INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS

DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIAS

CURSO DE ENGENHARIA INFORMÁTICA

EMANUEL CARNEIRO DOS SANTOS

CURVAS ELÍPTICAS E CRIPTOGRAFIA

LUANDA

27/06/2025

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS

DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIAS

CURSO DE ENGENHARIA INFORMÁTICA

EMANUEL CARNEIRO DOS SANTOS

CURVAS ELÍPTICAS E CRIPTOGRAFIA

Relatório apresentado à disciplina de Matemática Discreta, da turma EIN4_T3 do curso de Engenharia Informática do ISPTEC, como requisito parcial para avaliação contínua.

DOCENTE: VALTER PAULO TOMÉ

LUANDA

27/06/2025

RESUMO

Este trabalho explora a teoria e aplicação da criptografia de curvas elípticas (ECC). Inicia-se com uma fundamentação teórica das curvas elípticas, abordando sua definição, a equação de Weierstrass, a aritmética sobre corpos finitos e a lei de grupo que rege a adição de pontos. Em seguida, discute-se o Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP), que constitui a base de segurança da ECC. O estudo aprofunda-se nos principais protocolos criptográficos baseados em ECC, como o Elliptic Curve Diffie-Hellman (ECDH) para troca de chaves e o Elliptic Curve Digital Signature Algorithm (ECDSA) para assinaturas digitais, comparando suas características de segurança e eficiência com o algoritmo RSA. Adicionalmente, apresenta-se uma implementação prática em Python das operações fundamentais de curvas elípticas e uma simulação do protocolo ECDH, demonstrando sua funcionalidade. Por fim, são analisadas a complexidade computacional da ECC, suas vulnerabilidades conhecidas e um comparativo de curvas padrão, culminando numa reflexão sobre a importância das curvas elípticas para a segurança da informação moderna, especialmente em ambientes com recursos limitados..

Palavras-chave: Criptografia de Curva Elíptica (ECC), Curvas Elípticas, Equação de Weierstrass, Corpos Finitos, ECDH, ECDSA, Criptografia.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS, SIGLAS E ACRÓNIMOS

Símbolo	Descrição	Unidade
a, b	Coeficientes da equação de Weierstrass da curva elíptica.	N/A
ECC	Criptografia de Curva Elíptica (Elliptic Curve Cryptography).	N/A
ECDH	Protocolo Diffie-Hellman de Curva Elíptica (Elliptic Curve Diffie-Hellman).	N/A
ECDSA	Algoritmo de Assinatura Digital de Curva Elíptica (Elliptic Curve Digital Signature Algorithm).	N/A
F_p	Corpo finito com p elementos (onde p é um número primo).	N/A
F_{2^n}	Corpo finito binário com 2^n elementos.	N/A
G	Ponto base (gerador) da curva elíptica.	N/A
IoT	Internet das Coisas (Internet of Things).	N/A
k, n	Escalares (inteiros); geralmente utilizados como chaves privadas na multiplicação escalar.	N/A
O ou ∞	Ponto no Infinito; elemento neutro do grupo da curva elíptica.	N/A
$O(\log n)$	Notação Big-O para complexidade computacional logarítmica.	N/A
P, Q, R	Pontos genéricos na curva elíptica.	N/A
P	Número primo; característica do corpo finito sobre o qual a curva é definida.	N/A
R	Conjunto dos números reais.	N/A
RSA	Algoritmo RSA (Rivest-Shamir-Adleman), um criptossistema de chave pública.	N/A
SSL	Secure Sockets Layer (Camada de Soquetes Seguros).	N/A
TLS	Transport Layer Security (Segurança da Camada de Transporte).	N/A

Tabela 1 - Lista de Símbolos, Descrições e Unidades Utilizados no Documento

SUMÁRIO

1 INTRODUÇÃO

No cerne da segurança digital moderna encontra-se um desafio constante: a necessidade de criar sistemas de encriptação cada vez mais fortes para proteger dados sensíveis, ao mesmo tempo que estes sistemas devem ser eficientes o suficiente para funcionar em dispositivos com poder computacional limitado, como smartphones, cartões de crédito e sensores da Internet das Coisas (IoT). Criptossistemas estabelecidos, como o RSA, baseiam a sua segurança em chaves de grande dimensão (e.g., 2048 ou 3072 bits), o que pode ser computacionalmente dispendioso. É neste contexto que a Criptografia de Curva Elíptica (ECC) surge como uma alternativa elegante e poderosa. Proposta por Neal Koblitz e Victor S. Miller de forma independente em 1985, a ECC oferece níveis de segurança comparáveis aos do RSA, mas com chaves drasticamente menores , resultando em cálculos mais rápidos, menor consumo de energia e menor necessidade de largura de banda. A "magia" por trás desta eficiência não está em novos algoritmos criptográficos, mas sim na aplicação de algoritmos já existentes a uma estrutura matemática rica e complexa: o grupo de pontos de uma curva elíptica.

1.1 Definição de Curva Elíptica

Uma curva elíptica, apesar do nome, não é uma elipse. É uma curva algébrica, definida como o conjunto de pontos que satisfazem uma equação cúbica (de grau 3) não-singular. A forma geral de uma equação cúbica em duas variáveis, sobre um corpo K, é dada por:

$$ax3 + bx2y + cxy2 + dy3 + ex2 + fxy + gy2 + hx + iy + j = 0 \qquad (1)$$

Felizmente, para propósitos criptográficos, qualquer curva cúbica com um ponto racional pode ser transformada, através de mudanças de variáveis, para uma forma muito mais simples e prática, conhecida como **Forma de Weierstrass**.

1.2 Equação de Weierstrass e Singularidade

Se o corpo K sobre o qual a curva está definida tem característica diferente de 2 e 3 (o que é o caso para os corpos primos F_p usados em muitas aplicações), a equação da curva elíptica pode ser simplificada para a forma curta de Weierstrass:

$$y^2 = x^3 + ax + b \qquad \text{(Equação de Weierstrass)} \qquad (2)$$

onde *a* e *b* são constantes do corpo K.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

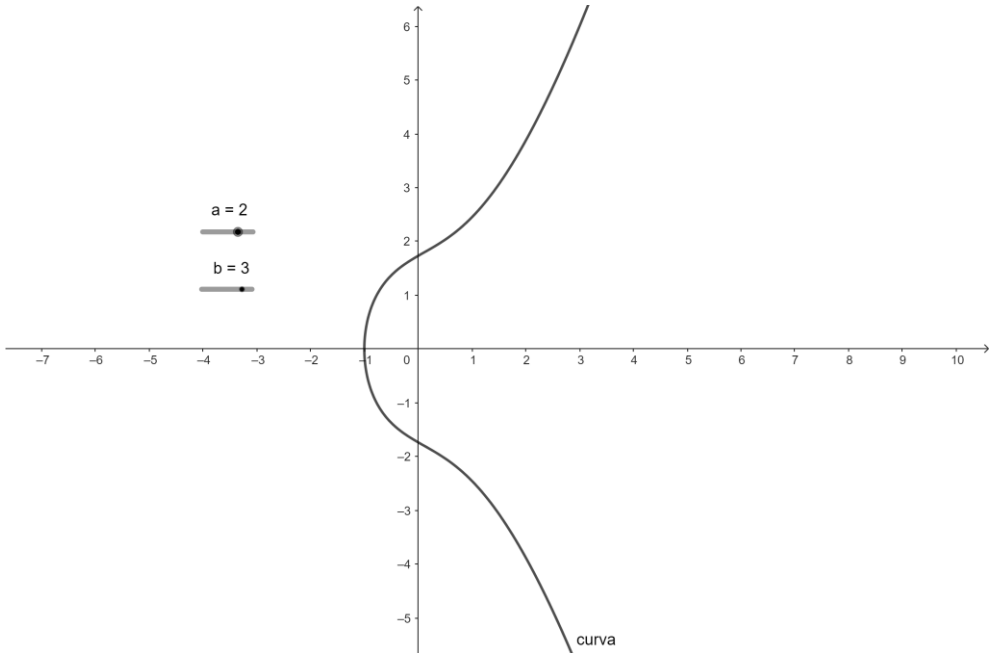

Figura 1 - Representação geométrica de uma curva elíptica suave em R

Para que esta equação defina uma curva elíptica válida para a criptografia como a curva acima $y^2 = x^3 + 2x + 3$, ela deve ser **não-singular**. Isto significa que a curva não pode ter "*cúspides*" (pontos aguçados) ou autointerseções. Geometricamente, a não-singularidade garante que em cada ponto da curva existe uma única reta tangente bem definida, uma propriedade essencial para a lei de grupo que exploraremos a seguir. Esta condição é satisfeita se o discriminante do polinómio cúbico for diferente de zero:

$$\Delta = -16(4a^3 + 27b^2) \neq 0 \quad \Rightarrow \quad 4a^3 + 27b^2 \neq 0 \qquad (3)$$

Se $\Delta = 0$, o polinómio $\boldsymbol{x^3 + ax + b}$ teria raízes múltiplas, o que levaria a pontos singulares na curva.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

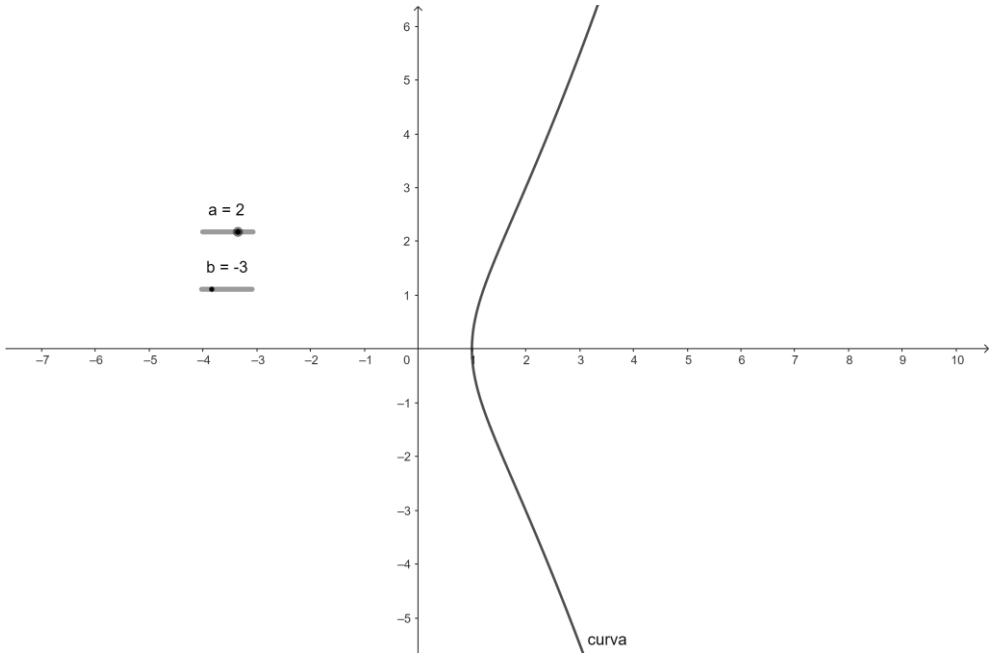

Figura 2 - Exemplos de curvas cúbicas com singularidades

Para corpos de característica 2 ou 3, utiliza-se uma forma mais geral da equação de Weierstrass:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad (4)$$

1.3 Curvas Elípticas sobre Corpos Finitos (F_P)

Enquanto as curvas sobre corpos como os números reais (R) são úteis para a visualização intuitiva, a criptografia utiliza curvas definidas sobre **corpos finitos**. Um corpo finito, também chamado de Corpo de Galois , é um conjunto finito de elementos onde as operações de adição, subtração, multiplicação e divisão (exceto por zero) são bem definidas.

O corpo finito mais simples para a criptografia de curvas elípticas é o **corpo primo** Fp, que consiste no conjunto de inteiros {0, 1, 2, … ,p−1}, onde p é um número primo. Todas as operações aritméticas são realizadas **módulo p**.

Uma curva elíptica sobre F_p é, portanto, o conjunto de todos os pontos (x, y), com $x, y \in F_p$ que satisfazem a equação de Weierstrass módulo p:

$$y^2 \equiv x^3 + ax + b \ (mod\ p) \quad (5)$$

Ao contrário da sua análoga sobre os números reais, que é uma curva contínua, uma curva elíptica sobre um corpo finito é um conjunto discreto de pontos. Visualmente, assemelha-se mais a um gráfico de dispersão do que a uma curva suave. No entanto,

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

o mais importante é que as propriedades algébricas que permitem a sua rica estrutura de grupo são perfeitamente preservadas.

1.4 A Lei de Grupo e a Adição de Pontos

A propriedade mais fundamental das curvas elípticas, que as torna tão valiosas para a criptografia, é que os seus pontos formam um **grupo abeliano** sob uma operação de "adição". Esta "adição" não é a simples soma de coordenadas, mas sim uma operação geométrica definida pela "regra da corda e da tangente".

1.5 O Ponto no Infinito (*O*)

Para que a estrutura de grupo seja completa, é necessário incluir um elemento especial: o **ponto no infinito**, denotado por **O**. Este ponto não tem coordenadas (x,y) finitas; ele pode ser visualizado como um ponto situado no topo e na base do eixo y simultaneamente. Geometricamente, todas as retas verticais se intersetam neste ponto. Algebricamente,

"O" funciona como o **elemento neutro (identidade)** do grupo.

1.6 Interpretação Geométrica da Adição de Pontos

A operação de adição de pontos, denotada por "+", é definida da seguinte forma:

1. **Identidade**: Para qualquer ponto P na curva, $P + O = P$.
2. **Inverso de um Ponto**: Para um ponto $P = (x, y)$, o seu inverso aditivo, −P, é a sua reflexão sobre o eixo x, ou seja, $-P = (x, -y)$. A reta que une P e −P é vertical, e o seu "terceiro" ponto de interseção com a curva é O. Assim, $P + (-P) = O$.
3. **Adição de Dois Pontos Distintos ($\boldsymbol{P + Q}$)**: Para somar dois pontos distintos P e Q, traça-se uma reta que passa por ambos. Pelo *Teorema de Bézout*, uma reta interseta uma cúbica em exatamente três pontos (contando com multiplicidades). A reta que passa por P e Q intersetará a curva num terceiro ponto, que chamamos de $P \cdot Q$. A soma $P + Q$ é então definida como a reflexão deste ponto sobre o eixo x, ou seja, $P + Q = -(P \cdot Q)$.
4. **Duplicação de um Ponto ($\boldsymbol{P + P}$ ou $\boldsymbol{2P}$)**: Para somar um ponto P a si mesmo, a "reta secante" torna-se a **reta tangente** à curva no ponto P. O processo é o mesmo: encontra-se o outro ponto de interseção da tangente com a curva, $P \cdot P$, e a sua reflexão é $2P$.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

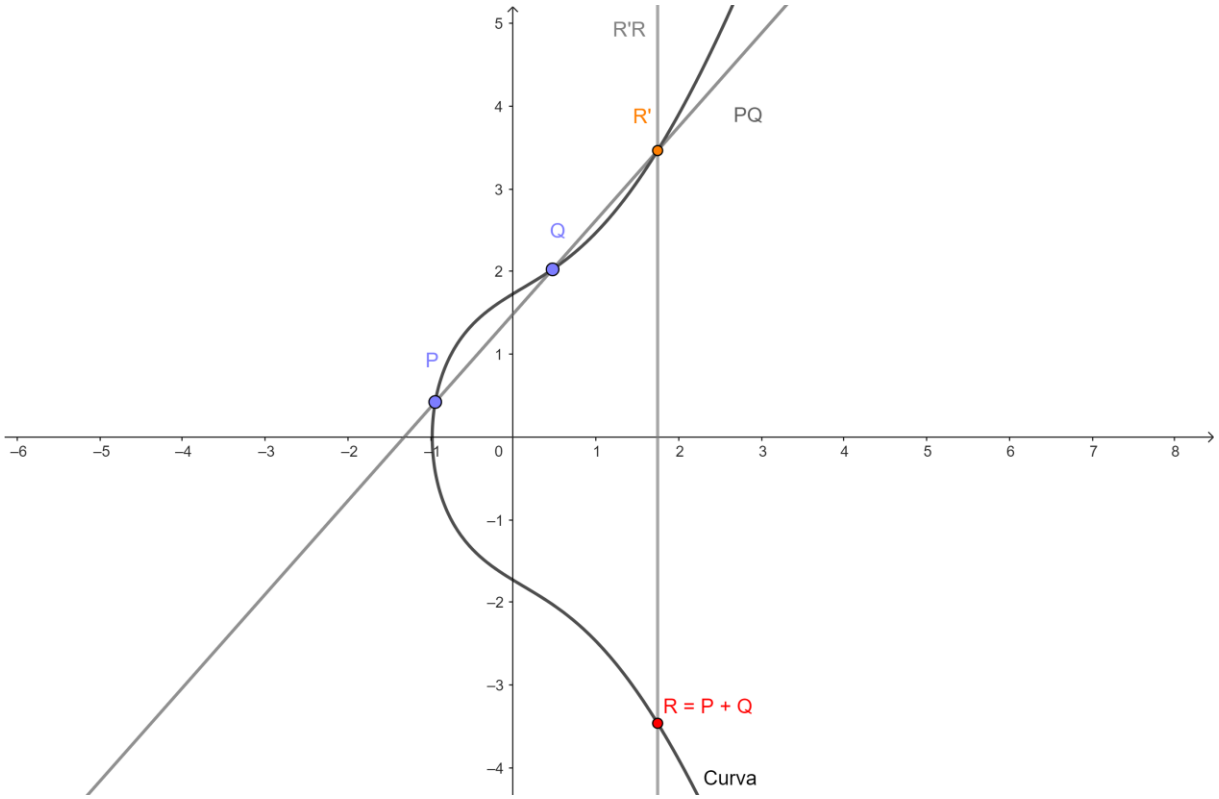

Figura 3 - Representação geométrica da operação $P + Q = R$ com reflexão sobre o eixo x, conforme a regra da tangente/secação em curvas elípticas.

1.7 Propriedades Algébricas do Grupo

Este conjunto de pontos, com a operação de adição e o ponto no infinito, forma um **grupo abeliano**, o que significa que satisfaz as seguintes propriedades:

- **Fechamento**: Se P e Q são pontos na curva, $P + Q$ também é um ponto na curva. Se P e Q têm coordenadas racionais, $P + Q$ também terá coordenadas racionais.
- **Associatividade**: $(P + Q) + R = P + (Q + R)$ para quaisquer pontos P, Q, R. Esta é a propriedade menos intuitiva, mas a sua prova (que envolve o Teorema de Cayley-Bacharach) garante que a operação é bem definida.
- **Elemento Neutro**: Existe um elemento identidade, O, tal que $P + O = P$.
- **Elemento Inverso**: Para cada ponto P, existe um inverso $-P$ tal que $P + (-P) = O$.
- **Comutatividade**: $P + Q = Q + P$. Isto é evidente, pois a reta que passa por P e Q é a mesma que passa por Q e P.

Esta estrutura de grupo é a fundação sobre a qual a segurança da Criptografia de Curva Elíptica é construída. O passo seguinte é entender qual problema matemático nesta estrutura é tão "difícil" de resolver que nos permite criar sistemas de encriptação seguros.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

2 FUNDAMENTOS MATEMÁTICOS

Após estabelecermos a definição e a estrutura de grupo das curvas elípticas, é fundamental aprofundar os conceitos matemáticos que sustentam a sua aplicação criptográfica. A segurança da Criptografia de Curva Elíptica (ECC) não reside na complexidade da curva em si, mas na dificuldade computacional de um problema específico dentro do seu grupo de pontos.

2.1 Problema do Logaritmo Discreto em Curvas Elíplicas (ECDLP)

A segurança de todos os criptossistemas baseados em curvas elípticas assenta na dificuldade computacional do **Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP)**.

Seja E uma curva elíptica sobre um corpo finito Fq. Sejam P um ponto na curva (conhecido como **ponto base**) e Q um outro ponto. O ECDLP consiste em, dado P e Q, encontrar um inteiro k tal que:

$$Q = kP \tag{2-1}$$

onde **kP** representa a operação de somar o ponto P a si mesmo k vezes ($P + P + \cdots + P$). Este inteiro k é o **logaritmo discreto** de Q na base P.

A essência do problema reside numa assimetria computacional:

- **Operação Direta (Multiplicação Escalar)**: Dado um inteiro k e um ponto P, calcular o ponto $Q = kP$ é uma tarefa computacionalmente eficiente. Pode ser realizada rapidamente através de algoritmos como o "double-and-add", que opera em tempo logarítmico em relação a k.
- **Operação Inversa (Logaritmo Discreto)**: Dado os pontos P e Q (onde $Q = kP$), encontrar o inteiro k é considerado um problema computacionalmente intratável para curvas bem escolhidas.

Até hoje, não são conhecidos algoritmos sub-exponenciais para resolver o ECDLP em curvas elípticas gerais. Acredita-se que este problema seja significativamente mais "difícil" de resolver do que os seus análogos, como o problema da fatoração de inteiros (no qual o RSA se baseia) ou o problema do logaritmo discreto em corpos finitos (usado no ElGamal tradicional). É esta dificuldade que permite à ECC oferecer níveis de segurança equivalentes com chaves muito menores.

2.2 Comparação entre Curvas sobre R e F_p

É importante distinguir claramente as curvas elípticas definidas sobre o corpo dos números reais (R) daquelas definidas sobre corpos finitos (F_p), pois elas servem a propósitos muito diferentes.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

- **Curvas sobre R (Números Reais)**:
    - **Natureza**: São curvas contínuas e suaves, com um número infinito de pontos.
    - **Visualização**: Correspondem aos gráficos familiares que podem ser desenhados no plano cartesiano, como os mostrados na Figura 1 da secção anterior.
    - **Utilidade**: A sua principal função é **pedagógica e intuitiva**. Elas permitem visualizar a lei de grupo geométrica da "corda e da tangente" de uma forma clara, facilitando a compreensão do conceito de adição de pontos.
- **Curvas sobre F_p (Corpos Finitos)**:
    - **Natureza**: São um **conjunto finito e discreto** de pontos. As coordenadas (x,y) de cada ponto são inteiros pertencentes a {0, 1, ...,p−1}.
    - **Visualização**: Se fossem desenhados, não se assemelhariam a uma curva, mas sim a um gráfico de dispersão com um número finito de pontos.
    - **Utilidade**: São a base da criptografia de curvas elípticas. O facto de o conjunto de pontos

E(F_p) formar um **grupo abeliano finito** é a propriedade central explorada pelos protocolos criptográficos.

A ponte entre os dois mundos é a **álgebra**. Embora a representação visual seja drasticamente diferente, as fórmulas algébricas para a adição de pontos e a multiplicação escalar são as mesmas, com a diferença crucial de que, em

Fp, todas as operações são realizadas sob a aritmética modular. As propriedades de grupo (fechamento, associatividade, identidade, inverso, comutatividade) são preservadas, o que permite que a intuição geométrica do mundo real seja transportada para a estrutura algébrica do mundo finito.

2.3 Curvas Elípticas sobre corpos binários (F_{2n})

Além dos corpos primos Fp, outra classe de corpos finitos de grande importância prática na ECC são os **corpos binários**, denotados por F_{2^n} (ou $GF(2^n)$).

- **Definição**: Um corpo F_{2^n} é uma extensão de grau n do corpo $F_2 = \{0, 1\}$. Os seus elementos são tipicamente representados como polinómios de grau menor que n com coeficientes em F_2.
- **Equação da Curva**: Como a característica do corpo é 2, a forma curta de Weierstrass não pode ser usada, pois a sua derivação envolve dividir por 2.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

Nesses casos, utiliza-se a forma geral da equação de Weierstrass, que se simplifica para uma de duas formas:

1) $y^2 + xy = x^3 + ax^2 + b$
2) $y^2 + cy = x^3 + ax + b$

- **Vantagens**: A aritmética em corpos binários pode ser implementada de forma muito eficiente em hardware digital, uma vez que as operações se traduzem bem em circuitos lógicos (por exemplo, a adição é um XOR bit-a-bit). Por esta razão, curvas sobre F_{2^n} são especificadas em vários padrões de criptografia, como os do IETF (RFC 2409 e RFC 2412).

Apesar da aritmética e da forma da equação serem diferentes, os princípios criptográficos fundamentais permanecem os mesmos: os pontos na curva formam um grupo abeliano e a segurança do sistema baseia-se na dificuldade do ECDLP.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

3 CRIPTOGRAFIA DE CURVA ELÍPTICA (ECC)

Com os alicerces teóricos e matemáticos estabelecidos, podemos agora explorar as aplicações práticas das curvas elípticas no domínio da criptografia. Os protocolos que utilizam curvas elípticas não são fundamentalmente novos; em vez disso, são adaptações de criptossistemas de chave pública já existentes, como o de Diffie-Hellman e o DSA (Digital Signature Algorithm), que substituem as operações de exponenciação modular por operações no grupo de pontos de uma curva elíptica. A grande vantagem desta abordagem, como veremos, reside na obtenção de um nível de segurança equivalente com chaves consideravelmente menores e operações mais eficientes.

3.1 Protocolo Diffie-Hellman de Curva Elíplica (ECDH)

O protocolo **Elliptic Curve Diffie-Hellman (ECDH)** é uma variante do protocolo de troca de chaves Diffie-Hellman. A sua finalidade é permitir que duas partes (Alice e Bob) estabeleçam um segredo partilhado através de um canal de comunicação inseguro, sem que nunca tenham partilhado qualquer informação secreta previamente. Este segredo pode então ser usado para derivar uma chave simétrica para cifrar a comunicação subsequente.

O protocolo funciona da seguinte forma:

1. **Acordo de Parâmetros de Domínio**: Primeiramente, Alice e Bob devem concordar publicamente sobre um conjunto de **parâmetros de domínio** da curva elíptica. Estes parâmetros, que podem ser denotados como D=(*p,a,b,G,n,h*), definem o ambiente criptográfico e incluem:
    - O primo *p* que define o corpo finito Fp.
    - Os coeficientes *a* e *b* da equação de Weierstrass da curva.
    - Um ponto base *G* (ou ponto gerador) na curva.
    - A ordem *n* do subgrupo gerado por *G*.
    - O cofator *h*, que é a razão entre a ordem da curva e a ordem *n*.
2. **Geração de Chaves**: Cada parte gera o seu próprio par de chaves, uma privada e uma pública:
    - Alice escolhe um inteiro aleatório d_A (a sua **chave privada**) no intervalo *[1, n−1]*. Ela então calcula o seu ponto correspondente, a **chave pública** $Q_A = d_A \cdot G$.
    - Bob faz o mesmo, escolhendo a sua chave privada dB e calculando a sua chave pública $Q_B = d_B \cdot G$.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

3. **Troca de Chaves Públicas**: Alice envia a sua chave pública Q_A para Bob, e Bob envia a sua chave pública Q_B para Alice. O canal pode ser inseguro, o que significa que um atacante (Oscar) pode intercetar Q_A e Q_B.
4. **Cálculo do Segredo Partilhado**:
    - Alice, ao receber Q_B, calcula o ponto secreto $S = d_A \cdot Q_B = d_A \cdot (d_B \cdot G)$.
    - Bob, ao receber Q_A, calcula o ponto secreto $S = d_B \cdot Q_A = d_B \cdot (d_A \cdot G)$.

Devido à propriedade associativa da multiplicação escalar, ambos chegam ao mesmo ponto $S = (d_A \cdot d_B) \cdot G$. O segredo partilhado é tipicamente a coordenada x deste ponto, S_x. Um atacante que intercete Q_A, e Q_B não consegue calcular S, pois para isso precisaria de d_A ou d_B, o que exigiria resolver o Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP).

3.2 Esquema de Assinatura Digital de Curva Elíptica (ECDSA)

O **Elliptic Curve Digital Signature Algorithm (ECDSA)** é uma variante do Digital Signature Algorithm (DSA) que utiliza a matemática das curvas elípticas. A sua função é permitir a criação de **assinaturas digitais**, que providenciam autenticidade da origem e garantia de integridade da mensagem. O protocolo é dividido em três fases: geração de chaves, geração da assinatura e verificação da assinatura.

1. Geração de Chaves Para que Alice possa assinar uma mensagem, ela primeiro precisa de um par de chaves:

- **Chave Privada (d_A)**: É um inteiro selecionado de forma aleatória no intervalo *[1, n−1]*.
- **Chave Pública (Q_A)**: É um ponto na curva calculado como $Q_A = d_A \cdot G$.

A chave privada d_A deve ser mantida em segredo absoluto, enquanto a chave pública Q_A pode ser distribuída livremente.

2. Geração da Assinatura Para assinar uma mensagem m, Alice executa os seguintes passos:

1. Calcula um "resumo criptográfico" (hash) da mensagem: $e = HASH(m)$. A função *HASH* (*e.g., SHA-256*) converte a mensagem de qualquer tamanho numa string de bits de tamanho fixo.
2. Seleciona um inteiro aleatório e efémero *k* no intervalo *[1, n−1]*.
3. Calcula o ponto na curva $(x_1, y_1) = k \cdot G$.
4. Calcula $r = x_1 (mod\ n)$. Se $r = 0$, deve voltar ao passo 2 e escolher um novo k.
5. Calcula $s = k^{-1}(e + r \cdot d_A)(mod\ n)$. Se $s = 0$, deve voltar ao passo 2.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

A **assinatura digital** é o par de inteiros (r,s).

3. Verificação da Assinatura Para que Bob verifique que a mensagem *m* foi realmente assinada por Alice e não foi alterada, ele precisa da chave pública de Alice, Q_A. O processo de verificação é o seguinte:

1. Bob verifica se *r* e *s* são inteiros no intervalo *[1, n−1]*.
2. Ele calcula o mesmo hash da mensagem que Alice calculou: $e = HASH(m)$.
3. Calcula $w = s^{-1}(mod\ n)$.
4. Calcula os valores $u_1 = e \cdot w(mod\ n)$ e $u_2 = r \cdot w(mod\ n)$.
5. Calcula o ponto na curva $P = (x_0, y_0) = u_1 \cdot G + u_2 \cdot Q_A$.
6. A assinatura é considerada **válida** se $x_0 \cdot (mod\ n) = r$.

Se a verificação for bem-sucedida, Bob tem a certeza de que a mensagem foi assinada com a chave privada correspondente à chave pública QA, e que a mensagem não foi alterada desde que foi assinada.

3.3 Comparação entre ECC vs. RSA

A principal e mais citada vantagem da ECC sobre o RSA é a capacidade de fornecer um nível de segurança equivalente com chaves significativamente menores. Isto deve-se ao facto de que o Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP) é considerado computacionalmente mais difícil do que o problema da fatoração de inteiros, no qual o RSA se baseia.

Característica	**RSA**	**ECC**
Princípio de Segurança	Dificuldade em fatorar o produto de dois números primos grandes ($n = p \cdot q$).	Dificuldade do Problema do Logaritmo Discreto em Curva Elíptica (ECDLP).
Tamanho da Chave	Muito maior para o mesmo nível de segurança. Uma chave de 2048-bits é comum.	Significativamente menor. Uma chave de 256-bits oferece segurança comparável a uma chave RSA de 3072-bits.
Eficiência	Mais lento, consome mais poder de processamento, largura de banda e armazenamento.	Mais rápido e eficiente, ideal para dispositivos com recursos limitados (smartphones, IoT).
Complexidade	Matematicamente mais simples de entender (aritmética modular).	Matematicamente mais complexo (geometria algébrica, corpos finitos).

Tabela 2 - Comparação entre Criptografia de Curva Elíptica (ECC) e RSA

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

A tabela seguinte, baseada em recomendações da Agência da União Europeia para a Cibersegurança (ENISA), ilustra a comparação dos tamanhos de chave (em bits) necessários para atingir níveis de segurança equivalentes em diferentes tipos de algoritmos.

Nível de Segurança (bits)	Criptografia Simétrica (e.g., AES)	ECC (tamanho do primo p)	RSA (tamanho do módulo)
80	80	160	1024
112	112	224	2048
128	128	256	3072
192	192	384	7680
256	256	512	15360

Tabela 3 - Comparação de Tamanhos de Chave (em bits) para Níveis de Segurança Equivalentes em Diferentes Algoritmos Criptográficos, conforme Recomendações da ENISA.

As implicações práticas desta diferença são imensas:

- **Eficiência Computacional**: Chaves menores resultam em cálculos mais rápidos. Operações como geração de chaves, assinatura e verificação consomem menos ciclos de CPU, o que é vital para servidores que precisam de processar um grande volume de transações seguras.
- **Economia de Recursos**: Chaves e assinaturas menores requerem menos espaço de armazenamento e menos largura de banda para serem transmitidas pela rede.
- **Adequação a Dispositivos Restritos**: A eficiência da ECC torna-a a escolha ideal para ambientes com poder de processamento, memória e bateria limitados, como smart cards, telemóveis e dispositivos IoT.

Em suma, embora o RSA continue a ser um sistema robusto e amplamente utilizado, a ECC oferece uma alternativa mais moderna e eficiente, que responde melhor às exigências dos ecossistemas digitais contemporâneos.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

4 IMPLEMENTAÇÃO EM PYTHON

A teoria por trás da Criptografia de Curva Elíptica (ECC) é fascinante, mas é na implementação prática que sua elegância e eficiência se revelam. Nesta seção, são apresentados os principais algoritmos implementados em Python, com o objetivo de demonstrar na prática os fundamentos explorados teoricamente. A abordagem cobre desde operações elementares sobre curvas elípticas como a adição de pontos e a multiplicação escalar até a aplicação desses conceitos na construção de protocolos criptográficos de chave pública, incluindo o **ECDH** (Elliptic Curve Diffie-Hellman) para troca segura de chaves e o **ECDSA** (Elliptic Curve Digital Signature Algorithm) para geração e verificação de assinaturas digitais.

4.1 Ferramentas e Abordagem

A implementação foi desenvolvida em linguagem Python. A lógica central foi construída manualmente, sem recorrer a bibliotecas criptográficas de alto nível, de forma a tornar explícitos os algoritmos subjacentes à criptografia de curva elíptica. Para garantir corretude algébrica, utilizou-se a função *mod_inverse* da biblioteca **sympy** no cálculo do inverso modular, operação essencial na definição da lei de grupo. A geração de chaves e parâmetros efêmeros foi baseada na função *randint* do módulo ***random***, enquanto o hash das mensagens, parte crucial do ECDSA, foi obtido com o algoritmo *SHA-256* da biblioteca **hashlib**.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

4.2 Implementação do Protocolo ECDH

O código apresentado a seguir encapsula a definição de uma curva elíptica e as operações essenciais para a criptografia.

Classe EllipticCurve

```python
class EllipticCurve:
    def __init__(self, a, b, p):
        self.a = a
        self.b = b
        self.p = p

    # Verifica se um ponto P(x, y) está na curva.
    def is_on_curve(self, P):
        return (y ** 2 - x ** 3 - self.a * x - self.b) % self.p == 0

    # Implementa a adição de dois pontos P e Q na curva elíptica.
    # - Pontos no infinito, pontos distintos e pontos duplicados
    def add(self, P, Q):

    # Implementa a multiplicação escalar de um ponto P por um inteiro n
    # - usando o algoritmo 'double-and-add
    def multiply(self, P, n):
```

Figura 4 - Estrutura da Classe EllipticCurve com Operações Fundamentais em Corpos Finitos

Classe EllipticCurve: Esta classe encapsula a definição da curva elíptica pelos seus coeficientes a, b e o primo p do corpo finito F_p.

- **is_on_curve(self, P):** Uma função auxiliar importante que verifica se um dado ponto P realmente satisfaz a equação da curva $y_2 \equiv x_3 + Ax + B \ \ (mod\ p)$. É crucial para validar pontos, incluindo o ponto base G.
- **add(self, P, Q):** Implementa a operação de adição de pontos na curva, seguindo a "regra da corda e da tangente". Este método lida com três cenários principais:
    - Adição com o ponto no infinito (*None*), que age como elemento neutro.
    - Adição de dois pontos distintos (P + Q).
    - Duplicação de um ponto (P + P), que usa a tangente ao invés da secante.
    - A utilização de *mod_inverse* da biblioteca ***sympy*** é essencial para realizar as divisões no contexto da aritmética modular.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

- **multiply(self, P, n):** Implementa a multiplicação escalar (ou "multiplicação de ponto") de um ponto P por um escalar *n*. Utiliza o eficiente algoritmo "*double-and-add*", que executa a operação em tempo logarítmico em relação a n, tornando-a computacionalmente viável mesmo para escalares grandes. Este algoritmo é a base para a geração de chaves públicas e o cálculo do segredo compartilhado em ECC.

Função is_on_curve(P)

```
# Verifica se um ponto P(x, y) está na curva.

def is_on_curve(self, P):
    if P is None: # O ponto no infinito (O) é considerado na curva.
        return True
    x, y = P

    # Verifica se y² % p == (x³ + ax + b) % p
    return (y ** 2 - x ** 3 - self.a * x - self.b) % self.p == 0
```

Figura 5 - Implementação do Método is_on_curve para Verificação de Pontos na Curva

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

Função add(P, Q)

```
# Implementa a adição de dois pontos P e Q na curva elíptica.

def add(self, P, Q):
    # Considera os casos para o ponto no infinito (None),
    # pontos distintos e duplicação de pontos.

    if P is None: return Q    # P + O = P
    if Q is None: return P    # Q + O = Q

    x1, y1 = P  # Obtêm as coordenadas de P
    x2, y2 = Q  # Obtêm as coordenadas de Q

    # Se P = -Q (reflexão no eixo x), a soma é o ponto no infinito (O)
    if x1 == x2 and y1 != y2:
        return None

    if P == Q: # Duplicação de ponto (P + P)
        # A inclinação da tangente é [(3x₁² + a) * (2y₁)⁻¹] mod p
        # É necessário verificar se 2y₁ ≠ 0 (mod p) para o inverso modular

        if (2 * y1) % self.p == 0: # Caso especial onde a tangente é vertical
            return None # Resulta no ponto no infinito
        s = (3 * x1**2 + self.a) * mod_inverse(2 * y1, self.p)

    else: # Adição de dois pontos distintos (P + Q)
        # A inclinação da secante é [(y₂ - y₁) * (x₂ - x₁)⁻¹] mod p
        if (x2 - x1) % self.p == 0: # Caso especial onde a linha é vertical
            return None # Resulta no ponto no infinito
        s = (y2 - y1) * mod_inverse(x2 - x1, self.p)

    # Garante que a inclinação está no intervalo do corpo finito
    s %= self.p

    # Calcula a coordenada x do ponto resultante
    x3 = (s ** 2 - x1 - x2) % self.p

    # Calcula a coordenada y do ponto resultante
    y3 = (s * (x1 - x3) - y1) % self.p

    return (x3, y3)  # Retorna o novo ponto na curva
```

Figura 6 - Implementação Detalhada do Método de Adição de Pontos (add) na Curva Elíptica

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

Função multiply(P, n)

```
# Implementa a multiplicação escalar de um ponto P
# por um inteiro n usando o algoritmo 'double-and-add'.

def multiply(self, P, n):

    if n == 0: return None # 0 * P = O
    if n < 0: # Para n negativo, calcula -n * (-P)
        P = (P[0], -P[1] % self.p) # Inverso aditivo de P
        n = abs(n)

    R = None # Inicializa o resultado como o ponto no infinito
    Q = P    # Q é o ponto que será dobrado (P, 2P, 4P, 8P, ...)

    while n > 0:
        # Se o bit menos significativo de n for 1, adiciona Q a R
        if n % 2 == 1:
            R = self.add(R, Q)
        Q = self.add(Q, Q) # Dobra Q
        n //= 2 # Desloca n para a direita (divide por 2)
    return R
```

Figura 7 - Implementação do Algoritmo Double-and-Add para Multiplicação Escalar de Pontos

Simulação do Protocolo ECDH: A segunda parte do código demonstra o protocolo Elliptic Curve Diffie-Hellman (ECDH).

- São definidos os parâmetros públicos da curva e um ponto base G.
- Alice e Bob, independentemente, geram suas chaves privadas aleatórias (*priv_A*, *priv_B*).
- A partir dessas chaves privadas e do ponto base G, cada um calcula sua chave pública (*pub_A*, *pub_B*) usando a multiplicação escalar.
- As chaves públicas são trocadas.
- Finalmente, Alice usa a chave pública de Bob e sua própria chave privada, e Bob usa a chave pública de Alice e sua própria chave privada, para computar o *mesmo* segredo compartilhado. Este segredo é a base para estabelecer uma comunicação segura.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

Simulação

```
# ------------------------------------------------------------ #
# Simulação do protocolo Elliptic Curve Diffie-Hellman (ECDH)
# ------------------------------------------------------------ #

# 1. Definição dos parâmetros de domínio da curva (p, a, b, G)
# Escolhemos um exemplo simples para demonstração
p = 97  # Um número primo para o corpo finito Fp
a = 2   # Coeficiente 'a' da equação de Weierstrass
b = 3   # Coeficiente 'b' da equação de Weierstrass
curve = EllipticCurve(a, b, p)

# Ponto base G, que deve pertencer à curva
G = (3, 6)
# É fundamental verificar se G está na curva definida
if not curve.is_on_curve(G):
    raise ValueError(f"O ponto base {G} não está na curva y^2 = x^3 + {a}x + {b} mod {p}")
# Também é importante que a curva seja não-singular, ou seja, 4a^3 + 27b^2 ≠ 0 mod p
discriminant = (4 * (a**3) + 27 * (b**2)) % p
if discriminant == 0:
    raise ValueError(f"A curva é singular (4a^3 + 27b^2 = 0 mod p). Escolha outros a e b.")

print(f"Parâmetros da Curva Elíptica: y^2 = x^3 + {a}x + {b} mod {p}")
print(f"Ponto Base (Gerador) G: {G}\n")

# 2. Geração de chaves privadas para Alice e Bob
# A chave privada é um número inteiro aleatório no intervalo [1, n-1],
# onde n é a ordem do subgrupo gerado por G. Para simplificar,
# usaremos p-1 como um limite superior razoável para este exemplo,
# embora em aplicações reais 'n' seja um valor específico para a curva.
priv_A = randint(1, p - 1) # Chave privada de Alice
priv_B = randint(1, p - 1) # Chave privada de Bob

print(f"Chave Privada de Alice (dA): {priv_A}")
print(f"Chave Privada de Bob (dB): {priv_B}\n")

# 3. Cálculo das chaves públicas
# A chave pública é calculada multiplicando o ponto base G pela chave privada.
pub_A = curve.multiply(G, priv_A) # Chave pública de Alice (QA = dA * G)
pub_B = curve.multiply(G, priv_B) # Chave pública de Bob (QB = dB * G)

print(f"Chave Pública de Alice (QA): {pub_A}")
print(f"Chave Pública de Bob (QB): {pub_B}\n")

# 4. Troca de chaves públicas e cálculo do segredo compartilhado
# Alice e Bob trocam suas chaves públicas.
# Cada um usa a chave pública do outro e sua própria chave privada para
# calcular o segredo compartilhado.
shared_A = curve.multiply(pub_B, priv_A) # Alice calcula S = dA * QB
shared_B = curve.multiply(pub_A, priv_B) # Bob calcula S = dB * QA

print(f"Segredo Compartilhado Calculado por Alice: {shared_A}")
print(f"Segredo Compartilhado Calculado por Bob: {shared_B}")

# A asserção final verifica se ambos calcularam o mesmo segredo,
# demonstrando a eficácia do ECDH.
assert shared_A == shared_B, "As chaves compartilhadas não são iguais!"
print("\nAs chaves compartilhadas são iguais. O protocolo ECDH foi bem-sucedido!")
```

Figura 8 - Simulação do Protocolo Elliptic Curve Diffie-Hellman (ECDH)

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

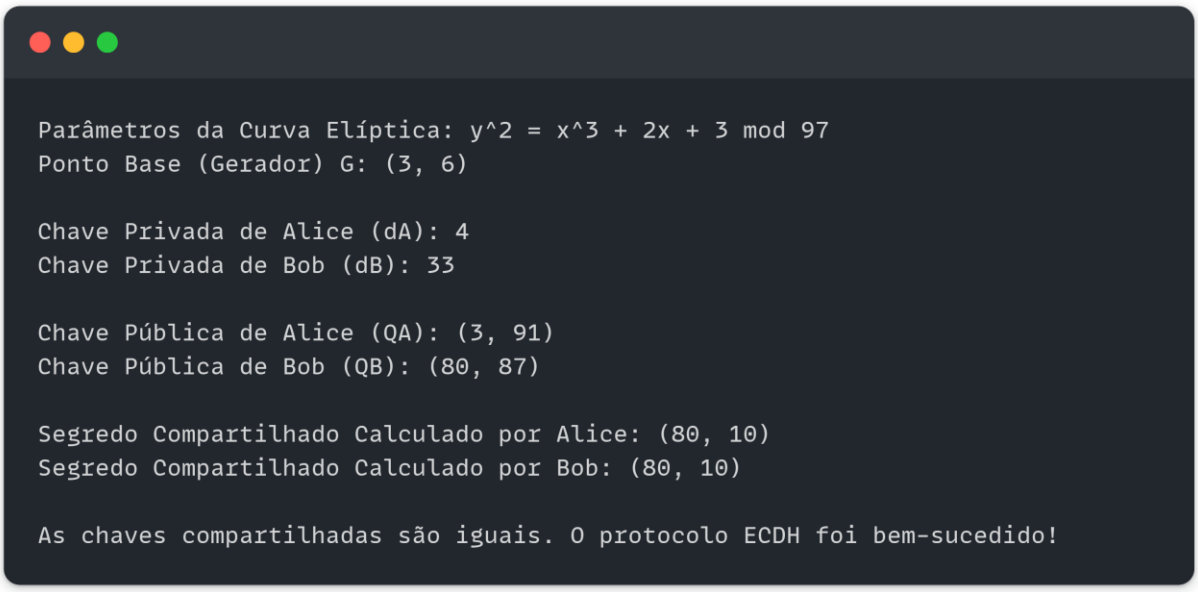

Figura 9 - Exemplo de Saída da Simulação do Protocolo Elliptic Curve Diffie-Hellman (ECDH)

A robustez deste sistema reside no Problema do Logaritmo Discreto em Curvas Elípticas (*ECDLP*) : é trivial para um atacante interceptar as chaves públicas pub_A e pub_B, mas é computacionalmente intratável derivar as chaves privadas priv_A ou priv_B a partir delas, o que impediria a computação do segredo compartilhado.

Esta implementação serve como um alicerce claro para entender como a matemática das curvas elípticas se traduz em um sistema criptográfico prático e eficiente, justificando a relevância da ECC na segurança digital moderna.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

4.3 Implementação do ECDSA

Funções ECDSA

```
# Calcula o hash SHA-256 da mensagem como inteiro.
def hash_message(message):

# Calcula a ordem do ponto G (mínimo n tal que nG = O).
def calculate_order(curve, G):

# Gera chave privada aleatória e chave pública correspondente.
def generate_keypair(curve, G, n):

# Assina uma mensagem com a chave privada d.
def sign(curve, G, n, d, message):

# Verifica uma assinatura digital usando a chave pública Q.
def verify(curve, G, n, Q, message, signature):
```

Figura 10 - Conjunto de Funções do Protocolo ECDSA Implementadas em Python

hash_message(message): Esta função converte uma mensagem arbitrária em um valor numérico determinístico, aplicando o algoritmo de hash SHA-256 da biblioteca hashlib. O resultado é convertido de hexadecimal para inteiro, sendo este valor que participa dos cálculos internos da assinatura digital. A função de hash é fundamental para garantir a integridade e a unicidade da mensagem assinada.

calculate_order(curve,G): Calcula a ordem do ponto base G, ou seja, o menor inteiro positivo nnn tal que $n \cdot G = O$, onde O é o ponto no infinito. Esta função é crucial, pois a segurança dos protocolos ECDSA e ECDH depende da escolha de um ponto de ordem elevada. O cálculo é realizado iterativamente através de multiplicações sucessivas até que o ponto neutro seja atingido.

generate_keypair(curve, G, n): Gera um par de chaves criptográficas para uso com curvas elípticas. A chave privada é um inteiro aleatório no intervalo $[1, n-1]$, gerado com randint, e a chave pública correspondente é obtida pela multiplicação escalar do ponto base G pela chave privada. Este processo reflete a assimetria central da criptografia de chave pública: fácil calcular $\boldsymbol{G}Q = d \cdot G$, mas computacionalmente inviável determinar ddd a partir de Q.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

sign(curve, G, n, d, message): Implementa a geração de uma assinatura digital ECDSA para uma dada mensagem. O processo envolve:

- o hash da mensagem,
- a escolha de um valor efêmero aleatório k,
- o cálculo de um ponto $k \cdot G$,
- e o uso de $\boldsymbol{k^{-1}\ mod\ n}$ para derivar o valor *s* da assinatura. As variáveis *r* e *s* compõem o par de assinatura. Caso $\boldsymbol{r = 0}\ ou\ \boldsymbol{s = 0}$, um novo k é escolhido, garantindo segurança e não repetição.

verify(curve, G, n, Q, message, signature): Executa a verificação de uma assinatura digital recebida, utilizando a chave pública Q. Recalcula o hash da mensagem, aplica inverso modular de sss, e reconstrói o ponto $\boldsymbol{P = u_1 \cdot G + u_2 \cdot Q}$. A assinatura é válida se a coordenada xxx deste ponto satisfaz $\boldsymbol{x \equiv r\ mod\ n}$. Este procedimento garante tanto a autenticidade da origem quanto a integridade da mensagem assinada.

```python
# Calcula o hash SHA-256 da mensagem como inteiro.
def hash_message(message):
    digest = sha256(message.encode()).hexdigest()
    return int(digest, 16)


# Calcula a ordem do ponto G (mínimo n tal que nG = O).
def calculate_order(curve, G):
    for i in range(1, curve.p + 1):
        if curve.multiply(G, i) is None:
            return i
    return None

# Gera chave privada aleatória e chave pública correspondente.
def generate_keypair(curve, G, n):
    d = randint(1, n - 1)  # chave privada
    Q = curve.multiply(G, d)  # chave pública
    return d, Q
```

Figura 11 - Funções Auxiliares e Geração de Chaves ECDSA

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

```
# Assina uma mensagem com a chave privada d."""
def sign(curve, G, n, d, message):
    e = hash_message(message) % n
    while True:
        k = randint(1, n - 1)
        R = curve.multiply(G, k)
        if R is None:
            continue
        r = R[0] % n
        if r == 0:
            continue
        try:
            k_inv = mod_inverse(k, n)
        except ValueError:
            continue
        s = (k_inv * (e + r * d)) % n
        if s == 0:
            continue
        return (r, s)

# Verifica uma assinatura digital usando a chave pública Q.
def verify(curve, G, n, Q, message, signature):
    r, s = signature
    if not (1 ⩽ r < n and 1 ⩽ s < n):
        return False
    e = hash_message(message) % n
    try:
        w = mod_inverse(s, n)
    except ValueError:
        return False
    u1 = (e * w) % n
    u2 = (r * w) % n
    P = curve.add(
        curve.multiply(G, u1),
        curve.multiply(Q, u2)
    )
    if P is None:
        return False
    x, _ = P
    return r == x % n
```

Figura 12 - Funções de Assinatura e Verificação ECDSA

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

ECDSA

```
from ecdh import EllipticCurve
from ecdsa import *

# Define curva e G
curve = EllipticCurve(a=2, b=3, p=97)
G = (3, 6)
n = calculate_order(curve, G)

# Geração de chave
d, Q = generate_keypair(curve, G, n)

# Mensagem
msg = "Segurança digital com ECC é eficiente"
assinatura = sign(curve, G, n, d, msg)

print("Assinatura:", assinatura)
print("Verificação:", "Válida ✅" if verify(curve, G, n, Q, msg, assinatura) else "Inválida ❌")
```

Figura 13 - Simulação do Protocolo Elliptic Curve Diffie-Hellman (ECDH)

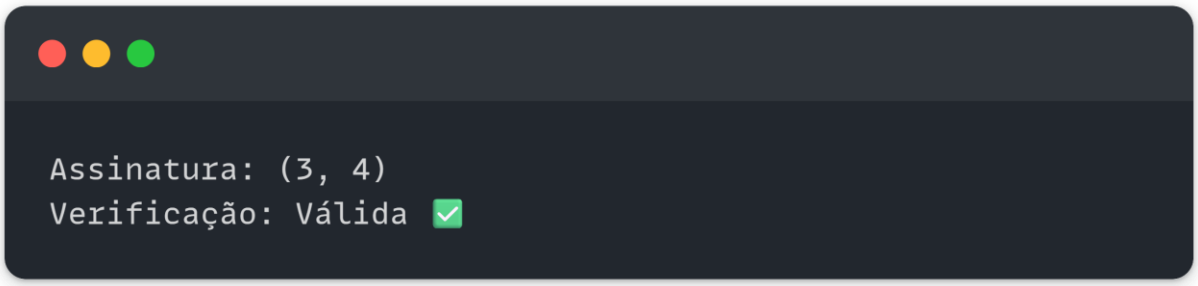

Figura 14 - Exemplo de Saída da Simulação do ECDSA

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

5 ANÁLISE E DISCUSSÃO

A implementação da Criptografia de Curva Elíptica (ECC) não se limita apenas à compreensão das suas operações fundamentais; ela se estende a uma análise crítica de sua eficiência, segurança e escolhas de design. Esta seção aprofunda a complexidade computacional das operações centrais, discute vulnerabilidades conhecidas e compara curvas elípticas proeminentes utilizadas na prática.

5.1 Análise de Complexidade Computacional

A eficiência da ECC é uma de suas maiores vantagens, e isso é particularmente evidente na operação de multiplicação escalar, que é a base para a geração de chaves e o cálculo de segredos compartilhados.

- **Multiplicação Escalar: O(logn)**.
    - A multiplicação escalar kP (somar o ponto P a si mesmo k vezes) é realizada de forma eficiente utilizando algoritmos como o "double-and add" (dobrar e somar).
    - Este algoritmo opera em tempo logarítmico em relação ao valor de k, ou seja, $O(\log k)$ ou $O(\log n)$ se n for a ordem do subgrupo.
    - Isso significa que, mesmo para chaves privadas (escalares) muito grandes, o número de operações de adição e duplicação de pontos necessárias é proporcional ao número de bits do escalar, tornando o cálculo rápido e prático. Esta característica contrasta favoravelmente com algoritmos que dependem de exponenciação modular com base em inteiros muito grandes, como o RSA, onde o custo computacional tende a ser maior para um nível de segurança equivalente.

5.2 Vulnerabilidades Conhecidas

A segurança da ECC, embora robusta, não está isenta de considerações e potenciais vetores de ataque que devem ser mitigados.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

Tipo de Ataque	Descrição	Contramedidas/Resistência
Força Bruta (Baby-step Giant-step)	Métodos genéricos para resolver o Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP). Curvas mal escolhidas ou com parâmetros de segurança insuficientes podem ser vulneráveis. A complexidade é significativamente maior que ataques a RSA ou ElGamal tradicionais.	A segurança reside na ausência de algoritmos sub-exponenciais para ECDLP em curvas bem escolhidas. Para resistir, o tamanho do campo primo p (e a ordem n do subgrupo) deve ser suficientemente grande (tipicamente 256 bits ou mais).
Canal Lateral (Side-channel Attacks)	Exploram informações vazadas inadvertidamente durante a execução de operações criptográficas em hardware (e.g., tempo de execução, consumo de energia, emissões eletromagnéticas). Podem revelar bits do escalar secreto (chave privada), especialmente em implementações diretas do algoritmo "double-and-add".	Uso de implementações cegas ("blinding"), duplicação condicional ou outras técnicas que tornam as operações de ponto indistinguíveis, independentemente do valor do bit do escalar, aumentando a resistência a tais ataques.

Tabela 4 - Tipos de ataques a criptossistemas baseados em curvas elípticas e respectivas contramedidas

5.3 Comparativo de Curvas

A escolha da curva elíptica é um fator crítico que impacta tanto a segurança quanto a performance de um sistema ECC. Existem diversas curvas padronizadas e amplamente aceitas, cada uma com suas características.

Característica	NIST P-256 (secp256r1)	Curve25519
Padrão/Origem	Padronizada pelo NIST (National Institute of Standards and Technology) dos EUA.	Introduzida por Daniel J. Bernstein.
Aceitação/Uso	Amplamente aceita e tradicionalmente usada em protocolos como TLS, SSL e assinaturas de software.	Eficiente, amplamente utilizada em aplicações modernas como Signal, OpenSSH e TLS 1.3.
Segurança	Considerada robusta, mas com escrutínio devido à origem governamental.	Projetada para ser mais segura contra falhas de implementação (imune a muitos ataques por canal lateral).
Eficiência	Boa, mas pode ser menos otimizada para certas implementações.	Mais eficiente, com estrutura algébrica que simplifica a aritmética de ponto, resultando em implementações mais rápidas e menos propensas a erros.

Tabela 5 - Comparação entre as curvas elípticas NIST P-256 e Curve25519.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

A escolha entre estas e outras curvas depende do balanço desejado entre interoperabilidade (padronização), performance e robustez contra ataques específicos, especialmente em ambientes onde a implementação precisa ser otimizada ou é sensível a vazamentos de informação.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

6 CONCLUSÃO E REFLEXÃO

Este trabalho explorou o universo das Curvas Elípticas e sua aplicação revolucionária na Criptografia de Curva Elíptica (ECC). Desde os fundamentos matemáticos que definem essas estruturas complexas até a sua concretização em protocolos de segurança modernos como o ECDH, buscamos demonstrar a elegância e a eficiência que a ECC oferece para o cenário digital atual.

6.1 Desafios da Implementação

A jornada de implementar as operações de curvas elípticas sobre corpos finitos trouxe consigo desafios que, ao serem superados, aprofundaram a compreensão dos conceitos teóricos:

- **Aritmética Modular Correta**: Um dos principais obstáculos foi garantir que todas as operações aritméticas (adição, subtração, multiplicação e, crucialmente, divisão via inverso modular) fossem executadas corretamente dentro do corpo finito F_p. Um erro na aplicação do operador módulo em qualquer etapa pode invalidar completamente os resultados e comprometer a segurança.
- **Validação de Pontos e Condição de Não-Singularidade**: Assegurar que os pontos utilizados nas operações (especialmente o ponto base G) realmente pertencem à curva e que a curva em si é não-singular foi um passo fundamental. A condição de não-singularidade ($\Delta = -16(4A^3 + 27B^2) \neq 0 \ (mod\ p)$) é vital para que a lei de grupo seja válida e consistente.
- **Gestão do Ponto no Infinito (O)**: A incorporação do conceito de "Ponto no Infinito" (representado como None na implementação) como o elemento neutro do grupo exigiu um tratamento cuidadoso nas funções de adição e duplicação, garantindo que as regras geométricas de Bézout fossem respeitadas algebricamente.
- **Garantia de Segurança e Correção nos Cálculos**: Embora a implementação tenha sido para fins demonstrativos, a constante preocupação em replicar com precisão os algoritmos criptográficos (como o "double-and-add") e em validar os resultados (como a igualdade das chaves compartilhadas no ECDH) foi essencial para consolidar a compreensão da robustez da ECC.

6.2 Importância das Curvas Elípticas na Criptografia Moderna

A criptografia com curvas elípticas transcende a esfera acadêmica para se tornar uma das tecnologias mais importantes e onipresentes na segurança digital contemporânea. Sua importância pode ser resumida em alguns pontos cruciais:

- **Eficiência e Desempenho Superior**: A principal vantagem da ECC reside na sua capacidade de oferecer níveis de segurança equivalentes aos de criptossistemas

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

mais antigos, como o RSA, mas com chaves drasticamente menores. Isso se traduz em cálculos mais rápidos, menor consumo de energia e menor necessidade de largura de banda, fatores críticos para a infraestrutura digital moderna.

- **Adequação a Dispositivos com Recursos Limitados**: A eficiência da ECC a torna a escolha ideal para uma vasta gama de dispositivos com poder de processamento, memória e bateria restritos, como smartphones, cartões de crédito, sensores da Internet das Coisas (IoT) e outros dispositivos embarcados. Sem a ECC, proteger dados sensíveis nesses ambientes seria um desafio muito maior.
- **Fundamentação Matemática Robusta**: A segurança da ECC baseia-se na dificuldade do Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP). Acredita-se que este problema seja significativamente mais "difícil" de resolver do que os problemas matemáticos que sustentam o RSA (fatoração de inteiros) ou o ElGamal tradicional (logaritmo discreto em corpos finitos), permitindo que a ECC forneça o mesmo nível de segurança com chaves menores.
- **Versatilidade em Aplicações Criptográficas**: A estrutura de grupo das curvas elípticas permite a adaptação de criptossistemas de chave pública já existentes, como o Diffie-Hellman e o DSA, para versões mais eficientes (ECDH e ECDSA). Isso prova a versatilidade da ECC em estabelecer segredos compartilhados e em criar assinaturas digitais, garantindo autenticidade e integridade da mensagem.

Em suma, a criptografia de curvas elípticas não é apenas uma área avançada da matemática aplicada, mas uma tecnologia fundamental que sustenta a comunicação segura em um mundo cada vez mais conectado e dependente de dispositivos digitais. Ela representa um pilar essencial para a privacidade e a segurança em praticamente todas as interações online.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao

7 REFERÊNCIAS

ANGULO, Rigo Julian Osorio. Criptografia de Curvas Elípticas. 2017. 100 f. Dissertação (Mestrado em Matemática) – Instituto de Matemática e Estatística, Universidade Federal de Goiás, Goiânia, 2017.

BARROS, Filipe Tancredo. Estudo e Implementação do Protocolo ECDSA. 2015. 213 f. Monografia (Bacharelado em Ciência da Computação) – Departamento de Ciência da Computação, Instituto de Ciências Exatas, Universidade de Brasília, Brasília, 2015.

COELHO, José Gustavo. Curvas elípticas sobre corpos finitos. 2023. 51 f. Dissertação (Mestrado em Matemática) – Instituto de Ciências Exatas, Universidade Federal de Minas Gerais, Belo Horizonte, 2020.

MADEIRA, Fernando Lima. Aplicação Prática de Criptografia de Curvas Elípticas (ECC). 2021. Monografia (Licenciatura em Computação) – Departamento de Ciência da Computação, Instituto de Ciências Exatas, Universidade de Brasília, Brasília, 2021.

NOGUEIRA, Guilherme da Silva. Criptografia de Curva Elíptica. 2012. Trabalho de Conclusão de Curso (Tecnologia em Segurança da Informação) – Faculdade de Tecnologia de Americana, Americana, SP, 2012.

RAMOS, Marcos da Silva. Geração de Parâmetros de Domínio de Curvas Elípticas para Uso em Criptografia. 2015. Monografia (Bacharelado em Engenharia de Software) – Faculdade UnB Gama, Universidade de Brasília, Brasília, DF, 2015.

SOUZA, Adenilce Oliveira. Pontos Racionais em Curvas Elípticas. 2012. 62 f. Dissertação (Mestrado em Matemática) – Programa de Pós-Graduação em Matemática, Faculdade de Matemática, Universidade Federal de Uberlândia, Uberlândia, MG, 2012.

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS
AV. LUANDA SUL, RUA LATERAL VIA S10, TALATONA – MUNICÍPIO DO BELAS – LUANDA/ANGOLA
Telefones: +244 226 430 334/+244 226 430 330 – Email: geral@isptec.co.ao