UNIVERSIDADE FEDERAL DO MARANHÃO
Fundação Instituída nos termos da Lei 5.152 de 21/10/1966 - São Luís - MA
Centro de Ciências Exatas e Tecnologia
Curso de Matemática – Bacharelado

Fernando Torreal Fonseca

# Aspectos Matemáticos da Criptografia: O Algoritmo RSA e Aplicações

São Luís - MA
2025

Fernando Torreal Fonseca

# Aspectos Matemáticos da Criptografia: O Algoritmo RSA e Aplicações

Monografia (Trabalho de Conclusão de Curso) apresentada à Coordenadoria dos cursos de Matemática, da Universidade Federal do Maranhão, como requisito parcial para obtenção do grau de Bacharel em Matemática.

Curso de Matemática – Bacharelado

Universidade Federal do Maranhão

Orientador: Prof. Me. Cleber Araujo Cavalcanti

São Luís - MA

2025

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Fonseca, Fernando Torreal.
Aspectos Matemáticos da Criptografia: O Algoritmo RSA e Aplicações / Fernando Torreal Fonseca. - 2025.
37 p.

Orientador(a): Cleber Araujo Cavalcanti.
Monografia (Graduação) - Curso de Matemática, Universidade Federal do Maranhão, São Luís - Ma, 2025.

1. Criptografia. 2. Teoria dos Números. 3. Segurança da Informação. 4. Rsa. I. Cavalcanti, Cleber Araujo. II. Título.

Fernando Torreal Fonseca

# Aspectos Matemáticos da Criptografia: O Algoritmo RSA e Aplicações

Monografia (Trabalho de Conclusão de Curso) apresentada à Coordenadoria dos cursos de Matemática, da Universidade Federal do Maranhão, como requisito parcial para obtenção do grau de Bacharel em Matemática.

Trabalho **APROVADO**. São Luís - MA, 08/08/2025

---

**Prof. Me. Cleber Araujo Cavalcanti**
DEMAT/UFMA
Orientador

---

**Prof. Dr. Anselmo Baganha Raposo Junior**
DEMAT/UFMA
Primeiro Examinador

---

**Prof. Dr. Ivaldo Paz Nunes**
DEMAT/UFMA
Segundo Examinador

*A tudo que doeu e me fez continuar.*

# Agradecimentos

Agradeço aos meus pais, que sempre acreditaram em mim. Pelo apoio silencioso, pelas palavras firmes e pela base que me deram para seguir em frente, minha eterna gratidão.

Ao meu orientador, Prof. Me. Cleber Araújo Cavalcanti, agradeço profundamente por não ter desistido de mim. Sua paciência, compreensão e firmeza foram essenciais para que este trabalho saísse do papel e ganhasse forma.

*"You become unstoppable when you heal in silence and grow in private."*

Unknown

# Resumo

A criptografia de chave pública é um dos pilares fundamentais da segurança da informação moderna, possibilitando a comunicação segura em ambientes digitais. Entre os algoritmos mais consagrados nesse campo está o RSA, cuja robustez se baseia em princípios da Teoria dos Números.

Este trabalho tem como objetivo apresentar uma abordagem teórica e aplicada do algoritmo RSA, com foco nos aspectos matemáticos que sustentam sua estrutura e segurança. A partir de conceitos como aritmética modular, função totiente de Euler, primalidade e Teorema Chinês do Resto, são discutidas a geração de chaves, a cifragem e a decifragem de mensagens.

Também são abordadas as principais aplicações do RSA em sistemas modernos de autenticação e sigilo, os padrões técnicos que o regulamentam, suas limitações práticas e os desafios impostos pela computação quântica. O estudo busca evidenciar a importância da matemática como ferramenta essencial para o desenvolvimento e a análise crítica de sistemas criptográficos.

**Palavras-chave**: Teoria dos Números, Criptografia, Segurança da Informação, RSA, Computação Quântica.

# Abstract

Public key cryptography is one of the foundational pillars of modern information security, enabling secure communication in digital environments. Among the most well-established algorithms in this field is RSA, whose robustness relies on principles of Number Theory.

This study aims to present a theoretical and applied approach to the RSA algorithm, focusing on the mathematical foundations that support its structure and security. Based on concepts such as modular arithmetic, Euler's totient function, primality, and the Chinese Remainder Theorem, this work discusses the processes of key generation, encryption, and decryption.

It also addresses the main applications of RSA in modern authentication and confidentiality systems, the technical standards that regulate it, its practical limitations, and the challenges posed by quantum computing. The study seeks to highlight the role of mathematics as an essential tool for the development and critical analysis of cryptographic systems.

**Keywords**: Number Theory, Cryptography, Information Security, RSA, Quantum Computing.

# Lista de abreviaturas e siglas

| | |
|---|---|
| AES | Advanced Encryption Standard |
| CRT | Chinese Remainder Theorem (Teorema Chinês do Resto) |
| ECC | Elliptic Curve Cryptography (Criptografia de Curvas Elípticas) |
| GPG | GNU Privacy Guard |
| ICP-Brasil | Infraestrutura de Chaves Públicas Brasileira |
| IETF | Internet Engineering Task Force |
| MIT | Massachusetts Institute of Technology |
| OAEP | Optimal Asymmetric Encryption Padding |
| OpenSSL | Open Source Secure Sockets Layer |
| PJe | Processo Judicial Eletrônico |
| PKCS | Public-Key Cryptography Standards |
| RSA | Rivest–Shamir–Adleman |
| SHOR | Algoritmo de Shor (não é sigla, mas citado como autor-chave) |
| S/MIME | Secure/Multipurpose Internet Mail Extensions |
| SSL/TLS | Secure Sockets Layer / Transport Layer Security |

# Sumário

# Introdução

A criptografia está presente em praticamente todos os aspectos da vida digital contemporânea. Desde uma simples troca de mensagens entre amigos até transações bancárias e autenticações em sistemas governamentais, há um algoritmo silenciosamente garantindo que as informações sejam transmitidas com segurança: o RSA (Rivest-Shamir-Adleman). Criado em 1977 por Ron Rivest, Adi Shamir e Leonard Adleman, esse sistema criptográfico assimétrico revolucionou a forma como se protege dados em ambientes abertos, permitindo que pessoas que nunca se comunicaram antes possam trocar informações sigilosas com confiança.

Mais do que um algoritmo de segurança, o RSA é um exemplo clássico de como a matemática pura, por vezes considerada abstrata demais, se conecta diretamente à prática. Conceitos como aritmética modular, números primos, função totiente de Euler e teoremas clássicos da Teoria dos Números são os pilares que sustentam sua estrutura. Entender como o RSA funciona é também compreender o poder da matemática quando aplicada a problemas reais de proteção da informação.

Este trabalho tem como objetivo apresentar uma abordagem teórica e aplicada do algoritmo RSA, partindo dos aspectos matemáticos que o tornam possível até suas aplicações práticas em sistemas modernos. Serão discutidas a construção das chaves, o processo de cifragem e decifragem, o papel da teoria dos números, os padrões técnicos associados, suas limitações e os riscos emergentes trazidos pela computação quântica.

Ao longo do texto, buscou-se construir uma narrativa que não apenas explique o funcionamento técnico do RSA, mas que valorize o elo entre teoria e prática, entre o número e a segurança, entre o papel acadêmico e a realidade computacional.

# 1 Aspectos Matemáticos da Criptografia

Antes de explorar o funcionamento do algoritmo RSA, é necessário compreender as ferramentas matemáticas que o tornam possível. Este capítulo aborda os principais conceitos de aritmética modular, estruturas algébricas e propriedades numéricas fundamentais para a criptografia.

## 1.1 Aritmética Modular

A aritmética modular é um ramo da Teoria dos Números que estuda os inteiros sob o ponto de vista da congruência. A ideia central é considerar dois números inteiros equivalentes quando apresentam o mesmo resto em uma divisão por um inteiro fixo, denominado *módulo*. Tal abordagem é fundamental para algoritmos criptográficos, como o RSA, que operam essencialmente sobre operações aritméticas em conjuntos finitos de inteiros (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.1.1 Congruências

Dois inteiros $a$ e $b$ são ditos congruentes módulo $n$, onde $n \in \mathbb{N}$, se sua diferença for divisível por $n$, ou seja, se $n \mid (a - b)$. Essa relação é denotada por:

$$a \equiv b \pmod{n} \tag{1.1}$$

Como exemplo, temos $17 \equiv 5 \pmod{12}$, pois $17 - 5 = 12$ é divisível por 12. Nesse sistema, os números "reiniciam" após atingir o valor do módulo, comportamento que motiva o apelido de *aritmética do relógio* (como um relógio que reinicia após 12 horas).

Figura 1.1 – Ilustração da congruência $17 \equiv 5 \pmod{12}$ em um círculo modular. O número 17 ocupa a mesma posição que o 5 em um sistema com 12 elementos, representando a ideia de ciclo.


**Fonte:** Autor.

A relação de congruência é uma relação de equivalência, pois satisfaz as propriedades de reflexividade, simetria e transitividade. Além disso, ela preserva as operações aritméticas fundamentais:

- Se $a \equiv b \pmod{n}$ e $c \equiv d \pmod{n}$, então:
  - $a + c \equiv b + d \pmod{n}$;
  - $a - c \equiv b - d \pmod{n}$;
  - $a \cdot c \equiv b \cdot d \pmod{n}$.

Tais propriedades permitem o desenvolvimento de estruturas algébricas, como os anéis $\mathbb{Z}_n$, nos quais operam os sistemas criptográficos modernos (ROSEN, 2010; HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.1.2 Inverso Modular

Seja $n \in \mathbb{N}$ fixado. Diz-se que um inteiro $a$ possui um inverso modular módulo $n$ se existe um inteiro $x$ tal que:

$$a \cdot x \equiv 1 \pmod{n} \tag{1.2}$$

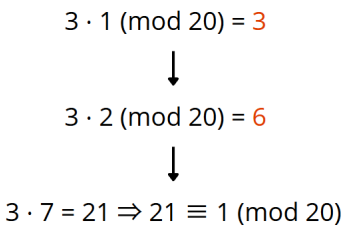
Tal número $x$ é chamado *inverso modular* de $a$ módulo $n$ e é denotado por $a^{-1} \pmod{n}$. Um elemento $a$ admite inverso modular se, e somente se, $\text{mdc}(a, n) = 1$, isto é, se $a$ e $n$ são coprimos (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

O cálculo do inverso modular pode ser realizado por meio do Algoritmo de Euclides Estendido, que será apresentado na próxima subseção. Como exemplo, considere:

$$3 \cdot 7 \equiv 1 \pmod{20} \tag{1.3}$$

Logo, 7 é o inverso modular de 3 módulo 20, pois a multiplicação $3 \cdot 7$ resulta em 21, que deixa resto 1 quando dividido por 20, ou seja, $3 \cdot 7 \equiv 1 \pmod{20}$. Isso satisfaz a definição de inverso modular: um número $x$ tal que $a \cdot x \equiv 1 \pmod{n}$.

Figura 1.2 – Demonstração visual da verificação do inverso modular de 3 módulo 20 por tentativa. A congruência $3 \cdot 7 \equiv 1 \pmod{20}$ confirma que 7 é o inverso buscado.




**Fonte:** Autor.

### 1.1.3 O Algoritmo de Euclides Estendido

O Algoritmo de Euclides Estendido é uma extensão natural do algoritmo de Euclides clássico, utilizado para encontrar o máximo divisor comum entre dois inteiros. Sua importância na criptografia moderna vai além da mera curiosidade matemática: ele é a ferramenta essencial para calcular o inverso modular, que, por sua vez, é o coração da geração de chaves no algoritmo RSA.

Dado dois inteiros positivos $a$ e $n$, com $\text{mdc}(a, n) = 1$, o algoritmo encontra inteiros $x$ e $y$ tais que:

$$ax + ny = 1$$

Nesse contexto, o valor de $x$ corresponde ao inverso modular de $a$ módulo $n$, isto é:

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

**Exemplo 1.1.** *Considere o problema de encontrar o inverso modular de* 3 *módulo* 20*, ou seja, um número* $x$ *tal que* $3x \equiv 1 \pmod{20}$*. Aplicando o algoritmo passo a passo:*

$$\begin{aligned} 20 &= 6 \cdot 3 + 2 \\ 3 &= 1 \cdot 2 + 1 \\ 2 &= 2 \cdot 1 + 0 \end{aligned}$$

*Retrocedendo as equações para encontrar o valor de* $x$*:*

$$\begin{aligned} 1 &= 3 - 1 \cdot 2 \\ &= 3 - 1 \cdot (20 - 6 \cdot 3) \\ &= 3 - 1 \cdot 20 + 6 \cdot 3 \\ &= 7 \cdot 3 - 1 \cdot 20 \end{aligned}$$

*Portanto,* $x = 7$*. Logo, o inverso modular de* 3 *módulo* 20 *é:*

$$3^{-1} \equiv 7 \pmod{20}$$

#### Aplicação no RSA

Esse algoritmo tem papel central na etapa de geração da chave privada $d$, no algoritmo RSA. Ele é a ferramenta usada para encontrar o valor de $d$ a partir da chave pública $e$.

Em essência, o Algoritmo de Euclides Estendido permite que essa etapa seja computacionalmente viável mesmo quando se trabalha com números muito grandes. Ele garante que a relação matemática entre a chave pública e a chave privada seja estabelecida de forma eficiente (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.1.4 Teorema de Fermat

O Pequeno Teorema de Fermat afirma que, se $p$ é primo e $a \not\equiv 0 \pmod{p}$, então $a^{p-1} \equiv 1 \pmod{p}$. Isso permite, por exemplo, simplificar exponenciações com expoentes muito grandes ao trabalhar com congruências. De modo semelhante, o Teorema de Euler afirma que, para $a \in \mathbb{N}$ tal que $\text{mdc}(a, n) = 1$, tem-se:

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

Essa propriedade é essencial para a eficiência computacional do algoritmo RSA, pois permite reduzir o tamanho dos expoentes em operações modulares, garantindo que as cifras e decifras possam ser realizadas de forma prática mesmo com números grandes (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.1.5 Função Totiente de Euler

A função totiente de Euler, denotada por $\varphi(n)$, é definida como o número de inteiros positivos menores que $n$ que são coprimos com $n$. Ou seja, é a quantidade de inteiros $a \in \mathbb{N}$ tal que $\text{mdc}(a, n) = 1$.

Essa função desempenha um papel central em diversos teoremas da Teoria dos Números, como o Teorema de Euler, e também está no coração do algoritmo RSA, pois é utilizada na geração da chave privada.

No caso em que $n = p$ e $p$ é primo, é fácil ver que todos os inteiros positivos $a$ com $1 \leq a < p$ são coprimos com $p$, pois não há divisor comum além de 1. Assim, $\varphi(p) = p - 1$.

Quando $n = pq$, com $p$ e $q$ primos distintos, vale

$$\varphi(pq) = (p-1)(q-1).$$

De fato, entre os inteiros $1, \ldots, pq-1$, os únicos que não são coprimos com $n$ são os múltiplos de $p$ ou de $q$. Há $q-1$ múltiplos de $p$ e $p-1$ múltiplos de $q$ nesse intervalo, e não há múltiplos de $pq$. Pelo princípio da inclusão–exclusão, a quantidade de não coprimos é $(q-1)+(p-1)$. Logo,

$$\varphi(pq) = (pq-1) - \Big[(q-1)+(p-1)\Big] = pq - p - q + 1 = (p-1)(q-1).$$

Veja, por exemplo, (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.1.6 Teorema de Euler

Generalizando o Teorema de Fermat, o Teorema de Euler afirma que, se $\text{mdc}(a, n) = 1$, então:

$$a^{\varphi(n)} \equiv 1 \pmod{n} \tag{1.4}$$

Esse teorema é a base teórica que garante que a exponenciação modular usada no algoritmo RSA é reversível, desde que se utilize o inverso modular de um expoente (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.1.7 Aplicações no RSA

Todos os conceitos abordados nesta seção, congruências, aritmética modular, inverso modular e os teoremas de Fermat e Euler, são aplicados diretamente no funcionamento do algoritmo RSA.

Durante a geração de chaves, por exemplo, a função totiente de Euler é usada para calcular $\varphi(n) = (p-1)(q-1)$, o que permite determinar o inverso modular do expoente público $e$, resultando no expoente secreto $d$, tal que $ed \equiv 1 \pmod{\varphi(n)}$ (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

Além disso, a operação central de cifragem e decifragem no RSA é baseada em exponenciação modular: $c \equiv m^e \pmod{n}$ para cifragem, e $m \equiv c^d \pmod{n}$ para decifragem. Ambas dependem diretamente das propriedades da aritmética modular para funcionar corretamente e de forma segura.

No entanto, é importante destacar que a base da segurança do RSA não está apenas na aritmética modular, mas sobretudo na dificuldade de fatorar o número $n = pq$, quando $p$ e $q$ são primos grandes. Sem o conhecimento desses fatores, torna-se computacionalmente inviável determinar $\varphi(n)$ e, consequentemente, a chave privada $d$.

Essas operações serão detalhadas no Capítulo 3, onde o algoritmo RSA será descrito formalmente, acompanhado de exemplos e aplicações práticas. Também será discutido como essas operações se conectam à dificuldade da fatoração de inteiros, base da segurança do sistema.

Conforme detalhado por Menezes et al. (MENEZES; OORSCHOT; VANSTONE, 1996), a estrutura matemática sobre a qual o RSA é construído é o que permite sua eficácia criptográfica, mas também impõe os desafios de segurança que serão analisados posteriormente.

## 1.2 Teoria dos Grupos, Anéis e Corpos Finitos

A criptografia moderna, especialmente algoritmos de chave pública como o RSA, fundamenta-se em estruturas algébricas bem definidas, como grupos, anéis e corpos finitos. A compreensão dessas estruturas não apenas aprofunda o entendimento desses algoritmos,

mas também fornece a base matemática para analisar sua segurança e comportamento (MENEZES; OORSCHOT; VANSTONE, 1996; HOFFSTEIN; PIPHER; SILVERMAN, 2014). Nesta seção, abordaremos brevemente os conceitos fundamentais dessas estruturas, focando em como suas propriedades são diretamente exploradas pelo RSA e outros sistemas criptográficos.

### 1.2.1 Grupos multiplicativos

Em termos gerais, um grupo é um conjunto não vazio $G$, munido de uma operação binária $*$, que satisfaz quatro propriedades fundamentais: fechamento, associatividade, existência de elemento neutro e existência de inverso para cada elemento. Quando a operação é comutativa, o grupo é dito abeliano (ROSEN, 2010).

Na prática da criptografia, lidamos frequentemente com grupos formados por inteiros sob uma operação de multiplicação módulo $n$. O conjunto dos inteiros positivos menores que $n$ e coprimos com $n$, sob multiplicação módulo $n$, forma um grupo, pois:

- a multiplicação é bem definida e fechada;
- a operação é associativa;
- o número 1 atua como elemento neutro;
- todo elemento possui inverso multiplicativo módulo $n$, desde que seja coprimo com $n$.

Por exemplo, o conjunto $\{1, 2, 4, 7, 8, 11\}$ forma um grupo sob multiplicação módulo 12, já que todos os seus elementos são coprimos com 12 e possuem inversos modulares.

Esses grupos são de importância central para algoritmos como o RSA. É sobre a estrutura de grupos multiplicativos, particularmente o conjunto dos inteiros coprimos com $n$ sob a multiplicação módulo n, que o RSA opera ao lidar com potências modulares e o cálculo de inversos multiplicativos, ambos essenciais para suas operações de cifragem e decifragem. (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.2.2 Estrutura de anéis e corpos finitos

Além dos grupos, a álgebra moderna define outras estruturas fundamentais para a construção de sistemas criptográficos: os anéis e os corpos. Essas estruturas generalizam operações aritméticas conhecidas e possibilitam trabalhar com mais robustez sobre conjuntos finitos, como é o caso dos sistemas utilizados em criptografia.

Um *anel* é um conjunto $R$, equipado com duas operações, geralmente chamadas de adição e multiplicação, que satisfazem certas propriedades: a adição forma um grupo

abeliano, a multiplicação é associativa e a distributividade é válida entre as duas operações. Quando a multiplicação também admite elemento neutro e inversos para todos os elementos não nulos, temos um *corpo* (ROSEN, 2010).

No contexto criptográfico, o que mais interessa são os *corpos finitos*, também chamados de *campos finitos*. Um corpo finito é um conjunto com um número finito de elementos em que todas as operações de adição, subtração, multiplicação e divisão (exceto por zero) são possíveis. O mais simples dos corpos finitos é o conjunto $\mathbb{Z}_p$, o conjunto dos inteiros módulo $p$, onde $p$ é um número primo. Esse conjunto forma um corpo, pois todos os elementos não nulos possuem inverso multiplicativo módulo $p$ (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

Os corpos finitos são amplamente utilizados em criptografia por permitirem a definição de operações bem comportadas em conjuntos limitados, o que é essencial para o funcionamento de algoritmos como RSA, ElGamal e curvas elípticas. No contexto criptográfico, os anéis $\mathbb{Z}_n$ são de particular interesse, pois o algoritmo RSA opera essencialmente dentro da estrutura desse anel, definindo suas operações de maneira consistente. Além disso, os corpos finitos, um tipo mais específico de anel, são cruciais para a construção de diversos outros algoritmos criptográficos modernos, como os de curvas elípticas e alguns algoritmos simétricos, permitindo operações bem-definidas em conjuntos limitados.

### 1.2.3 Teorema Chinês do Resto

O Teorema Chinês do Resto (TCR) é um resultado clássico da Teoria dos Números que permite resolver sistemas de congruências simultâneas com módulos coprimos. Embora seu nome remeta a origens antigas, sua utilidade se mantém até hoje, sendo aplicada, por exemplo, em otimizações no processo de decriptação no algoritmo RSA.

De forma simplificada, o teorema afirma que, dado um sistema de congruências:

$$\begin{cases} x \equiv a_1 \pmod{n_1} \\ x \equiv a_2 \pmod{n_2} \\ \vdots \\ x \equiv a_k \pmod{n_k} \end{cases} \tag{1.5}$$

onde os módulos $n_1, n_2, \ldots, n_k$ são dois a dois coprimos, então existe uma solução única $x$ módulo $N = n_1 n_2 \cdots n_k$, e essa solução pode ser explicitamente construída (ROSEN, 2010).

No contexto da criptografia RSA, o TCR é utilizado para acelerar o processo de decriptação por meio de uma técnica chamada *CRT optimization*. Em vez de calcular $m \equiv c^d \pmod{n}$ diretamente, é possível calcular $m_p \equiv c^d \pmod{p}$ e $m_q \equiv c^d \pmod{q}$

separadamente e, em seguida, reconstruir $m \pmod{n}$ utilizando o TCR, onde $n = pq$. Como as exponenciações modulares sobre $p$ e $q$ são computacionalmente mais baratas que sobre $n$, essa técnica traz um ganho expressivo de desempenho, especialmente em dispositivos com recursos limitados (HOFFSTEIN; PIPHER; SILVERMAN, 2014; MENEZES; OORSCHOT; VANSTONE, 1996).

Além de sua aplicação prática, o TCR também evidencia a riqueza da aritmética modular e sua capacidade de construir soluções únicas a partir de restrições múltiplas, exatamente o tipo de abordagem que a criptografia moderna explora para garantir segurança e funcionalidade.

### 1.2.4 Corpos finitos e polinômios

Embora o algoritmo RSA não dependa diretamente de corpos finitos construídos via polinômios, vale mencionar que essa estrutura é amplamente utilizada em outros sistemas criptográficos, como AES, curvas elípticas e códigos corretores de erro. Conforme discutido por Masuda e Panario (MASUDA; PANARIO, 2007), os corpos finitos $\mathbb{F}_q$, especialmente aqueles construídos como quocientes de anéis de polinômios sobre $\mathbb{F}_p$, formam a base para diversas aplicações onde estruturas algébricas mais sofisticadas são requeridas.

Um corpo finito pode ser construído não apenas como $\mathbb{Z}_p$, mas também por meio de quocientes de anéis de polinômios. Por exemplo, seja $\mathbb{F}_2[x]$ o conjunto dos polinômios com coeficientes em $\mathbb{F}_2 = \{0, 1\}$. Podemos construir um corpo finito $\mathbb{F}_{2^3}$ como o conjunto de resíduos de $\mathbb{F}_2[x]$ módulo um polinômio irreducível de grau 3, como $x^3 + x + 1$:

$$\mathbb{F}_2[x] \longrightarrow \mathbb{F}_{2^3} = \mathbb{F}_2[x]/\langle x^3 + x + 1\rangle$$

Neste contexto, os elementos do corpo $\mathbb{F}_{2^3}$ são representados por polinômios de grau inferior a 3 com coeficientes em $\mathbb{F}_2$, ou seja, expressões da forma $a_0 + a_1x + a_2x^2$, com $a_i \in \{0, 1\}$. A aritmética é feita módulo 2 nos coeficientes e módulo $x^3 + x + 1$ nas multiplicações.

Embora esta construção via polinômios não seja diretamente empregada no funcionamento central do algoritmo RSA, ela é fundamental para outros sistemas criptográficos modernos, como o AES e a criptografia de curvas elípticas. Sua menção aqui visa demonstrar a amplitude e a sofisticação das estruturas algébricas que fundamentam grande parte da criptografia contemporânea (ROSEN, 2010; HOFFSTEIN; PIPHER; SILVERMAN, 2014; HANKERSON; MENEZES; VANSTONE, 2004).

## 1.3 Primalidade e Fatoração: visão teórica

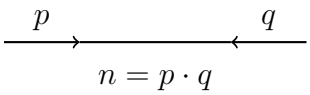
A segurança do algoritmo RSA está profundamente ligada à dificuldade de fatorar grandes números inteiros. Mais especificamente, ao construir uma chave pública, escolhem-se dois primos grandes $p$ e $q$, cujo produto $n = pq$ é divulgado publicamente. O segredo da chave privada está relacionado à fatoração de $n$, o que nos leva à importância de compreender tanto os números primos quanto os desafios associados à fatoração (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 1.3.1 Números primos e sua importância

Um número primo é um inteiro maior que 1 que possui apenas dois divisores: 1 e ele mesmo. A distribuição de primos é um tema central na Teoria dos Números, e, embora sua densidade diminua com o crescimento dos inteiros, os primos continuam infinitos, como demonstrado por Euclides há mais de dois mil anos (ROSEN, 2010). Além disso, propriedades como a indivisibilidade, a ausência de padrões regulares entre os primos, e sua aplicabilidade direta na construção de sistemas criptográficos são detalhadas de forma acessível por Lemos (LEMOS, 2010), que também discute sua importância na geração segura de chaves.

Na criptografia RSA, os primos são fundamentais porque sua multiplicação é fácil, mas sua fatoração é difícil. Esse "desequilíbrio de esforço computacional"é o que sustenta a segurança do sistema.

**Geração da chave RSA**




### 1.3.2 O problema da fatoração

Fatorar um número significa expressá-lo como produto de fatores primos. Embora existam algoritmos eficientes para verificar se um número é primo, fatorar um número grande, como os usados no RSA, continua sendo uma tarefa computacionalmente custosa. Não existe, até o momento, um algoritmo de tempo polinomial conhecido para fatoração de inteiros grandes (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

Diversos algoritmos têm sido desenvolvidos para essa finalidade, no entanto, à medida que o tamanho de $n$ aumenta (geralmente acima de 2048 bits), esses algoritmos se tornam inviáveis na prática, o que reforça a confiança no RSA.

### 1.3.3 Papel da teoria na prática criptográfica

A teoria dos primos e da fatoração não apenas sustenta o RSA, mas também influencia decisões práticas, como o tamanho ideal das chaves, os testes de primalidade aplicados na geração de $p$ e $q$, e os limites de segurança esperados. Esses fundamentos oferecem suporte direto às decisões práticas na implementação do RSA. (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

# 2 O Algoritmo RSA: Construção e Fundamentos

Com base nos princípios teóricos abordados no capítulo anterior, este capítulo introduz o algoritmo RSA, um dos sistemas de criptografia de chave pública mais conhecidos e utilizados na atualidade. Sua importância transcende o âmbito acadêmico, sendo amplamente empregado em ambientes reais, como conexões seguras na internet, sistemas bancários e protocolos de autenticação digital.

## 2.1 Introdução ao Algoritmo RSA

O algoritmo RSA foi apresentado em 1977 por Ron Rivest, Adi Shamir e Leonard Adleman, pesquisadores do MIT (Massachusetts Institute of Technology). Seu surgimento marcou uma mudança de paradigma na criptografia ao viabilizar a troca de mensagens seguras entre partes que nunca haviam se comunicado anteriormente, através de um par de chaves distintas: uma pública e uma privada (RIVEST; SHAMIR; ADLEMAN, 1978).

Essa proposta inverte a lógica dos métodos simétricos tradicionais, como o algoritmo de César, ao permitir que a cifragem de uma mensagem seja feita com uma chave pública amplamente conhecida, enquanto apenas a chave privada correspondente pode realizar a operação inversa, a decifragem.

A base matemática do RSA se ancora em propriedades fundamentais da Teoria dos Números, como a aritmética modular, os números primos e a função totiente de Euler (HOFFSTEIN; PIPHER; SILVERMAN, 2014). Esses elementos tornam possível a construção de um sistema assimétrico cuja segurança decorre da impossibilidade prática de se reverter determinadas operações, como será discutido em detalhes nas próximas seções.

Além de garantir confidencialidade, o RSA também permite a autenticação por meio de assinaturas digitais, ampliando sua aplicabilidade em sistemas modernos de segurança da informação. (STALLINGS, 2006).

Nas seções seguintes, serão detalhados os procedimentos de geração de chaves, cifragem, decifragem e a fundamentação matemática que assegura a correção e a robustez do algoritmo.

## 2.2 Geração de Chaves

O primeiro passo na utilização do algoritmo RSA consiste na geração do par de chaves: uma pública, utilizada para cifrar mensagens, e uma privada, usada para decifrá-las. Este processo depende de conceitos fundamentais da aritmética modular e da teoria dos números primos.

A geração de chaves no RSA segue as seguintes etapas:

1. Escolhem-se dois números primos distintos $p$ e $q$, grandes o suficiente para garantir a segurança do sistema. A escolha de primos longos e aleatórios é essencial, pois a dificuldade de fatorar seu produto é o que protege a chave privada.

2. Calcula-se $n = p \cdot q$. Esse valor $n$ compõe o módulo comum das operações de cifragem e decifragem, e será parte da chave pública.

3. Calcula-se a função totiente de Euler aplicada a $n$, ou seja, $\varphi(n) = (p-1)(q-1)$.

4. Escolhe-se um número inteiro $e$ tal que $1 < e < \varphi(n)$ e $\text{mdc}(e, \varphi(n)) = 1$ , ou seja, $e$ deve ser coprimo com $\varphi(n)$. Esse valor será o expoente da chave pública.

5. Calcula-se o inverso modular de $e$ módulo $\varphi(n)$, isto é, um inteiro $d$ tal que:

$$e \cdot d \equiv 1 \pmod{\varphi(n)}$$

   Esse valor $d$ será o expoente da chave privada.

A chave pública será então o par $(e, n)$ e a chave privada, o par $(d, n)$. A segurança do algoritmo reside no fato de que, mesmo conhecendo-se $e$ e $n$, é computacionalmente inviável determinar $d$ sem fatorar $n$ e obter os valores de $p$ e $q$ (HOFFSTEIN; PIPHER; SILVERMAN, 2014; LEMOS, 2010).

Na prática, algoritmos específicos são utilizados para gerar os primos $p$ e $q$ com rapidez e segurança, como os testes de primalidade probabilísticos. Os tamanhos das chaves, frequentemente com 2048 bits ou mais, garantem níveis adequados de proteção contra ataques baseados em força bruta ou fatoração direta. (MENEZES; OORSCHOT; VANSTONE, 1996; MORIARTY et al., 2016).

## 2.3 Processo de Criptografia e Decriptografia

Uma vez gerado o par de chaves, o algoritmo RSA permite a realização de dois processos fundamentais: cifragem (criptografia) e decifragem (decriptografia). Esses procedimentos são matematicamente simples, mas extremamente eficazes dentro da estrutura algébrica modular em que operam.

## Cifragem

Seja $m$ a mensagem a ser enviada, representada como um número inteiro tal que $0 < m < n$. A cifragem é feita utilizando a chave pública $(e, n)$, por meio da seguinte operação:

$$c \equiv m^e \pmod{n}$$

O valor $c$ representa o texto cifrado. O remetente realiza essa operação utilizando a chave pública do destinatário, o que garante que apenas o proprietário da chave privada correspondente poderá decifrar a mensagem.

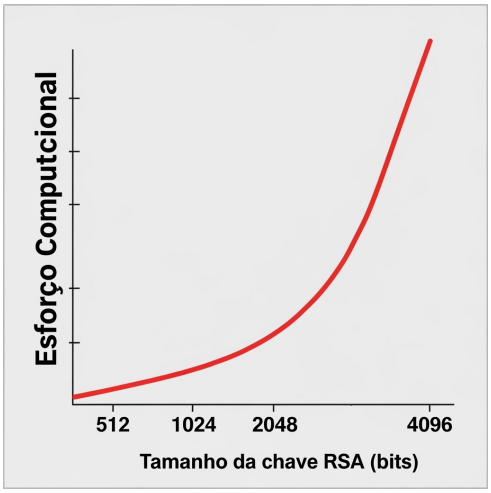
## Decifragem

Para recuperar a mensagem original $m$, o destinatário utiliza sua chave privada $(d, n)$ e realiza a operação inversa:

$$m \equiv c^d \pmod{n}$$

Esse resultado é garantido matematicamente pelos teoremas de Fermat e Euler, uma vez que $e$ e $d$ foram escolhidos de modo que $ed \equiv 1 \mod \varphi(n)$.

Figura 2.1 – Relação entre o tamanho da chave RSA (em bits) e o esforço computacional necessário para fatorar o número $n = pq$. O crescimento exponencial da complexidade torna o ataque inviável com chaves suficientemente grandes.




**Fonte:** Autor.

Esse mecanismo garante a confidencialidade da comunicação, pois mesmo que terceiros tenham acesso ao valor cifrado $c$ e ao valor público $n$, não conseguem obter $m$ sem conhecer a chave privada $d$, cuja obtenção exige a fatoração de $n$.

Além disso, o mesmo processo pode ser invertido para permitir assinaturas digitais: ao cifrar com sua própria chave privada, um usuário pode garantir a autoria da mensagem, que poderá ser validada por qualquer pessoa com acesso à sua chave pública (HOFFSTEIN; PIPHER; SILVERMAN, 2014; LEMOS, 2010).

## 2.4 Justificativa matemática da correção do algoritmo RSA

Uma das garantias fundamentais do algoritmo RSA é que, após a cifragem de uma mensagem $m$ utilizando a chave pública $(e, n)$, a aplicação da chave privada $d$ deve recuperar exatamente a mensagem original. Em termos matemáticos, isso significa que:

$$(m^e)^d \equiv m \pmod{n} \quad \Rightarrow \quad m^{ed} \equiv m \pmod{n}$$

Essa propriedade é assegurada por resultados da Teoria dos Números, em especial o Teorema de Euler. Como $e$ e $\varphi(n)$ são escolhidos de modo que $\text{mdc}(e, \varphi(n)) = 1$, existe um inverso multiplicativo $d$ tal que:

$$ed \equiv 1 \pmod{\varphi(n)}$$

Isso implica que existe um inteiro $k$ para o qual $ed = 1 + k\varphi(n)$. Substituindo na potência:

$$m^{ed} = m^{1+k\varphi(n)} = m \cdot (m^{\varphi(n)})^k$$

De acordo com o Teorema de Euler, se $\text{mdc}(m, n) = 1$, então $m^{\varphi(n)} \equiv 1 \pmod{n}$, o que nos leva a:

$$m^{ed} \equiv m \cdot 1^k \equiv m \pmod{n}$$

Assim, a operação de decriptação realmente recupera a mensagem original, validando matematicamente o funcionamento do RSA.

No caso em que $m$ não é coprimo com $n$, ou seja, $\text{mdc}(m, n) \neq 1$, o resultado ainda se mantém sob certos ajustes teóricos que envolvem o Teorema Chinês do Resto. Contudo, como na prática $m < n$ e $n = pq$ com $p$ e $q$ grandes, a chance de $m$ ser múltiplo de $p$ ou $q$ é desprezível. Portanto, essa hipótese é válida na maioria dos cenários criptográficos reais (HOFFSTEIN; PIPHER; SILVERMAN, 2014; MENEZES; OORSCHOT; VANSTONE, 1996).

## 2.5 Segurança do RSA

A segurança do algoritmo RSA está diretamente relacionada à dificuldade de fatorar um número inteiro $n = pq$, onde $p$ e $q$ são primos grandes. Embora a multiplicação de dois primos seja uma operação simples, a operação inversa, fatorar o número resultante, é computacionalmente difícil quando os primos são suficientemente grandes.

Dado que a chave pública inclui apenas os valores $(e, n)$, um atacante que deseje obter a chave privada $d$ precisaria, em teoria, calcular a função totiente $\varphi(n)$ para então encontrar o inverso modular de $e$. No entanto, sem conhecer os valores de $p$ e $q$, esse cálculo é inviável na prática, pois exige a fatoração de $n$. Essa assimetria é o que torna o RSA seguro em ambientes clássicos.

Diversos ataques ao RSA já foram estudados, sendo alguns deles baseados em:

- **Expoentes pequenos:** quando $e$ é pequeno (como $e = 3$), certas mensagens podem ser decifradas com ataques simples caso não sejam devidamente preenchidas com redundância ou técnicas de preenchimento seguro.
- **Fatoração eficiente:** algoritmos como o de Fermat, de Pollard-$\rho$, e o crivo generalizado (GNFS) podem ser usados para fatorar $n$, mas todos têm complexidade alta para valores grandes (acima de 2048 bits).
- **Ataques por canal lateral:** como cronometragem de operações ou análise de consumo de energia durante a execução do algoritmo.

Apesar desses ataques, o RSA continua seguro desde que os parâmetros sejam escolhidos corretamente: $n$ suficientemente grande (2048 bits ou mais), $e$ coprimo com $\varphi(n)$ e com valor adequado (geralmente $e = 65537$), e mecanismos de preenchimento como OAEP sejam utilizados para proteger contra ataques estruturais.

Vale ressaltar que essa segurança é relativa ao paradigma computacional clássico. Em um cenário com computadores quânticos suficientemente poderosos, o RSA estaria vulnerável ao algoritmo de Shor, que pode fatorar inteiros em tempo polinomial (SHOR, 1997). O impacto desse cenário é reforçado nas considerações finais.

Para um panorama técnico detalhado sobre os parâmetros seguros e os tipos de ataques conhecidos, ver Menezes et al. (MENEZES; OORSCHOT; VANSTONE, 1996).

## 2.6 Computação quântica e implicações futuras

Embora o RSA seja considerado seguro no modelo de computação clássico, esse cenário pode mudar radicalmente com o avanço da computação quântica. Em particular, o algoritmo de Shor, proposto por Peter Shor em 1994, demonstrou que é possível fatorar inteiros grandes em tempo polinomial utilizando um computador quântico (SHOR, 1997).

Como a segurança do RSA depende da dificuldade de fatorar o número $n = pq$, o algoritmo de Shor representa uma ameaça direta. Se computadores quânticos em larga escala se tornarem viáveis, eles poderão quebrar o RSA ao calcular $\varphi(n)$ com base na fatoração eficiente de $n$, comprometendo assim a geração da chave privada.

Atualmente, ainda não existem computadores quânticos capazes de executar o algoritmo de Shor em instâncias de RSA com tamanho real (como chaves de 2048 bits), mas os avanços na área têm motivado a pesquisa em criptografia pós-quântica, sistemas que se mantêm seguros mesmo diante de adversários com capacidade quântica.

Segundo Bernstein e Lange (BERNSTEIN; LANGE, 2017), embora máquinas quânticas capazes de ameaçar o RSA ainda não existam, os riscos são altos o bastante para justificar a transição antecipada a algoritmos resistentes a ataques quânticos. Diversas iniciativas de padronização e testes já estão em andamento para garantir segurança mesmo em um futuro cenário com computadores quânticos operacionais.

A análise completa do algoritmo de Shor e das propostas de criptografia pós-quântica foge ao escopo deste trabalho, mas sua relevância como ameaça potencial aos sistemas clássicos, como o RSA, é inquestionável.

# 3 Aplicações Práticas do RSA

Com os fundamentos teóricos consolidados, este capítulo foca nas aplicações do RSA. Viu-se como propriedades da aritmética modular, da função totiente de Euler e da teoria da fatoração fundamentam a construção segura desse sistema criptográfico. Serão apresentadas uma implementação didática com valores reduzidos, os padrões técnicos associados e os principais contextos de uso, como conexões seguras na internet e assinaturas digitais. Por fim, discutem-se suas limitações e comparações com algoritmos criptográficos modernos, como os baseados em curvas elípticas. Este capítulo conecta a teoria desenvolvida até aqui com aplicações concretas e contemporâneas no campo da segurança da informação.

## 3.1 Implementação simplificada do RSA

Para ilustrar na prática o funcionamento do algoritmo RSA, apresentamos a seguir um exemplo numérico com valores reduzidos. Embora tais valores não ofereçam segurança criptográfica real, eles são suficientes para compreender a lógica de geração de chaves, cifragem e decifragem.

### Escolha dos primos

Selecionamos dois primos pequenos:

$$p = 11, \quad q = 13$$

Calculamos o módulo $n$ e a função totiente de Euler $\varphi(n)$:

$$n = p \cdot q = 11 \cdot 13 = 143, \quad \varphi(n) = (p-1)(q-1) = 10 \cdot 12 = 120$$

### Escolha do expoente público $e$

Escolhemos um número $e \in \mathbb{N}$ tal que $1 < e < \varphi(n)$ e $\text{mdc}(e, \varphi(n)) = 1$. Vamos adotar:

$$e = 7, \quad \text{mdc}(7, 120) = 1$$

### Cálculo do expoente secreto $d$

Buscamos o inverso modular de $e$ módulo $\varphi(n)$:

$$d \equiv e^{-1} \pmod{\varphi(n)} \Rightarrow d \equiv 7^{-1} \pmod{120}$$

Usando o Algoritmo de Euclides Estendido, obtemos:

$$d = 103, \quad \text{pois } 7 \cdot 103 = 721 \equiv 1 \pmod{120}$$

## Chaves geradas

- **Chave pública:** $(e, n) = (7, 143)$
- **Chave privada:** $(d, n) = (103, 143)$

## Cifragem

Suponha que a mensagem $m = 9$ será cifrada usando a chave pública:

$$c \equiv m^e \pmod{n} \Rightarrow c \equiv 9^7 \pmod{143}$$

Calculando:

$$9^7 = 4782969 \Rightarrow 4782969 \mod 143 = 48$$

Logo, a mensagem cifrada é:

$$c = 48$$

## Decifragem

Para recuperar $m$, aplicamos a chave privada:

$$m \equiv c^d \pmod{n} \Rightarrow m \equiv 48^{103} \pmod{143}$$

Esse valor pode ser calculado de forma eficiente com exponenciação modular. O resultado é:

$$m = 9$$

Portanto, a mensagem original foi corretamente recuperada, comprovando o funcionamento correto do RSA, mesmo em um exemplo simplificado (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

## 3.2 Padrões e protocolos baseados em RSA

O algoritmo RSA, embora conceitualmente simples, raramente é utilizado em sua forma "pura" nos sistemas modernos. Em aplicações reais, ele é implementado dentro de padrões criptográficos bem definidos, que adicionam camadas de segurança e compatibilidade. Entre os mais relevantes estão o PKCS #1 e os esquemas de preenchimento como o OAEP.

### 3.2.1 PKCS #1

O padrão PKCS #1 (Public-Key Cryptography Standards), desenvolvido pela RSA Laboratories, define a sintaxe e os procedimentos para implementar o RSA com segurança. Ele especifica:

- O formato das chaves pública e privada.
- Métodos de preenchimento (padding) para evitar ataques a mensagens previsíveis.
- Procedimentos para assinatura digital e verificação.

Esse padrão é amplamente utilizado em bibliotecas criptográficas como OpenSSL e em protocolos como TLS/SSL (MORIARTY et al., 2016).

### 3.2.2 Preenchimento seguro com OAEP

Um dos riscos do uso direto do RSA é que, se a mensagem cifrada tiver estrutura previsível, um atacante pode explorar essa previsibilidade. Para mitigar esse risco, utiliza-se um esquema de preenchimento (padding), e o mais comum é o OAEP (Optimal Asymmetric Encryption Padding).

O OAEP adiciona aleatoriedade à mensagem antes da cifragem, dificultando ataques por análise de padrões e garantindo que uma mesma mensagem cifrada mais de uma vez produza resultados diferentes (MORIARTY et al., 2016).

### 3.2.3 Importância prática

Esses padrões tornam o RSA viável em ambientes reais, permitindo que sistemas distintos se comuniquem de forma segura e padronizada. Sem eles, o uso direto do RSA seria vulnerável a ataques conhecidos, como ataque de texto plano escolhido com expoente pequeno.

Conforme documentado pela IETF no RFC 8017 (MORIARTY et al., 2016), o PKCS #1 v2.2 fornece uma especificação completa das primitivas criptográficas, esquemas

de assinatura, esquemas de cifragem e codificação baseados em RSA, incluindo o uso formal do OAEP e suas garantias de segurança.

## 3.3 Aplicações reais do RSA

O algoritmo RSA é amplamente utilizado em sistemas modernos de segurança da informação. Embora raramente operando isoladamente, ele atua como componente central em diversos protocolos e infraestruturas, sobretudo na proteção de dados durante a transmissão, autenticação de identidades e assinatura digital.

### 3.3.1 Conexões seguras (HTTPS/TLS)

Uma das aplicações mais comuns do RSA está nos protocolos HTTPS e TLS, que garantem a comunicação segura entre navegadores e servidores. Durante o processo de estabelecimento da conexão (handshake), o servidor envia seu certificado digital contendo a chave pública RSA. O cliente, ao verificar a autenticidade do certificado, utiliza essa chave para negociar uma chave simétrica temporária, protegendo os dados da sessão (MENEZES; OORSCHOT; VANSTONE, 1996; MORIARTY et al., 2016).

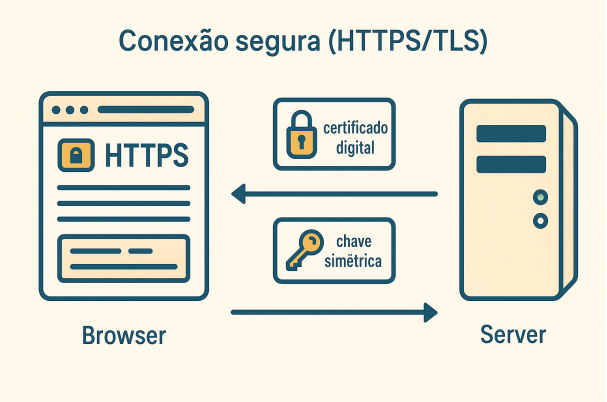



Figura 3.1 – Representação simplificada do processo de estabelecimento de conexão segura via HTTPS/TLS utilizando RSA. O cliente (browser) obtém o certificado digital do servidor, contendo sua chave pública RSA, e o utiliza para negociar uma chave simétrica temporária, usada para proteger os dados da sessão.

**Fonte:** Autor.

### 3.3.2 Assinaturas digitais e certificação

O RSA também é utilizado para garantir a integridade e a autenticidade de documentos por meio de assinaturas digitais. Nesse cenário, a chave privada do emissor é usada para assinar uma mensagem ou documento, e qualquer receptor pode verificar a assinatura com a respectiva chave pública (RIVEST; SHAMIR; ADLEMAN, 1978).

Infraestruturas como a ICP-Brasil utilizam o RSA para emissão e validação de certificados digitais, que permitem o uso de assinaturas eletrônicas em ambientes como o

sistema judiciário (PJe [Processo Judicial Eletrônico]), serviços públicos (gov.br), bancos e outras entidades que exigem autenticação forte (TECNOLOGIA DA INFORMAÇÃO (ITI), 2025).

### 3.3.3 Autenticação em sistemas bancários e tokens físicos

Dispositivos de autenticação, como tokens bancários e smartcards, também empregam o RSA para gerar códigos ou validar transações. Além disso, sistemas bancários utilizam o RSA em canais internos e externos para proteger credenciais e informações sensíveis durante o transporte entre servidores e aplicações (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 3.3.4 Mensageria segura e e-mail criptografado

Protocolos como o S/MIME e ferramentas como o GPG utilizam RSA para garantir que mensagens só possam ser lidas pelo destinatário correto. Isso é feito cifrando a chave de sessão com a chave pública do receptor, que a decifra com sua chave privada (STALLINGS, 2006).

### 3.3.5 Interoperabilidade

Uma das vantagens práticas do RSA é sua ampla aceitação e padronização. Por estar presente em bibliotecas como OpenSSL, Java Cryptography Architecture (JCA) e ambientes como .NET e Android, o RSA possibilita interoperabilidade entre diferentes plataformas, o que contribui para sua adoção contínua, mesmo diante de alternativas mais modernas (MENEZES; OORSCHOT; VANSTONE, 1996; STALLINGS, 2006).

## 3.4 Comparação com outros algoritmos criptográficos

Embora o RSA seja um dos algoritmos mais conhecidos e amplamente utilizados na criptografia assimétrica, ele não é o único disponível, e em muitos casos, não é o mais eficiente. Outros sistemas, como os baseados em curvas elípticas, vêm ganhando espaço por oferecerem níveis comparáveis de segurança com tamanhos de chave significativamente menores.

### 3.4.1 RSA vs. ECC (Curvas Elípticas)

O principal concorrente moderno do RSA em ambientes práticos é a criptografia de curva elíptica (ECC – Elliptic Curve Cryptography). A principal vantagem da ECC está na eficiência: para alcançar um mesmo nível de segurança, as chaves usadas na ECC podem

ser até oito vezes menores que as do RSA (HANKERSON; MENEZES; VANSTONE, 2004; MENEZES; OORSCHOT; VANSTONE, 1996).

Por exemplo, enquanto o RSA exige chaves de 2048 bits para alcançar um nível de segurança considerado adequado atualmente, a ECC atinge esse mesmo patamar com apenas 256 bits. Essa eficiência se traduz em diversos ganhos práticos: menor uso de banda e energia, cifragem e decifragem mais rápidas, e a possibilidade de implementação em dispositivos com recursos computacionais limitados. Isso faz com que a ECC seja especialmente vantajosa em ambientes embarcados, aplicações móveis, autenticação de documentos digitais e sistemas onde o armazenamento e a velocidade são fatores críticos. A equivalência de segurança entre ambos os métodos, apesar da diferença drástica no tamanho das chaves, é um dos principais argumentos a favor da adoção crescente da criptografia baseada em curvas elípticas.

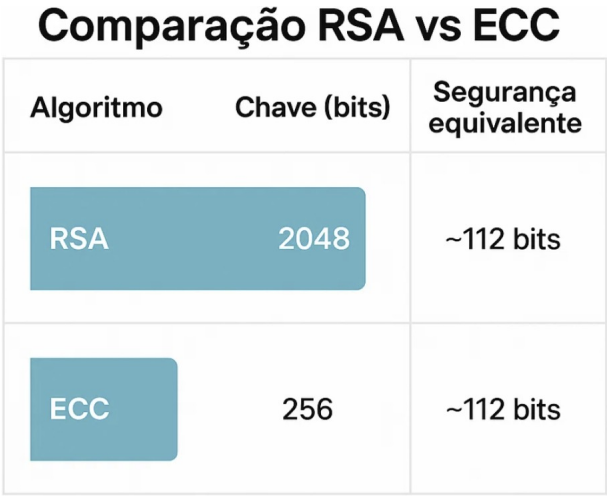



Figura 3.2 – Comparação entre o RSA e o ECC com relação ao tamanho da chave e segurança equivalente.

Figura 3.3 – **Fonte:** Autor.

### 3.4.2 Segurança e maturidade

O RSA é um algoritmo bem estudado, com décadas de uso e padronização, o que o torna confiável em muitos contextos. No entanto, sua segurança depende da dificuldade da fatoração de inteiros grandes, um problema que, embora ainda considerado difícil, pode ser eventualmente superado com avanços na computação quântica, tema já discutido no contexto de ameaças futuras ao RSA. (SHOR, 1997; BERNSTEIN; LANGE, 2017).

Por outro lado, a segurança da ECC baseia-se no problema do logaritmo discreto em curvas elípticas, que também é considerado resistente no modelo clássico, mas possui diferentes vulnerabilidades em cenários pós-quânticos.

### 3.4.3 Adoção e interoperabilidade

O RSA continua sendo amplamente adotado em protocolos tradicionais como TLS, S/MIME e OpenPGP, além de estar presente em infraestruturas como a ICP-Brasil (TECNOLOGIA DA INFORMAÇÃO (ITI), 2025). A ECC, por sua vez, está ganhando espaço em sistemas mais modernos e de alta performance, incluindo o uso em dispositivos móveis e protocolos como o Signal, SSH com ED25519 e bibliotecas modernas como libsodium.

### 3.4.4 Resumo comparativo

Ambos os algoritmos possuem méritos e limitações, e sua escolha depende do contexto. Em ambientes onde desempenho e tamanho de chave são críticos, a ECC tende a ser preferida. Já em sistemas legados e de ampla compatibilidade, o RSA ainda é uma escolha sólida.

## 3.5 Limitações práticas do RSA

Embora o RSA seja um dos algoritmos mais conhecidos e amplamente utilizados na criptografia moderna, ele apresenta limitações importantes que afetam sua eficiência e aplicabilidade em determinados contextos.

### 3.5.1 Desempenho e tamanho de chave

Uma das principais limitações do RSA está relacionada ao tamanho necessário das chaves para garantir níveis adequados de segurança. Por exemplo, enquanto algoritmos baseados em curvas elípticas (ECC) oferecem segurança robusta com chaves de 256 bits, o RSA exige chaves de pelo menos 2048 bits para garantir uma proteção equivalente. Isso impacta diretamente a performance, especialmente em dispositivos com recursos computacionais limitados, como smartcards e sistemas embarcados (MENEZES; OORSCHOT; VANSTONE, 1996; HANKERSON; MENEZES; VANSTONE, 2004).

Além disso, as operações de criptografia e decriptação com RSA são relativamente lentas quando comparadas a sistemas simétricos ou a criptografia baseada em curvas elípticas, o que o torna pouco eficiente para transmissão contínua de dados em tempo real (HOFFSTEIN; PIPHER; SILVERMAN, 2014).

### 3.5.2 Não adequado para grandes volumes de dados

O RSA não é ideal para criptografar grandes quantidades de informação diretamente. Devido à sua estrutura matemática, o algoritmo opera sobre blocos de dados cujo tamanho é limitado pelo comprimento da chave. Isso significa que, para proteger grandes volumes

de dados, o RSA geralmente é combinado com algoritmos de criptografia simétrica, como o AES, sendo utilizado apenas para proteger a chave de sessão simétrica, prática comum em protocolos como o TLS (STALLINGS, 2006).

### 3.5.3 Dependência de preenchimento seguro

Outra limitação crítica do RSA está na sua vulnerabilidade a ataques caso o algoritmo seja utilizado sem um esquema de preenchimento (padding) adequado. Mensagens com estrutura previsível, se cifradas diretamente, podem ser alvos de ataques de texto escolhido ou outras formas de análise. Por isso, o uso de esquemas como o OAEP é essencial para a segurança do RSA, conforme especificado no padrão PKCS #1 (MORIARTY et al., 2016).

### 3.5.4 Sensibilidade à implementação

Por fim, a segurança do RSA depende fortemente da forma como é implementado. Ataques por canal lateral, como análise de tempo de execução ou consumo de energia, podem comprometer a chave privada mesmo sem quebrar a criptografia em si. Isso exige o uso de práticas rigorosas de engenharia de software e hardware seguro para evitar vazamentos (MENEZES; OORSCHOT; VANSTONE, 1996). Apesar dessas limitações, o RSA continua sendo uma peça fundamental da criptografia moderna, especialmente quando utilizado em conjunto com boas práticas e algoritmos complementares. Essa relevância será retomada nas considerações finais.

# 4 Considerações finais

Este trabalho teve como objetivo principal apresentar os fundamentos matemáticos do algoritmo RSA, destacando como estruturas clássicas da Teoria dos Números sustentam um dos pilares da criptografia moderna. A abordagem percorreu desde os conceitos básicos de congruência, função totiente de Euler e primalidade até a formalização do algoritmo, suas aplicações práticas e limitações.

Ao longo dos capítulos, foi possível compreender como a aritmética modular, os teoremas de Fermat e Euler, e a dificuldade computacional de certos problemas (como a fatoração de inteiros grandes) formam a base da segurança do RSA. A estrutura assimétrica do algoritmo, aliada à possibilidade de autenticação via assinaturas digitais, o consolidou como um dos sistemas criptográficos mais difundidos no mundo.

Também foram evidenciadas as limitações do RSA, como seu desempenho inferior em dispositivos com restrição de recursos, a necessidade de chaves maiores para garantir segurança e a obrigatoriedade de utilizar preenchimentos seguros (como o OAEP) para evitar ataques estruturais. Tais restrições tornam o algoritmo menos adequado para aplicações que exigem alta eficiência ou mobilidade.

Além disso, discutiu-se o impacto potencial da computação quântica sobre algoritmos clássicos como o RSA. O algoritmo de Shor, embora ainda impraticável em larga escala, representa uma ameaça concreta ao modelo de segurança atual. Esse cenário tem impulsionado o desenvolvimento de soluções pós-quânticas, que buscam garantir a continuidade da confidencialidade e da autenticidade dos dados diante de novas capacidades computacionais.

Conclui-se, portanto, que a matemática não apenas fundamenta os sistemas criptográficos, como também orienta sua evolução e define seus limites. O estudo do RSA proporciona uma compreensão técnica de seu funcionamento e promove uma reflexão crítica sobre os desafios contínuos da segurança da informação em um mundo digital em constante transformação.

# Bibliografia

BERNSTEIN, Daniel J.; LANGE, Tanja. Post-quantum cryptography. **Nature**, v. 549, n. 7671, p. 188–194, 2017. DOI: 10.1038/nature23461. Citado nas pp. 27, 33.

HANKERSON, Darrel; MENEZES, Alfred; VANSTONE, Scott. **Guide to Elliptic Curve Cryptography**. New York: Springer, 2004. Citado nas pp. 19, 33, 34.

HOFFSTEIN, Jeffrey; PIPHER, Jill; SILVERMAN, Joseph H. **An Introduction to Mathematical Cryptography**. 2. ed. New York: Springer, 2014. Citado nas pp. 12–25, 29, 32, 34.

LEMOS, Manoel. **Criptografia, Números Primos e Algoritmos**. 4. ed. Rio de Janeiro: IMPA, 2010. Citado nas pp. 20, 23, 24.

MASUDA, Ariane M.; PANARIO, Daniel. **Tópicos de Corpos Finitos com Aplicações em Criptografia e Teoria de Códigos**. Rio de Janeiro: IMPA, 2007. Citado na p. 19.

MENEZES, Alfred; OORSCHOT, Paul van; VANSTONE, Scott. **Handbook of Applied Cryptography**. Boca Raton: CRC Press, 1996. Citado nas pp. 16, 17, 19, 23, 25, 26, 31–35.

MORIARTY, Kathleen et al. **PKCS #1: RSA Cryptography Specifications Version 2.2**. [S.l.: s.n.], 2016. https://tools.ietf.org/html/rfc8017. RFC 8017, November 2016. Citado nas pp. 23, 30, 31, 35.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. **Communications of the ACM**, v. 21, n. 2, p. 120–126, 1978. DOI: 10.1145/359340.359342. Citado nas pp. 22, 31.

ROSEN, Kenneth H. **Elementary Number Theory and Its Applications**. 6. ed. Boston: Pearson, 2010. Citado nas pp. 13, 17–20.

SHOR, Peter W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. **SIAM Journal on Computing**, v. 26, n. 5, p. 1484–1509, 1997. DOI: 10.1137/S0097539795293172. Citado nas pp. 26, 33.

STALLINGS, William. **Criptografia e segurança de redes: princípios e práticas**. 4. ed. São Paulo: Pearson Prentice Hall, 2006. Citado nas pp. 22, 32, 35.

TECNOLOGIA DA INFORMAÇÃO (ITI), Instituto Nacional de. **Estrutura detalhada da Infraestrutura de Chaves Públicas Brasileira — ICP-Brasil**. [S.l.: s.n.], 2025. Disponível em: https://www.gov.br/iti/pt-br/assuntos/icp-brasil/estrutura-detalhada-icp-brasil. Citado nas pp. 32, 34.