasto na execução de operações criptográfica ou respostas que o sistema dá na ocorrência (acidental ou proposital) de erros podem também revelar informações sobre a implementação do protocolo.

Tais ataques colaterais, em geral, não podem ser evitados simplesmente através de seleção criteriosa de parâmetros de domínio. Modificações sobre algoritmos, implementações em HW e SW estão entre as medidas mais eficazes para lidar com este tipo de ataque.

6.3 Robustez a Ataques

Para garantia de robustez a ataques, faz-se necessário contínuo acompanhamento das técnicas publicadas pelo meio acadêmico, indústria e organismos responsáveis por padrões no que concerne a fragilidades expostas pelos sistemas de criptografia de curvas elípticas. Desta forma, deve-se ter uma constante atualização dos critérios de exceção visando eliminar no processo de escolha de parâmetros aqueles que sabidamente resultem em sistemas criptográficos frágeis a ataques conhecidos.

A tabela 6.3 lista os ataques mais comuns e as medidas de prevenção, do ponto de vista de escolha de parâmetros, que podem ser tomadas para torná-los pouco eficazes. Nela consideramos que o sistema faz uso de uma curva elíptica E definida sobre um corpo finito, com ponto base $P \in E$ de ordem $n \geq 2^{160}$. Os ataques a canais colaterais não são considerados aqui porque, em geral, unicamente através da escolha de parâmetros não se consegue fazer prevenção eficiente.

Este capítulo listou uma série de ataques conhecidos contra sistemas ECC. Foi também

mostrado como se pode fazer uma seleção de parâmetros de domínio que aumente a robustez dos sistemas de criptografia a tais ataques. A seguir veremos como os padrões mais amplamente utilizados lidam com a questão de escolha de parâmetros.

Tabela 6.1: Classificações de Ataques

Ataque	Classificação
Pohlig-Hellman	Geral
Pollard	Geral
MOV	Específico
Anômalo	Específico
Logaritmos Múltiplos	Geral
Descida de Weil	Específico

Tabela 6.2: Classes de Curvas Supersingulares

Classe da Curva	n_1	t	k	c
I	$q+1$	0	2	1
II	$(q+1)/2$	0	2	2
III	$q+1 \mp \sqrt{q}$	$\pm\sqrt{q}$	3	$\sqrt{q} \pm 1$
IV	$q+1 \mp \sqrt{2q}$	$\pm\sqrt{2q}$	4	$q \pm \sqrt{2q}+1$
V	$q+1 \mp \sqrt{3q}$	$\pm\sqrt{3q}$	6	$(q+1)(q \pm \sqrt{3q}+1)$
VI	$\sqrt{q} \mp 1$	$\pm 2\sqrt{q}$	1	1

Tabela 6.3: Prevenções a ataques

Ataque	Prevenção
Pohlig-Hellman	n (ordem do ponto) deve ser primo.
Pollard	n tal que $\sqrt{n}$ implique em grande esforço de computação.
Logaritmos Múltiplos	n tal que $\sqrt{n}$ implique em grande esforço de computação.
MOV	n não pode ser divisor de $q^k - 1$, $1 \leq k \leq 20$
Descida de Weil	Evitar corpos binários compostos: $\mathbb{F}_{2^m}$, com m não-primo
Anômalo	$n \neq q$

Capítulo 7

Padrões em Criptografia sobre Curvas Elípticas

Neste capítulo veremos como os padrões para ECC mais amplamente utilizados lidam com a questão de escolha de parâmetros. As opções sugeridas serão discutidas sob os aspectos de segurança e desempenho.

7.1 P1363

O P1363 é uma padronização do IEEE (Instituto de Engenheiros Elétricos e Eletrônicos) para criptografia de chave pública. Há quatro áreas distintas cobertas por este padrão, como mostrado abaixo:

- P1363-2000, P1363A-2004: Criptografia de chave pública tradicional, incluindo esquemas para assinatura digital e troca de chaves, através de várias abordagens matemáticas.
  - Fatoração de Inteiros (RSA)
  - Logaritmo Discreto (Diffie-Hellman, DSA)
  - Logaritmo Discreto sobre Curvas Elípticas (MQV)
- P1363.1: Criptografia de chave pública baseada em reticulados, incluindo cifragem e assinatura digital (NTRUEnrypt, NTRUSign)
- P1363.2: Criptografia de chave pública baseada em senhas:
  - Troca de chaves autenticada por senha (EKE, SPEKE, SRP)

  - Esquemas de recuperação de chaves autenticada por senha (Ford e Kaliski)

- P1363.3: Criptografia de chave púbica baseada em identidade, utilizando emparelhamentos bilineares

Nesta seção serão cobertos os documentos P1363-2000 e P1363A-2004 no que se refere a seleção de parâmetros para uso em criptografia sobre curvas elípticas. Estes documentos têm por objetivo prover uma referência para especificações de técnicas comuns de criptografia de chaves públicas correntemente disponíveis para uso. Desta forma, tais técnicas são definidas numa forma estruturada que lhes permita ser selecionadas de acordo com as necessidades de uma aplicação particular. Para considerações sobre geração de números aleatórios, pode-se consultar a seção D.6 da especificação P1363-2000, ao passo que a seção A.15 do referido padrão discorre sobre testes de primalidade.

7.1.1 Modelo Hierárquico

As diferentes técnicas cobertas por este modelo são mostradas de acordo com um modelo abstrato de três camadas:

- Primitivas: são operações matemáticas básicas, geralmente relacionadas a problemas cuja resolução é difícil. Isoladamente não são capazes de prover segurança. Tal objetivo é de responsabilidade dos esquemas, dos quais são blocos construtores. Têm implementação de baixo nível, como aceleradores criptográficos ou módulos de software.

- Esquemas: são coleções de operações relacionadas, combinando primitivas e métodos adicionais. Eles fornecem mecanismos de segurança quando utilizados de forma adequada na construção de protocolos. Têm implementação de nível médio, como bibliotecas de serviços criptográficos.

- Protocolos: são seqüências de Primitivas e Esquemas a serem executadas por diferentes entidades para atingir algum propósito de segurança. Este objetivo, no entanto, depende de sua correta implementação. Têm implementação de alto nível, como conjuntos de aplicações.

7.1.2 Primitivas

O conjunto de primitivas cobertas por este padrão é dado por:

- Derivação de valores secretos (SVDP): são utilizadas como componentes dos esquemas de troca de chaves.
- Assinatura (SP) e verificação (VP): são blocos de construção dos esquemas de assinaturas.
- Cifragem (EP) e decifragem (DP): a partir delas são construídos os esquemas de cifragem.

Por si só, as primitivas não conseguem garantir segurança. Faz-se necessário que os esquemas dos quais fazem parte lidem com validação dos dados passados às primitivas para processamento. Elas são especificadas através do seguinte conjunto de informações:

- entradas;
- suposições relativas às entradas a serem utilizadas nas operações executadas pela primitiva;
- saídas;
- operações executadas pela primitiva, na forma de uma seqüência de passos;
- recomendações sobre a região de conformidade descrevendo os requisitos mínimos do conjunto de entradas sobre as quais uma dada implementação deve garantir.

7.1.3 Esquemas

Os esquemas incluem operações de gerenciamento de chaves, como seleção de chave privada e obtenção da chave pública da outra parte envolvida da comunicação. Para obter um grau satisfatório de segurança é necessário garantir a autenticidade das chaves e dos parâmetros de domínio utilizados. A geração de parâmetros de domínio e chaves também precisa ser feita e validada adequadamente para não haver prejuízo de segurança.

O conjunto de esquemas cobertos por este padrão é dado por:

- Esquema de troca de chaves (KAS): através dele duas entidades usam seus pares chaves públicas e privadas para negociar uma chave secreta compartilhada. Tal chave pode, por exemplo, ser utilizada em criptografia simétrica.
- Assinatura com apêndice (SSA): por meio deste esquema uma entidade assina o resumo de uma mensagem utilizando sua chave privada de forma verificável por qualquer outra entidade que tenha em mãos a mensagem, a assinatura e a chave pública de quem a efetuou.

- Cifragem (ES): o transmissor da mensagem utiliza a chave pública do receptor para cifrar a mensagem, e apenas este, fazendo uso de sua chave privada, pode decriptá-la. Estes esquemas também podem ser empregados na obtenção de chaves secretas a serem utilizadas em criptografia simétrica.

A especificação de esquemas é feita através do seguinte conjunto de informações:

- opções: lista as primitivas e métodos adicionais disponíveis;
- operações: dadas como série de passos a serem executados;
- recomendações sobre a região de conformidade descrevedo os requisitos que as implementações devem satisfazer.

7.1.4 Métodos Adicionais

O padrão P1363 utiliza os seguintes métodos adicionais:

- métodos de codificação de mensagem, que são parte dos esquemas de assinatura e cifragem:
  - codificação para assinaturas com apêndice (EMSA);
  - codificação para cifragem (EME);
- funções de obtenção e distribuição de chaves (KDF), utilizadas nos esquemas de troca de chaves;
- funções auxiliares:
  - funções geradoras de máscara (MGF), utilizada no método EME;
  - funçoes de Hash usadas nos métodos EMSA, EME, KDF e MGF.

7.1.5 Parâmetros de Domínio para Curvas Elípticas (EC)

Notação:

q: parâmetro de domínio que reflete o tamanho do corpo finito utilizado.

a, b: parâmetros de domínio pertencentes ao corpo finito que correspondem aos coeficientes que definem a curva elíptica.

E: curva elíptica definidas por (a, b) sobre o corpo finito.

$\#E$: número de pontos da curva elíptica

r: parâmetro de domínio correspondente a ordem do grupo G

G: domínio de parâmetro correspondente ao ponto da curva E gerador de um sub-grupo de ordem r.

k : cofator, definido por $\#E/r$

s, u, s', u': números inteiros correspondentes às chaves privadas associadas às chaves públicas $W, V, W'V'$, respectivamente

W, V, W', V': pontos da curva coorrespondentes às chaves públicas associados às chaves privadas s, u, s', u', respectivamente

$(s, W), (u, V)$: pares de chave, onde o primeiro elemento é a chave privada e o segundo, a pública.

z, z_1, z_2: elementos do corpo finito correspondentes aos valores secretos compartilhados obtidos a partir de primitivas.

K: chave secreta obtida a partir de esquema de negociação de chave

(c, d): par de inteiros correspondente a assinatura obtida via primitiva de geração de assinatura.

f: número inteiro representando a mensagem M, o qual foi obtido via operação de codificação

M: mensagem na forma de vetor de octetos.

7.1.5.1 Parâmetros de Domínio

Estes parâmetros são usados por cada primitiva e esquema e constituem um componente implícito das chaves. Este conjunto constitui-se de:

- corpo (primo ou binário);
- coeficientes (a, b) da curva elíptica, pertencentes ao corpo;
- número primo r que seja divisor do número de pontos da curva;
- um ponto da curva que tenha ordem r (denotado como gerador de um sub-grupo de ordem r);
- no caso de corpos binários, é necessário indicar qual representação será usada nas primitivas de conversão;
- cofator dado por $k = \#E/r$, onde $\#E$ é o número de pontos da curva;

- indicação se será necessário fazer validação da chave, ou se ECSVDP-DHC ou ECSVDP-MQVC serão usados. Caso afirmativo, $MDC(k, r) = 1$.

Dependendo do esquema e protocolo usados, uma dada entidade pode ter que gerar seus próprios parâmetros de domínio ou usar os parâmetros gerados por outros. Neste último caso, faz-se necessário determinar sua autenticidade. De acordo com esquema ou primitiva em execução, pode haver mais de um conjunto de parâmetros de domínio sendo usado, em funçaõ do número de chaves e dos requisitos associados. Ao contrário de chaves, os parâmetros de domínio em geral devem ser compartilhados e públicos. A segurança dos esquemas não se se baseia no segredo dos parâmetros de domínio.

7.1.5.2 Autenticação de Posse

É a garantia de que uma entidade está associada a uma chave pública e o conjunto correspondente de parâmetros de domínio. Opcionalmente inclui a garantia de que a entidade também possui a chave privada correspondente, evitando que um potencial oponente leve maliciosamente outros a acreditarem que os resultados de operações efetuadas com as chaves estejam associados a este ao invés do dono verdadeiro da chave privada. Esta operação pode fazer parte do processo de gerenciamento de chaves como parte de um certificado assinado por uma autoridade certificadora. Quem quer que conheça a chave pública da autoridade certificadora e confie nesta pode verificar a assinatura do certificado, e, por conseqüência, verificar que uma outra entidade está associada a uma dada chave. Estas autoridades certificadoras podem certificar outras autoridades, possibilitando que a confinça numa autoridade-raiz possa ser estendida a uma rede de autoridades.

7.1.5.3 Validação dos Parâmetros de Domínio

A maioria das primitivas especificadas neste padrão não tem comportamento definido quando o conjunto de parâmetros de domínio ou chaves são inválidos. As implementações devem dar tratamento adequado para tais casos, sob pena de ter seus esquemas e primitivas expostos a fragilidades de segurança. Tal validação pode ser feita explicitamente antes que os dados sejam passados como entrada para primitivas. Dado que uma entidade tem controle sobre sua chave privada, não é obrigatória a implementação de mecanismo para validá-la. Métodos para validação de chaves públicas não são cobertos por este padrão.

Alternativamente, os parâmetros de domínio e chaves podem ser validados (ou até mesmo gerados) por autoridades certificadoras. Alguns padrões (X9.62-1998 [B11], ANSI X9.63 [B12], NIST [B119], SEC 2) podem publicar conjuntos de domínios de parâmetros, servindo como entidade confiável que garante sua validade. Estes métodos garantem que as chaves geradas são válidas, mas não garantem que tal geração ocorreu de forma segura.

Isto pode ser obtido restrigindo a geração de chaves a determinados módulos previamente validados.

O conjunto de parâmetros de domínio pode ser gerado por uma das partes que pretenda usá-lo, por um terceiro ou por combinações destas alternativas. Caso algum parâmetro seja gerado aleatoriamente, faz-se necessário fornecer evidências verificáveis, como a semente utilizada e funções adequadas de hash. Isto ajuda a garantir que tais parâmetros não tenham sido escolhidos de um subconjunto particular que resulte em alguma fraqueza no aspecto de segurança.

O par de chaves {pública, privada} pode ser gerado pelo próprio dono ou por este em conjunto com uma autoridade certificadora. Quando há evidência de que a escolha foi aleatória, tem-se maior garantia de que as chaves não tenham sido escolhidas de forma a possuirem alguma propriedade que comprometa propositalmente a segurança. Além disso, isto evita a necessidade de verificação adicional de posse da chave privada. A semente utilizada na geração aleatória não pode ser fornecida, no entanto, pois revelaria o valor da chave privada.

Cada chave privada deveria ser gerada independentemente das demais. Se os números aleatórios utilizados neste processo tiverem sido gerados adequadamente, a probabilidade de que chaves privadas sejam compartilhadas torna-se extremamente baixa. Para evitar a possibilidade de que usuários desonestos façam algum arranjo para que suas chaves privadas sejam as mesmas com o propósito de repudiar suas assinaturas, pode-se deixar a tarefa de geração de chaves a cargo de uma autoridade confiável que garanta que duas chaves públicas jamais coincidam dentro de um dado sistema.

7.1.5.4 Algoritmo para Obtenção dos Parâmetros de Domínio

O algoritmo abaixo lista a seqüência de passos a ser seguida para obtenção de parâmetros de domínio

---

ALGORITMO: Obtenção dos parâmetros do domínio

ENTRADA:
- corpo finito de tamanho q (cujo valor é 2^m ou um número primo ímpar p);
- limites $[r_{min}, r_{max}]$ para a ordem de ponto-base;
- limite l_{max} divisor da ordem da curva;
- tipo de representação, caso corpo seja binário.
- indicação se parâmetros serão usados para validação de chave (ECSVDP-DHC, ECSVDP-MQVC)
- limiar B para teste de condição de MOV

SAÍDA:
- a, b, r, G, k(opcional)

1. Gerar coeficientes (a,b) de uma curva elíptica tal forma que ela possua um ponto G de ordem prima no intervalo [rmin, rmax], com cofator k.
2. Se q é primo e $q = r$ (ou seja,a curva é anômala e sujeita a ataques de redução), voltar para passo 1.
3. Se os parâmetros forem usados para validação de chave (ECSVDP-DHC ou ECSVDP-MQVC) e r for divisor de k, voltar para passo 1
4. Fazer verificação da condição de MOV.
4.1. $t \leftarrow 1$
4.2. para $i = 1, ..., B$ faça
4.2.1 $t \leftarrow tq \bmod r$
4.2.2 Se $t = 1$, voltar para passo 1.
5. Sair com a, b, r, G, k (se necessário)

7.1.5.5 Algoritmo para Validação de Parâmetros de Domínio

O algoritmo abaixo é sugerido pelo padrão P1363 para que seja assegurada a adequação dos parâmetros de domínio para uso nos diferentes esquemas e primitivas.

ALGORITMO: Validação dos Parâmetros de Domínio

ENTRADA:
- Parâmetros de domínio q (onde q é um primo ímpar ou 2^m), a, b, r, G;
- o cofator k (opcional);
- a representação dos elementos do corpo, caso este seja binário;
- indicação se parâmetors serão usados para validação de chave (ECSVDP-DHC ou ECSVDP-MQVC)
- indicação se curva foi gerada aleatoriamente de forma verificável e respectivos parâmetros
SAÍDA:
- Verdadeiro se parâmetros são válidos; falso, caso contrário

1. Se k não for fornecido então
1.1 Se $r \leq 4\sqrt{q}$ e os parâmetros forem usados para validação de chave (ECSVDP-DHC ou ECSVDP-MQVC) sair com resultado Falso.

1.2 Ir para passo 4.
2. Se $r < \sqrt{q} + 1$ e os parâmetros forem usados para validação de chave (ECSVDP-DHC ou ECSVDP-MQVC) e r dividir k, sair com resultado Falso.
3. Se cofator não for válido, sair com resultado falso (A12.3)
4. Se r não for primo > 2, sair com resultado Falso.
5. Se $q = p$, então
5.1 Se p não for primo > 2, sair com resultado Falso
5.2 Se uma das condições $0 \leq a < p$ e $0 \leq b < p$ for falsa, sair com resultado Falso.
5.3 Se não for verificada a aleatoriedade da curva, sair com Falso.
5.4 Se $4a^3 + 27b^2 \equiv 0 \mod p$, sair com resultado Falso.
5.5 Se $G = (x_G, y_G)$ não for diferente do ponto no infinito, sair com resultado Falso.
5.6 Se uma das condições $0 \leq x_G < p$ e $0 \leq y_G < p$ for falsa, sair com resultado Falso.
5.7 Se $y_G^2 \neq x_G^3 + ax_G + b \,(\mathrm{mod}\, p)$, sair com resultado Falso.
5.8 Se $rG \neq \infty$, sair com resultado Falso.
5.9 Se curva for anômala ou frágil a ataque MOV, sair com resultado Falso.
6. Se $q = 2^m$, então
6.1 verificar representação dos elementos do corpo
6.1.1 Se polinômio não for irredutível de grau m, sair com resultado falso.
6.1.2 Se representação for normal gaussiana tipo T e m for divisível por 8 ou não existir uma base para esta configuração, sair com resultado Falso.
6.1.3 Se representação for normal geral e o polinômio não for irredutível de grau m ou não for nornal, sair com resultado Falso.
6.2 Se a, b não forem elementos do corpo, sair com resultado falso.
6.3 Se a curva for aleatória e não for possível verificar a aleatoriedade, sair com falso.
6.4 Se $b = 0$, sair com resultado Falso.
6.5 Se $G = (x_G, y_G)$ ponto no infinito, sair com resultado Falso.
6.6 Se x_G, y_G não forem elementos do corpo, sair com resultado Falso.
6.7 Se $y_G^2 + x_G y_G \neq x_G^3 + ax_G^2 + b$ no corpo, sair com resultado Falso
6.8 Se $rG \neq \infty$, sair com resultado Falso.
6.9 Se curva for susceptível ao ataque MOV, sair com o resultado Falso.
7. Sair com resultado Verdadeiro.

7.1.5.6 Geração de Curvas Elípticas Aleatórias sobre Corpo Binário

O algoritmo abaixo gera aleatoriamente coeficientes de uma curva elíptica, além de ponto gerador de sub-grupo, a serem usados como parâmetros de domínio. Considera-se que a função HASH retorne um vetor de B bits, com $B \geq \left\lceil \log_2(r_{min}^{1/2}) \right\rceil$. O comprimento das entradas em bits satisfaz a restrição: $L \geq B$. Também devem ser consideradas as definições $s = \lfloor (m-1)/B \rfloor$ e $w = m - Bs$.

ALGORITMO: Geração dos coeficientes de uma curva elíptica

ENTRADA:
- Corpo $\mathbb{F}_{2^m}$;
- limites (r_{min}, r_{max}) para r (ordem do ponto-base);
- limite para divisor $l_{max} < r_{min}$

SAÍDA:
- parâmetros X, a, b, r, G. Onde X é a evidência de aleatoriedade

1. Escolher um vetor X arbritrário de comprimento L.
2. Calcular $h = \text{HASH}(X)$
3. $W_0 \leftarrow w$ bits mais à direita de h
4. Converter o vetor X para um inteiro z
5. Para $i = 1, ..., s$ faça:
5.1 Converter $(z + i) \bmod 2^L$ num vetor X_i de compriento L.
5.2 $W_i \leftarrow \text{HASH}(X_i)$.
6. $W \leftarrow$ Concatenação$(W_0, W_1, ..., W_s)$
7. Converter W no elemento b pertencente ao corpo binário.
8. Se $b = 0$ voltar para passo 1.
9. Escolher a arbitrariamente.
10. Calcular a ordem u da curva elíptica E sobre $\mathbb{F}_{2^m}$ dada por $y^2 + xy = x^3 + ax^2 + b$.
11. Se u não for primo (ou quase-primo), voltar para passo 1.
12. Se u não satisfizer a condição $u = kr$, r primo $r_{min} < r < r_{max}$, voltar para passo 1.
13. Escolher um ponto G de ordem r sobre E .
14. Sair com (X, a, b, r, G)

7.1.5.7 Verificação de Aleatoriedade de Curvas Elípticas sobre Corpo Binário

O algoritmo abaixo verifica a validade de curva elíptica sobre corpo binário passados como parte dos parâmetros de domínio, bem como a aleatoriedade na geração da curva.

ALGORITMO: Verificação de curva gerada aleatoriamente

ENTRADA:
- Vetor X de comprimento em bits dado L
- parâmetros $\mathbb{F}_{2^m}, a, b, r, G = (x_G, y_G)$

SAÍDA: Verdadeiro, caso a curva seja válida e aleatoriedade da geração possa ser verificada; falso, caso contrário.

1. Compute $h \leftarrow \text{HASH}(X)$
2. $W_0 \leftarrow$vetor formado pelos w bit mais à direita de h
3. Converter X num inteiro z
4. Para $= 1, ..., s$, faça:
4.1 Converter o inteiro $(z + i)$ mod 2^L num vetor X_i de L bits.
4.2 $W_i \leftarrow \text{HASH}(X_i)$
5. $W \leftarrow$ concatenação$(W_0, W_1, ..., W_s)$
6. Converter o vetor W de m bits num elemento b' do corpo $\mathbb{F}_{2^m}$
7. Se $b = 0$ sair com Falso.
8. Se $b \neq b'$ sair com Falso.
9. Se $G \neq \infty$ sair com Falso.
10 . Se $y_G^2 + x_G y_G \neq x_G^3 + a x_G^2 + b$, sair com Falso.
11. Se $rG \neq \infty$, sair com Falso.
12. Sair com Verdadeiro.

7.1.5.8 Construção de Curvas Elípticas Aleatórias sobre Corpo Primo

O algoritmo a seguir produz um conjunto de parâmetros para curva elíptica sobre um corpo primo $\mathbb{F}_p$ bem como informação suficiente para que se possa verificar que a geração da curva foi aleatória. O ponto-base procurado deve ter ordem pertencente ao intervalo $[r_{min}, r_{max}]$, a função HASH utilizada tem saída de tamanho B bits, com $B \geq \left\lceil \log_2(r_{min}^{1/2}) \right\rceil$. Sua entrada tem tamanho $L \geq B$. Considerar ainda a seguinte notação $v = \lceil \log_2 p \rceil$, $s = \lfloor (v-1)/B \rfloor$, $w = v - Bs - 1$.

ALGORITMO: Geração aleatória de parâmetros de uma curva elíptica

ENTRADA:
- p , número primo que define o corpo

- (r_{min}, r_{max}), limites para a ordem do ponto-base
- $l_{max} < r_{min}$, limite máximo para divisor
SAÍDA:
- vetor X, a ser usado para verificação de aleatoriedade na geração da curva
- (a, b) , coeficientes da curva
- G , ponto-base
- r , ordem do ponto-base

1. Escolher arbitrariamente um vetor X de comprimento L em bits.
2. $h \leftarrow \text{HASH}(X)$
3. $W_0 \leftarrow$vetor formado pelos w bits mais à direita de h.
4. Converter o vetor X de L bits num inteiro z.
5. Para $i = 1, ..., s$ faça:
5.1 Converter o inteiro $(z + i) \bmod 2^L$ num vetor X_i de L bits
5.2 $W_i \leftarrow \text{HASH}(X_i)$
6. $W \leftarrow$concatenação $(W_1, W_2, ..., W_s)$
7. Converter o vetor de W de comprimento $(v - 1)$ num inteiro c.
8. Se $c = 0$ ou $4c + 27 \equiv 0 \,(\text{mod}\, p)$ voltar para o passo 1.
9. Escolher inteiros $a, b \in \mathbb{F}_p$ tais que $cb^2 \equiv a^3 \,(\text{mod}\, p)$
10. Calcular a ordem u da curva elíptica E sobre $\mathbb{F}_p$ dada por $y^2 = x^3 + ax + b$
11. Se u não for primo (ou quase-primo) voltar para o passo 1. Devemos ter $u = kr$, com r primo, com $r_{min} < r < r_{max}$
13. Selecionar um ponto G de ordem r sobre E.
14. Sair com X, a, b, r, G

7.1.5.9 Verificação de Curvas Aleatoriamente Geradas sobre Corpo Primo

O algoritmo abaixo faz a verificação da validade dos parâmetros de uma curva que tenha sido aleatoriamente gerada, bem como do ponto-base selecionado.

ALGORITMO: Verificação de curva elíptica gerada aleatoriamente

---

ENTRADA:
- vetor X de comprimento L bits
- $q, a, b, r, G = (x_G, y_G)$
SAÍDA: Verdadeiro, caso os parâmetros tenham passado no teste; Falso, caso contrário.

1. Compute $h \leftarrow \text{HASH}(X)$

2. $W_0 \leftarrow$vetor formado pelos w bits mais à direita de h
3. Converter o vetor X num número inteiro z
4. Para $i = 1, ..., s$ faça:
4.1 Converter o inteiro $(z + i) \bmod 2^L$ num vetor X_i de comprimento L bits.
4.2 $W_i \leftarrow \text{HASH}(X_i)$
5. $W \leftarrow$concatenação$(W_0, W_1, ..., W_s)$
6. Converter o vetor W num inteiro c
7. Se $c = 0$ retornar Falso
8. Se $(4c + 27) \bmod p = 0$ retornar Falso
9. Se $cb^2 \neq a^3 \bmod p$ retornar Falso
10. Se $G = \infty$ retornar Falso
11. Se $y_G^2 \neq x_G^3 + ax_G + b \bmod p$ retornar Falso
12. Se $rG \neq \infty$ retornar Falso.
13. Retornar Verdadeiro.

7.2 ANSI X9.62 / 2005

Este padrão define métodos para geração e verificação de assinatura digital para proteção de mensagens e dados através do uso do algoritmo de assinatura digital de curvas elípticas (ECDSA). Este algoritmo é o análogo sobre curvas elípticas do DSA (ANS X9.30). Deve ser usado em conjunto com funções hash aprovadas pelo item 00003 do Registro X9, também conhecido por Padrão Seguro de Hash (SHS). O conjunto de tais funções é dado por {SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512}.

O padrão ECDSA provê métodos e critérios para geração de chaves públicas e privadas, bem como mecanismos de controle necessários para uso seguro de tais chaves em algoritmos. Além disso, fornece métodos e critérios para geração de parâmetros de domínio para curvas elípticas, bem como mecanismos de controle para utilização adequada de tais parâmetros. O anexo D do referido padrão discorre sobre geração de números aleatórios, ao passo que a anexo A lida sobre geração de números primos.

Quando implementadas de forma apropriada, as técnicas descritas neste padrão possibilitam garantir:

- integridade de dados;
- autenticação da origem dos dados;
- irrevogabilidade do conteúdo da mensagem e de sua origem.

7.2.1 Níveis de Segurança

Nível de segurança é expresso em bits e representa a quantidade de operações que um adversário precisa executar para comprometer a segurança de um dado sistema, com probabilidade acima de 50%. Desta forma, um nível de segurança de 128 bits implica que o adversário deveria executar 2^{127} operações para quebrar a segurança do sistema de criptografia. O aumento de 1 bit de segurança implica que os esforços do adversário precisam dobrar para comprometer a segurança do sistema.

O conjunto dos níveis de segurança em bits aprovados por este padrão é dado por {80, 112, 128, 192, 256}. O nível de segurança de um dado sistema é dado pelo nível do componente mais fraco que o compõe.

7.2.2 Preparação do Sistema

A entidade responsável pela execução da assinatura e a responsável pela verificação desta devem executar os passos abaixo para estarem em conformidade com o ECDSA:

- a entidade que assina a mensagem deve estabelecer um conjunto de parâmetros de domínio consistindo de $\{q, a, b, x_G, y_G, n, semente, h, base\}$.
  - $q = 2^m$ para corpos binários ou p primo para corpos primos
  - $\{a, b\}$ são elementos do corpo correspondentes aos coeficientes da curva elíptica
  - (x_G, y_G) são as coordenadas do ponto-base, pertecente à curva. Tal ponto é o gerador do sub-grupo de pontos da curva a serem usados pelo sistema.
  - n é a ordem do sub-grupo gerado por (x_G, y_G)
  - *semente* (opcional) corresponde a entrada para função de hash h para o caso de curvas elípticas aleatoriamente geradas
  - h (opcional) é a função de hash utilizada na geração aleatória da curva elíptica.

Tais parâmetros devem ter sido gerados de acordo com os métodos permitidos por este padrão. Caso a entidade que executa a assinatura não tenha gerado os parâmetros, é necessário que ela os valide antes de usá-los. Os parâmetros devem fornecer o nível de segurança requerido pelo esquema.

- A entidade que gera a assinatura deve selecionar uma função de hash adequada ao nível de segurança desejado pelo processo.

- Chaves privadas temporárias e estáticas devem ser criadas usando gerador de números aleatórios aprovado pelo padrão, oferecendo nível de segurança compatível com o requerido pelo sistema. As chaves privadas estáticas devem ser geradas pela entidade que assina ou por um terceiro em quem se confie. As chaves privadas temporárias devem ser geradas apenas pela entidade que assina.

- A entidade que gera a assinatura deve ter obtido um par de chaves (d, Q) associadas aos parâmetros de domínio mencionados no primeiro item. O mecanismo de geração das chaves deve ser compatível com o descrito pelo padrão, usando gerador de números aleatórios. É necessário assegurar validade do par de chaves e posse da chave privada.

- A entidade que verifica a assinatura deve obter de forma autenticada os parâmetros de domínio, a função de hash para resumo da mensagem e a chave pública a ser empregada na assinatura. As seguintes verificações são necessárias:

  – validade dos parâmetros de domínio;

  – validade da chave pública;

  – garantia de que a chave privada pertença à entidade que gera a assinatura

7.2.3 Execução de Assinatura

O algoritmo abaixo deve ser seguido pelo originador da mensagem ao assiná-la:

ALGORITMO: Geração de Assinatura

ENTRADA:
- vetor de bits M correspondente à mensagem a ser assinada;
- os parâmetros de domínio da curva elíptica;
- função de hash a ser utilizada;
- chave privada a ser utilizada na assinatura;

SAÍDA:
- Par de inteiros (r, s) no intervalo $[1, n-1]$

1. Gerar um par temporário de chaves (k, R), com $R = (x_{R,}, y_R)$, associados aos parâmetros de domínio;
2. Converter o elemento x_R num inteiro j;
3. $r \leftarrow j \bmod n$. Se $r = 0$ voltar para passo 1.

4. $H \leftarrow$HASH(M).
5. Se $\lceil \log_2 n \rceil \geq$comprimento em bits de H'
5.1 então $E \leftarrow H$
5.2 senão $E \leftarrow$ vetor dos $\lceil \log_2 n \rceil$ bits mais à esquerda de H
6. Converter E num inteiro e
7. $s \leftarrow k^{-1}(e + dr) \bmod n$. Se $s = 0$, voltar para passo 1.
8. (opcionalmente) se (r, s) não for uma assinatura válida, voltar para passo 1.
9. Sair com (r, s)

7.2.4 Verificação de Assinatura / Chave Pública

Este algoritmo é utilizado pelo receptor da mensagem com o propósito de verificar a assinatura do transmissor.

ALGORITMO: Verificação de Assinatura

ENTRADA:
- Mensagem recebida M';
- Assinatura dada por (r', s');
- Parâmetros de domínio da curva elíptica;
- Função hash utilizada na assinatura;
- Chave pública Q da entidade que transmitiu e assinou a mensagem

SAÍDA:
- Verdadeiro, caso a assinatura seja válida; Falso, caso contrário;

1. Se r' ou s' não estiverem no intervalo $[1, n-1]$ sair com Falso.
2. $H' \leftarrow$HASH(M')
3. Se $\lceil \log_2 n \rceil \geq$comprimento em bits de H'
3.1 Então $E' \leftarrow H'$
3.2 Senão $E' \leftarrow$ os $\lceil \log_2 n \rceil$ bits mais à esquerda de H'.
4. Converter o vetor E' num inteiro e'
5. $u_1 \leftarrow e'(s')^{-1} \bmod n$
6. $u_2 \leftarrow r'(s')^{-1} \bmod n$
7. $R \leftarrow u_1 G + u_2 Q$
8. Se $R = \infty$ sair com Falso.
9. Dado que $R = (x_R, y_R)$, converter x_R num inteiro j
10. $v \leftarrow j \bmod n$
11. Se $v \neq r'$ sair com Falso.

12. Sair com Verdadeiro.

7.2.5 Verificação de Assinatura / Chave Privada

Este algoritmo é utilizado pelo transmissor da mensagem com o propósito de verificar sua própria assinatura.

ALGORITMO: Verificação de assinatura

ENTRADA:
- Mensagem M'
- Assinatura (r', s')
- Parâmetros de domínio da curva elíptica
- Chave privada d do transmissor da mensagem

SAÍDA:
- Verdadeiro, caso a verificação tenha sido bem sucedida; Falso, caso contrário.

1. Se r', s' não estiverem no intervalo $[1, n-1]$ sair com Falso
2. $H' \leftarrow$HASH(M')
3. Se $\lceil \log_2 n \rceil \geq$ comprimento de H' em bits
3.1 Então $E' \leftarrow H'$
3.2 Senão $E' \leftarrow$ os $\lceil \log_2 n \rceil$ bits mais à esquerda de H'
4. Converter E' num inteiro e'
5. $u_1 \leftarrow e'(s')^{-1} \bmod n$
6. $u_2 \leftarrow r'(s')^{-1} \bmod n$
7. $k' \leftarrow u_1 + u_2 d \bmod n$
8. $R \leftarrow k'G$
9. Se $R = \infty$ sair com Falso
10. Dado que $R = (x_R, y_R)$, converter x_R num inteiro j
11. $v \leftarrow j \bmod n$
12. Se $v \neq r'$ sair com Falso
13. Sair com Verdadeiro.

7.2.6 Algoritmo para Geração de Curvas Elípticas Aleatórias

O algoritmo abaixo utiliza funções de hash com auxiliares na geração aleatória de parâmetros de curvas elípticas:

ALGORITMO: Geração de parâmetros de uma curva elíptica

ENTRADA:
- Semente h a ser usada como entrada da função de hash cuja entrada tem comprimento de t bits
- tamanho do corpo q

SAÍDA:
- Parâmetros a, b definindo a curva elíptica, ou Falso em caso de falha.

1. $m \leftarrow \lceil \log_2 q \rceil$
2. $s \leftarrow \lfloor (m-1)/t \rfloor$
3. $k \leftarrow m - st$ se o corpo for binário ou $k \leftarrow m - st - 1$ se o corpo for primo
4. $H \leftarrow$HASH(h)
5. Converter H num inteiro e
6. $c_0 = e \bmod 2^k$
7. Para $j = 1, ..., s$ faça $c_j \leftarrow$HASH$(h + j \bmod 2^q)$
8. $c \leftarrow c_0 2^{ts} + c_1 2^{t(s-1)} + ... + c_s$
9. Converter c num elemento do corpo r
10. Escolher arbitrariamente um elemento a do corpo F_q
11. Para corpo binário, se $r \neq 0$ então $b \leftarrow r$, senão sair com Falso.
12. Se corpo for primo, então

12.1 Encontrar b pertencente ao corpo finito, tal que $b^2 r = a^3$. Caso tal b não exista, sair com Falso.

12.2 Se $4a^3 + 27b^2 = 0$ sair com Falso.

13. Sair com (a, b)

7.2.7 Condições Necessárias para Curvas Elípticas Seguras

7.2.7.1 Condição de MOV

O ataque MOV reduz o problema do logaritmo discreto numa curva elíptica sobre o corpo F_q ao problema de logaritmo discreto sobre a extensão F_{q^B}, para $B \geq 1$, que tem complexidade sub-exponencial. O ataque é prático apenas se B for pequeno. Para a maior parte das curvas elípticas, no entanto, tal redução não é possível. A condição de MOV tem por objetivo assegurar que uma dada curva não é vulnerável a este tipo de ataque. Antes de aplicar o algoritmo de verificação, é necessário selecionar um limiar de MOV, que consiste num inteiro positivo B tal que a extração de logaritmos sobre F_{q^B} seja tão difícil quanto calcular logaritmo discreto numa curva elíptica sobre F_q. O padrão X9.62

requer $B \geq 100$. Tal valor de B elimina curvas super-singulares.

ALGORITMO: Verificação da condição de MOV

ENTRADA:
- Limiar de MOV B;
- Potência de número primo q;
- Número primo n tal que n seja divisor de $\#E(F_q)$, número de pontos da curva elíptica E definida sobre o corpo F_q;

SAÍDA:
- Verdadeiro, se o teste passar; Falso, caso contrário.

1. $t \leftarrow 1$
2. Para $i = 1, ..., B$ faça:
2.1. $t \leftarrow tq \bmod n$
2.2. Se $t = 1$ sair com Falso.
3. Sair com Verdadeiro.

7.2.7.2 Condição Anômala

As curvas elípticas anômalas são curvas para as quais o número de pontos coincide com o número de elementos do corpo finito sobre o qual está definida. [11] e [59] mostraram que consegue-se determinar isomorfismos sobre tais curvas de tal forma a resolver o problema de logaritmo discreto em tempo sub-exponencial. Para evitar curvas vulneráveis a este ataque, basta testar o número de pontos contra o tamanho do corpo. Se forem iguais, a curva deve ser descartada.

7.2.8 Validação de Curvas Elípticas

O algoritmo abaixo retorna Verdadeiro, caso a curva elíptica a ser verificada seja válida, e Falso, caso contrário.

ALGORITMO: Validação de parâmetros de uma curva elíptica

ENTRADA:
- Corpo finito F_q
- Parâmetros da curva (a, b)
- Opcionalmente, semente h para uso em função de hash, caso a aleatoriedade da curva precise ser verificada

SAÍDA:
- Verdadeiro, caso a curva elíptica seja válida; falso, caso contrário.

1. Se o q for ímpar e não-primo, sair com Falso.
2. Se $q = 2^m$ e m não for primo, sair com Falso.
3. Se a ou b não forem elementos do corpo F_q, sair com Falso.
4. Se a curva elíptica for singular (ou seja, $4a^3 + 27b^2 \equiv 0 \pmod q$ para corpo primo, ou $b = 0$ para corpo binário), sair com Falso.
5. Se a semente h for fornecida e os coeficientes gerados de acordo com o algoritmo do item 7.2.6 não coincidirem com (a, b) sair com Falso.
6. Sair com Verdadeiro.

7.2.9 Parâmetros de Domínio

Este padrão recomenda o uso dos quinze conjuntos de parâmetros de domínio listados pelo padrão FIPS 186-2. Para cada um dos cinco níveis de segurança, há três conjuntos definidos. Além destes, o usuário tem a liberdade de gerar seus próprios parâmetros utilizando os métodos descritos pelo padrão X9.62.

Os parâmetros de domínio para curvas elípticas, segundo este padrão, consistem de: tamanho do corpo finito q, indicação de base utilizada para corpos binários, semente utilizada na geração aleatória de curvas elípticas, parâmetros (a, b) que definem a curva elíptica E, ponto $G(x_G, y_G)$ sobre E de ordem prima n, cofator h.

- Escolha da base: quando o corpo finito escolhido é binário $\mathbb{F}_{2^m}$, a forma de se interpretar os vetores de bits correspondentes aos elementos deste corpo é determinada pela base, a qual pode ser polinomial ou normal. Não há aspectos de segurança associados à escolha. Externamente, todos os usuários de um dado conjunto de parâmetros de domínio utilizam a mesma base, embora internamente sejam livres para utilizar a representação que desejarem, desde que produza resultados equivalentes.

- Função canônica de hash: utilizada na determinação aleatória opcional dos parâmetros (a, b) de definição da curva elíptica, bem como na determinação do ponto-base

G. O uso desse tipo de fução contribui para a diminuição da probabilidade de uso ataques contra grupos especiais de curva. Entretanto, o uso de algumas curvas específicas (como as de Koblitz) pode representar melhoria no desempenho. Por este motivo, o padrão X9.62 permite os dois tipos de escolha.

- Escolha do ponto-base G : desde que sua ordem seja prima, não há restrições conhecidas de segurança. Entretanto, todos os usuários de um dado conjunto de parâmetros de domínio devem usar o mesmo ponto-base. Para diminuir possíveis vulnerabilidades a ataques ou erros de implementação, a escolha pode ser feita de forma aleatória. Por exemplo, se não se faz a verificação do valor de r de forma a ter-se $r \neq 0$ na assinatura (r, s), um adversário poderia escolher G de tal forma a gerar uma assinatura inválida $(0, s)$ para qualquer mensagem. Com isto, ele pode repudiar a assinatura. Se a escolha de G for aleatória, entretanto, tem-se mais robustez a este tipo de ataque.

- Validação de parâmetros de domínio: o gerador dos parâmetros de domínio de curvas elípticas deveria assegurar que eles satisfazem os critérios listados nos anexo A.3 e C.1 deste padrão. Todos os que utilizam tais parâmetros tambem deveriam testar sua validade, segundo os mesmos critérios, dependendo do grau de confiança em quem gerou tais parâmetros. A garantia de validade destes parâmetros deve ser obtida por seus usuários, sob pena de tornarem-se vulneráveis aos tipos de ataques previnidos pelo padrão.

- Período de validade dos parâmetros de domínio: um usuário pode utilizar um dado conjunto de parâmetros de domínio para gerar um ou vários pares de chaves. A quantidade de usuários compartilhando um dado conjunto e o número pares de chaves que cada um é autorizado a gerar é uma decisão de política de segurança. Assim como um par de chaves tem um período considerado adequado para seu uso, o mesmo pode ser dito em relação a um conjunto de parâmetros de domínio. Uma vez que tenha sido quebrada a segurança de um par de chaves, a dificuldade de se quebrar a segurança dos pares restantes diminui. Portanto, o padrão deve assegurar que a primeira quebra seja tão difícil quanto possível. Para um dado grau de segurança s , é obrigatório que o ponto-base tenha ordem maior que 2^{2s-1} .

- Limiar de MOV: é um número inteiro B tal que a obtenção de logaritmos discretos sobre corpo F_{q^B} seja pelo menos tão difícil quanto a obtenção de logaritmos discretos sobre a curva elíptica $E(F_q)$. Este padrão exige que $B \geq 100$.

- Determinação da ordem n do ponto-base G: este padrão requer que, para um dado nível de segurança s a ordem n do ponto-base seja maior que r_{min} (dado pelo

usuário), 2^{160} e 2^{2s-1}. Além disso, a curva elíptica deve ser tal que sua ordem seja dada por $u = hn$, onde h é chamado cofator, com fator-primo no máximo dado por l_{max} fornecido pelo usuário.

- Compressão de pontos: o padrão permite que os pontos sejam representados na forma comprimida, não-comprimida e híbrida.
- Corpos binários de grau composto: os corpos da forma $\mathbb{F}_{2^m}$ devem necessariamente ter m primos para evitar ataques baseados na descida de Weil.

7.2.10 Geração de Parâmetros de Domínio

O algoritmo abaixo gera parâmetros de domínio a partir do nível de segurança desejado, assegurando que as curvas geradas não sejam sucsceptíveis aos ataques anômalo e MOV.

ALGORITMO: Geração de parâmetros de domínio

ENTRADA:
- indicação da necessidade de se gerar a curva de forma aleatória
- indicação da necessidade de se escolher o ponto-base de forma aleatória
- tipo de corpo finito desejado (binário, primo)
- nível se segurança s

SAÍDA:
- tamanho do corpo q
- representação, para o caso de corpos binários
- coeficientes da curva (a, b)
- ponto-base G e respectiva ordem n
- cofator
- função de hash e sementes utilizadas na geração aleatória de parâmetros

1. Se for desejado gerar uma curva ou ponto-base aleatórios, selecionar um valor para a semente h
2. Selecionar o tamanho q do corpo de acordo com a indicação do tipo desejado (primo ou binário) e do nível de segurança. A condição $q \geq 2^{2s-1}$ deve ser satisfeita,
3. Gerar os parâmetros de uma curva elíptica de acordo com o algoritmo do item 7.2.6.
4. Se a geração da curva resultar em falha, votar ao passo 1.
5. Se a ordem da curva não for prima (ou quase prima) voltar ao passo 1.
6. Selecionar ponto-base G pertencente à curva, com ordem prima superior a 2^{2s-1}.
7. Se os parâmetros gerados não forem válidos, voltar ao passo 1.

8. Sair com q, representação (se corpo for binário), coeficientes da curva (a, b), ponto-base G e respectivas ordem n e cofator c, função de hash utilizada, indicação de aleatoriedade na geração dos parâmetros da curva e/ou ponto-base juntamente com as sementes utilizadas.

7.2.11 Verificação de Parâmetros de Domínio

O algoritmo abaixo gera parâmetros de domínio a partir do nível de segurança desejado, assegurando que as curvas geradas não sejam sucsceptíveis aos ataques anômalo e MOV.

ALGORITMO: Verificação de parâmetros de domínio

ENTRADA:
- nível de segurança s
- corpo finito q sobre o qual a curva elíptica E está definida
- coeficientes (a, b) da curva elíptica
- tipo de representação (para corpos binários)
- ponto-base G e respectiva ordem n
- cofator c
- indicação da necessidade de se verificar a aleatoriedade na geração da curva
- função de hash e sementes utilizadas na geração da curva e na escolha do ponto-base.

SAÍDA:
- Verdadeiro, se parâmetros forem válidos; Falso, caso contrário.

1. Se $n < 2^{160}$ ou $n < 2^{2s-1}$ sair com Falso.
2. Se n não for primo, sair com Falso.
3. Se a semente e função de hash não reproduzirem os coeficientes da curva, sair com Falso.
4. $c' \leftarrow \lfloor (\sqrt{q} + 1)^2 / n \rfloor$
5. Se o cofator c não coincidir com c' sair com Falso.
6. Se $c' > 2^{s/8}$ sair com falso
7. Se a condição de MOV falhar, sair com Falso.
8. Se a condição anômala falhar, sair com Falso.
9. Se o ponto-base não corresponder à semente e função de hash passadas como entrada, sair com Falso.
10. Sair com Verdadeiro.

7.3 ANSI X9.63 / 2001

Este padrão especifica esquemas de estabelecimento de chaves assimétricas, tanto sob aspecto de negociação quanto de transporte. Os esquemas utilizam operações aritméticas sobre um grupo de pontos de uma curva elíptica definida sobre um corpo finito. Para geração de números pseudo-aleatórios deve-se consultar a seção A.4 deste padrão, ao passo que testes de primalidade são cobertos pelo anexo A.2.

Duas entidades quaisquer U e V conseguem derivar uma chave simétrica empregando os protocolos definidos neste padrão, utilizando como base esquema de estabelecimento de chaves assimétricas. Cada entidade possui seu próprio par de chaves (pública, privada). Quando executam o esquema apropriado, conseguem derivar um valor conhecido por ambos, a partir do qual a chave simétrica pode ser obtida. O método utilizado neste último passo não faz parte do escopo do esquema.

7.3.1 Parâmetros de Domínio

Este padrão segue regras semelhantes às do X9.62 na obtenção e verificação de parâmetros de domínio. As mesmas questões relativas a segurança discutidas na seção anterior aplicam-se a esta.

7.4 WAP WTLS

O padrão WAP ("Wireless Application Protocol") tem por objetivo definir um conjunto de especificações a serem usadas por aplicações que operem sobre redes de comunicação sem fio. Ele tem arquitetura hierárquica e abrange transporte, segurança, transação, sessão e aplicação. Geração de números aleatórios e primos é coberta pela seção 11 do referido padrão.

A camada de segurança é chamada WTLS ("Wireless Transport Layer Security") e opera sobre a camada de transporte. É modular e depende do nível de segurança requerido, provendo as camadas superiores do WAP com interface de transporte seguro. Entre suas responsabilidades está o gerenciamento de conexões seguras. Além disso, assegura privacidade, integridade de dados e autenticação entre as duas aplicações que se comuinicam. A funcionalidade que provê é semlhante à de TLS 1.0, incorporando algumas características novas, tais como suporte a datagrama, protocolo de "handshake" otimizado e atualização dinâmica de chaves. Este padrão foi desenvolvido tendo em vista redes de banda pequena e tempos longos de latência.

7.4.1 Parâmetros de Domínio

De forma bastante semelhante aos padrões listados anteriormente, os seguintes elementos são definidos:

- *corpo*: identifica o tipo de corpo finito sobre o qual a curva elíptica será definida;
- *primo_p*: número primo ímpar, para o caso de corpos primos;
- m: para corpos binários $\mathbb{F}_{2^m}$, define o grau da característica;
- k: para representação polinomial de corpos binários para os quais o polinômio redutor tem forma de trinômio $x^m + x^k + 1$;
- k_1, k_2, k_3: para representação polinomial de corpos binários para os quais o polinômio redutor tem forma de pentatômio $x^m + x^{k_3} + x^{k_2} + x^{k_1} + 1$;
- a, b: correspondentes aos coeficientes da curva elíptica;
- P: ponto-base da curva elíptica;
- n: ordem do ponto-base;
- h: cofator;

7.4.2 Utilização dos Padrões P1363 / X9.62 / SEC 1 / SEC 2

- Os cálculos sobre curva elíptica utilizados na versão EC Diffie-Hellman são os especificados pelo padrão P1363, empregando-se a primitiva ECSVDP-DH para geração de valores secretos.
- Os parâmetros de domínio podem ser transmitidos explicitamente ou pode ser feito uso de valores pre-definidos.
- Os pontos da curva elíptica são representados como vetores de octetos gerados pelas rotinas de conversão dos padrões X9.62 ou SEC 1, com a possiblidade de uso de compressão de pontos.
- A representação dos coeficientes da curva elíptica (a, b) usa convenção do padrão X9.62
- A assinatura ECDSA e sua verificação seguem a recomendação P1363 para assinaturas com apêndice. As rotinas de conversão de dados utilizadas são as descritas no padrão X9.62.

Além do uso de curvas abitrariamente geradas pelo usuário, este padrão possibilita o uso de curvas pre-definidas. Um total de 12 curvas são listadas, juntamente com ponto-base. Algumas dela são comuns a FIPS 186-2 , SEC 2 e X9.62 como mostrado na tabela 7.1.

7.5 FIPS 186.2 (DSS)

Acrônimo de "Digital Signature Standard", este padrão especifica algoritmos para aplicações que necessitem assinaturas digitais. Tais assinaturas utilizam-se de um conjunto de regras e parâmetros para assegurar a integridade dos dados assinados e a identidade de quem os assinou. Há algoritmos especificados para geração (através de chave privada) e verificação (através de chave pública) de assinaturas. Cada par de chaves está associado a uma entidade. A chave pública é de conhecimento geral, ao passo que a chave pública é de conhecimento restrito a seu dono. Para obtenção de formas condensadas da mensagem é feito uso de funções de hash, especificadas no padrão FIPS 180-1.

Este padrão inclui três especificações: DSA, RSA e ECDSA.

A versão para curvas elípticas ECDSA do algoritmo de assinatura digital DSA é especificada no padrão X9.62. As curvas elípticas utilizadas pelo FIPS 186.2 são sujeitas à aprovação do governo norte-americano e listadas no apêndice 6 da especificação. Os pontos-base fornecidos como exemplo não são de uso obrigatório. O usuário é livre para gerar os seus, se assim o desejar. A geração de números aleatórios é coberta no apêndice 3, ao passo que a de números primos é abordada no apêndice 2 do referido padrão.

7.5.1 Parâmetros de Domínio

Dentre os parâmetros de domínio, os mais relevantes correspondem à curva elíptica E e ao ponto-base pertencente à curva. Este deve ter ordem prima r bastante elevada. A ordem da curva deve ser da forma $n = fr$, onde f é o cofator, o qual não pode ser divisível por r. Por questões de desempenho, é desejável que f seja tão pequeno quanto possível. As curvas listadas pelo padrão têm cofator 1, 2 ou 4. Isto faz com que as chaves públicas tenham aproximadamente o mesmo comprimento.

Tantos corpos primos quanto binários são permitidos pelo padrão. No caso binário, não há exigência de que o expoente da característica seja primo, ao contrário do padrão X9.62. Para tais corpos, tanto bases polinomiais quanto normais são permitidas. Dois tipos de curvas são utilizadas neste padrão: aleatoriamente geradas e curvas especiais (Koblitz).

7.5.2 Curvas sobre corpos primos $\mathbb{F}_p$

A equação é dada por $y^2 = x^3 - 3x + b \pmod{p}$, com ordem prima r (e, portanto, cofator 1) e seguintes parâmetros associados:

- módulo p
- ordem r
- semente s usada com função SHA-1(para geração pseudo-aleatória de curvas), resultando em c
- coeficiente b satisfazendo $b^2c \equiv -27 \pmod{p}$
- Ponto-base $G = (x_G, y_G)$

7.5.3 Curvas sobre corpos binários $\mathbb{F}_{2^m}$

Para cada grau m é dada uma curva que tenha sido aleatoriamente gerada, da forma $y^2 + xy = x^3 + x^2 + b$, e uma curva de Koblitz da forma $y^2 + xy = x^3 + ax^2 + 1$, com $a \in \{0, 1\}$. As curvas aleatórias têm cofator 2. As de Koblitz, têm cofator 2 para $a = 1$ e cofator 4 para $a = 0$. Os seguintes parâmetros são dados:

- Representação do corpo finito: normal ou poinomial
- Para curva de Koblitz: coeficiente a, ordem r, ponto-base $G = (x_G, y_G)$
- Curva aleatória : ordem r
  - Base polinomial: coeficiente b, ponto-base $G = (x_G, y_G)$
  - Base normal: semente s para o algoritmo SHA-1, coeficiente b, ponto-base $G = (x_G, y_G)$

Foram cobertos neste capítulo os padrões de ECC mais difundidos na atualidade, no que concerne à escolha de parâmetros de domínio, visando a resultados que tenham bom desempenho e boa segurança. No próximo capítulo serão sumarizados os aspectos discutidos nos capítulos anteriores, reforçando os passos necessários a uma seleção bem sucedida de tais parâmetros.

Tabela 7.1: Padrões usados em WTLS

Curva#	FIPS 186-2	SEC 2	X9.62
3	K-163	sect163k1	-
4	-	sect113r1	-
5	-	-	c2pnb163v1
7	-	secp160r1	-
10	K-233	sect233k1	-
11	B-233	sect233r1	-
12	P-224	secp224r1	-

Capítulo 8

Conclusões

Este capítulo resume os aspectos importantes a serem considerados numa escolha de parâmetros de domínio que tenha bom desempenho e boa segurança. Alerta, ainda, quanto a opções que devem ser evitadas para que o sistema criptográfico esteja mais protegido contra os ataques listados neste trabalho.

A seleção de parâmetros de domínio constitui-se numa etapa importante da implementação de qualquer sistema de criptografia baseado em curvas elípticas. Dela dependerão aspectos significativos de segurança e desempenho. Dependendo das necessidades específicas do sistema sendo construído, um destes aspectos pode ter prioridade sobre o outro.

Levando-se em consideração o corpo finito sobre o qual a curva elíptica será definida e a forma como tal curva é obtida, há padrões como o FIPS 186-2 que restringem significativamente as escolhas e outros como o X9.62 que, apesar de sugerir algumas opções, dá ao usuário a possibilidade de gerar arbitrariamente seus próprios parâmetros. É necessário lembrar que a segurança dos esquemas de criptografia baseados em curvas elípticas não reside em se manter secretos ou escolher aleatoriamente os parâmetros de domínio, mas no fato do problema matemático subjacente aos protocolos (o logaritmo discreto sobre curva elíptica) ser de difícil solução, apresentando complexidade exponencial.

Há escolhas que resultam em implementações em hardware muito eficientes como os corpos binários com bases gaussianas. Para plataformas de uso geral, os corpos primos NIST permitem operações de redução de módulo com desempenho diferenciado. Independente do corpo escolhido, as curvas de Koblitz contribuem para a melhora de desempenho devido à eficiência apresentada na multiplicação de pontos por escalares.

Quanto maior o grau de liberdade na escolha de parâmetros (como geração aleatória de curva elíptica via método CM e escolha aleatória de ponto-base, com devida eliminação de escolhas frágeis), maior o grau de segurança que o sistema pode apresentar, por se evitarem futuros ataques de propósito específicos contra tipos especiais de curva (como teme-se que aconteça um dia com as de Koblitz). A escolha aleatória diminui a probabilidade de se

escolher uma curva vulnerável, desde que as escolhas sabidamente inseguras (como as anômalas e as susceptíveis aos ataques do tipo MOV) sejam adequadamente filtradas. O consórcio SECG e o padrão X9.62 listam restrições que, se devidamente, seguidas, minimizam tais riscos de segurança. O excesso de liberdade, entretanto, pode comprometer a interoperabilidade com outras implementações, como visto na especificação SEC 2. Uma análise criteriosa dos requisitos do sistema em implementação é fundamental para guiar as escolhas a serem feitas para atribuir o peso adequado a desempenho, segurança e interoperabilidade.

A versão de 2005 do padrão X9.62 é bastante exigente quanto à eliminação de escolhas que possam resultar em fragilidades de segurança. Ela, por exemplo, não permite que o corpo binário escolhido tenha característica com expoente composto, que os pontos-base tenham ordem não-prima ou que os cofatores sejam arbitrariamente grandes. Ela é mais restritiva que a SEC 2, proibindo o uso níveis de segurança inferiores a 160 bits.

Também em 2005, a Agência NSA dos Estados Unidos da América publicou uma série de recomendações na chamada "Suite B", instruindo a utilizar para proteção de documentos classificados como SECRET corpos primos com módulo de 256 bits. Além disso, para proteção de documentos classificados como TOP SECRET, a recomendação da agência é que se utilizem corpos primos com módulo de 384 bits.

Para casos especiais mostrados na seção 6.2 é possível reduzir o problema do ECDLP a um outro mais simples, como o de DLP (Problema de Logaritmo Discreto) sobre um corpo finito, que possa ser resolvido em tempo sub-exponencial. É necessário eliminar do processo de seleção de parâmetros de domínio aqueles que resultem em tais fragilidades. Como constantes pesquisas são feitas na área de ECC, é necessário contínuo acompanhamento para prevenção contra novos tipos de ataques para os quais os sistemas já implementados apresentem vulnerabilidade.

Como uma opção de trabalho futuro existe a possibilidade de se analisarem as implicações de escolhas de parâmetros associados a sistemas criptográficos baseados em curvas hiper-elípticas, que são uma extensão natural de ECC, e para os quais as definições e padronizações ainda estão comparativamente em estágio inicial.

Referências Bibliográficas

[1] ANS X9.52-1998: Triple Data Encryption Algorithm, Modes of Operation. American National Standard or Financial Service, 1998.

[2] ANS X9.62-2005: Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA). American National Standard or Financial Service, 2005.

[3] ANS X9.62-2001: Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography. American National Standard or Financial Service, 2001.

[4] ANS X9.79, PKI Practices and Policy Framework. American National Standard or Financial Service, 2000.

[5] IEEE Std 1363-2000: IEEE Standard Specification for Public-Key Cryptography. IEEE Computer Society, 2000.

[6] IEEE Std 1363a-2004: IEEE Standard Specification for Public-Key Cryptography - Ammendment 1: Additional Techniques. IEEE Computer Society, 2004.

[7] FIPS 186-2: Digital Siganture Standards. Federal Information Processing Standards Publication, 2000.

[8] SEC 1. Elliptic Curve Cryptography. Standards for Efficient Cryptography Group, 2000.

[9] SEC 2. Recommended Elliptic Curve Domain Parameter Standards for Efficient Cryptography Group, 2000.

[10] Wireless Transport Layer Security, Versão 06. WAP Forum, 2001.

[11] ARAKI, K.; SATOH, T. Fermat Quotients and the Polynomial Time Discrete Log Algorithm for Anomalous Elliptic Curves. Comm. Math Unvi. Sancti. Pauli, v. 47, 1998.

[12] ATKIN, A.; MORAIN, F. Elliptic curves and primality proving. Math. Comp., V. 61, 1993.

[13] COHEN, H. A Course In Computational Algebraic Number Theory. Springer-Verlag, 1993.

[14] DEMARRAIS, J.; ADLEMAN, L. A subexponential algorithm for discrete logarithm over all finite fields. Mathematics of Computation, v. 61, 1993.

[15] ELGAMAL, T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Info. Theory, v. 31, p. 469-472, 1985.

[16] FLOYD, R. W. Non-deterministic Algorithms. J. Assoc. Computing, p. 636-644, 1967.

[17] FREY, G. Applications of arithmetical geometry to cryptographic constructions. IN: 5th International Conference on Finite Fields and Their Applications. Proceedings of the 5th International Conference on Finite Fields and Their Applications, Springer, 2001.

[18] FREY, G. Weil Descent. Waterloo University, 1998.

[19] FREY, G.; RÜCK, H. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. Mathematics of Computation, v. 62, 1994.

[20] GALLANT, R. Faster elliptic curve cryptography using efficient endomorphisms. ECC99, Waterloo University, 1999.

[21] GALLANT, R.; LAMBERT, R.; VANSTONE S. Improving the Parallelized Pollard Lambda Search on Binary Anomalous Curves. Mathematics of Computation, v. 69, 2000.

[22] GALLANT, R.; LAMBERT, R.; VANSTONE, S. Faster point multiplication on elliptic curves with efficient endomorphisms. Advances in CryptologyCrypto 2001, Lecture Notes in Computer Science, v. 2139, Springer-Verlag, 2001.

[23] GAUDRY, P. An algorithm for solving the discrete logarithm problem on hyperelliptic curves. EUROCRYPT 2000, Springer-Verlag LNCS 1807, 2000.

[24] GORDON, D. M. Discrete logarithms in GF(p) using the number field sieve. Siam Journal on Discrete Mathematics, v. 6, 1993.

[25] HAFNER, J. L.; MCURLEY, K. S. A rigorous subexponential algorithm for computation of class groups. Journal of the American Mathematical Society, v. 2, 1989.

[26] HELLMAN, M. E.; DIFFIE, W. New directions in cryptography. IEEE Trans. Information Theory, v. 22, p. 644-654, 1976.

[27] HELLMAN, M.; POHLING, S. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. IEEE Transactions on Information Theory, v. 24, 1978.

[28] HESS, F.; GAUDRY, P.; SMART, N. Constructive and destructive facets of Weil descent on elliptic curves. Journal of Cryptology, v. 15, 2002.

[29] KOBLITZ, N. Elliptic Curve Cryptosystems. Mathematics of Computation, v. 48, 1987.

[30] KOBLITZ, Neal. Algebraic Aspects of Cryptography. Springer, 1999.

[31] KOBLITZ, Neal. Hyperelliptic Cryptosystems. J. Cryptology, v. 1, 1989.

[32] KOBLITZ, Neal. CM-curves with good cryptographic properties. Advances in Cryptology: Crypto 91, v. 576, Springer-Verlag, 1992.

[33] KUHN, F.; STRUIK, R. Random walks revisited: Extensions of Pollards rho algorithm for computing multiple discrete logarithms. Lecture Notes in Computer Science, v. 2259, 2001.

[34] LEE, Byung Kwan; YANG, Seung; LEE, Tai-Chi. HESSL (Highly Enhanced Security Socket Layer) Protocol. E-Commerce Technology, 2005.

[35] MENEZES, A. Evaluation of Security Level of Cryptography: The Elliptic Curve Discrete Logarithm Problem (ECDLP). Waterloo University, 2001.

[36] MENEZES, A. J.; VANSTONE, S. A.; OKAMOTO, T. Reducing elliptic curve logarithms to a finite field. IEEE Trans. Info. Theory, v. 39, 1993.

[37] MENEZES, A.; HANKERSON, D; HERNANDEZ, J. L. Software Implementation of Elliptic Curve Cryptography Over Binary Fields. Lecture Notes in Computer Science, v. 1965, 2000.

[38] MENEZES, A.; KOBLITZ, N.; VANSTONE, S. The state of elliptic curvecryptography. Designs, Codes and Cryptography, v. 19, 2000.

[39] MENEZES, A.; QU, M. Analysis of the Weil descent attack of Gaudry, Hess and Smart. Topics in Cryptology CT-RSA 2001, Lecture Notes in Computer Science, v. 2020, Springer, 2001.

[40] MENEZES, A.; STEIN, A.; JACOBSON, M. Solving Elliptic Curve Discrete Logarithm Problems Using Weil Descent. Combinatorics and Optimization Research Report, v. 31, Waterloo University, May 2001.

[41] MENEZES, A.; TESKE, E.; MAURER, M. Analysis of the GHS Weil descent attack on the ECDLP over characteristic two finite fields of composite degree. LMS Journal of Computation and Mathematics, v. 5, 2002.

[42] MENEZES, A.; VAN OORSCHOT, P. Handbook of Applied Cryptography. CRC Press, 1996.

[43] MENEZES, Alfred ; HANKERSON, Darrel; VANSTONE, Scott. Guide to Elliptic Curve Cryptography Springer, 2004.

[44] MENEZES, Alfred; WENG, Annegret; TESKE, Edlyn. Weak Fields for ECC. IACR, 2003.

[45] MILLER, V. Uses of Elliptic Curves in Cryptography. IN: Crypto '85. Advances in Cryptology-Crypto '85, LNCS 218, 1986.

[46] MONTGOMERY, P. Speeding the Pollard and elliptic curve methods of factorization. Mathematics of Computation, v. 48, 1987.

[47] MORAIN, F.; LERCIER, R. Counting the number of points on elliptic curves over finite fields: strategies and performances. Eurocript 96, Springer-Verlag, 1996.

[48] NING, P; YIN, Y. Efficient software implementation for finite field multiplication in normal basis. Informatioon and Communications Security 2001 (LCNS 2229), p. 177-189, 2001.

[49] ODLYZKO, A. Discrete logarithms in finite fields and their cryptographic significance. IN: Eurocrypt '84. Advances in Cryptology - Proceedings of Eurocrypt '84, Lecture Notes in Computer Science, Springer-Verlag, 1985.

[50] ODLYZKO, A.; COOPERSMITH, D.; SCHROEPPEL, R. Discrete Logarithms in GF(p). Algorithmica, v. 1, 1986.

[51] OMURA, J. Omura; MASSEY, J. Computational method and apparatus for finite field arithmetic. U.S. Patent number 4,587,627, 1986.

[52] ONYSZCHUCK, I. et al. Optimal normal bases in GF(p^n). Discrete Applied Mathematics, v. 22, 1988.

[53] POLLARD, J.M. Monte Carlo methods for index computation (mod p). Math. Comp., v. 32, 1978.

[54] SCHEIDLER, R. Cryptography in quadratic function fields. Designs, Codes and Cryptography, v. 22, 2001.

[55] SEMAEV, I. A. Evaluation of Discrete Logarithms on some Elliptic Curves. Mathematics of Computation, v. 67, p. 353-356, 1998.

[56] SEROUSSI, Gadiel; BLAKE, Ian; SMART, Nigel. Elliptic Curves in Cryptography. Cambridge Univerity Press, 2002.

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

[58] SILVERMAN, J. H. The Arithmetic of Elliptic Curves. Springer-Verlag, 1986.

[59] SMART, N. The Discrete Logarithm Problem on Elliptic Curves of Trace One. Journal of Cryptology, v. 12 , 1999.

[60] STEIN, A.; WILLIAMS, H. C.; SCHEIDLER, R. Key exchange in real quadratic congruence function fields. Designs, Codes and Cryptography, v. 7, 1996.

[61] STEIN, Andreas; JACOBSON, Michael. Faster Cryptographic Key Exchange on Hyperelliptic Curves. Pre-print, Jun. 2005.

[62] TESKE, Edlyn. An ellyptic curve trapdoor system. Cryptology ePrint Archive Report, v. 058, 2003.

[63] WIENER, M.; VAN OORSCHLOT, P. Parallel collision search with cryptanalytic applications. Journal of Cryptology, v. 12, 1999.

[64] WIENER, M.; ZUCCHERATO, R. Fast Attacks on Elliptic Curve Cryptosystems. IN: Fifth Annual Workshop on Selected Areas in Cryptography SAC 98. Lecture Notes in Computer Science, Springer-Verlag, 1998.