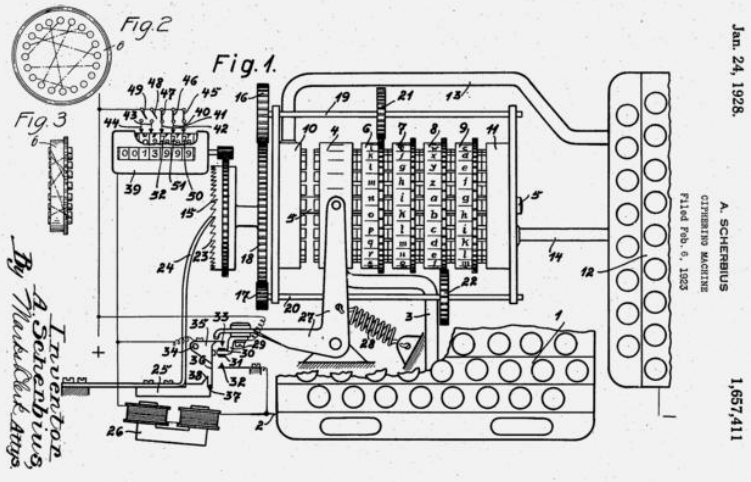
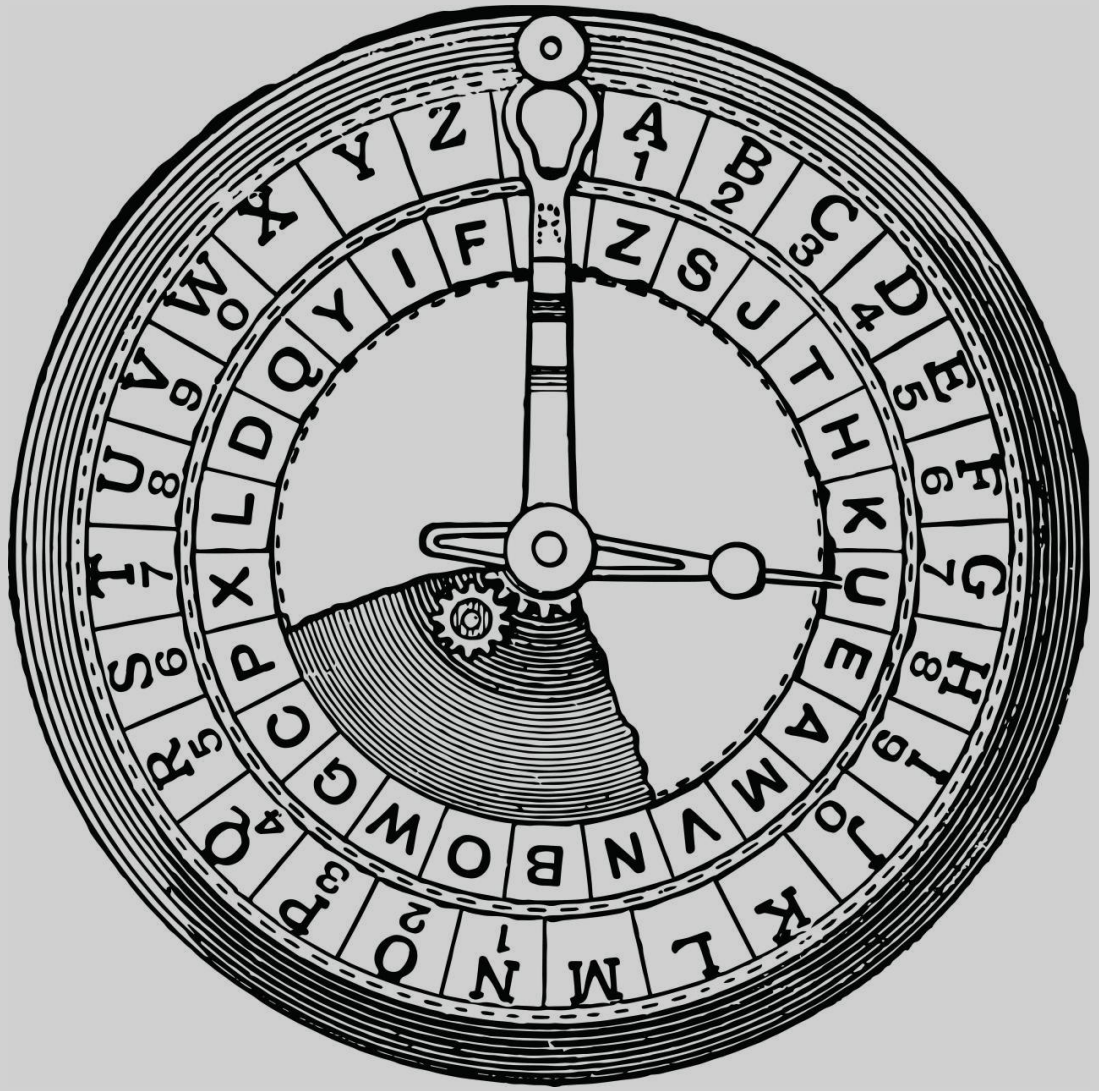
Cifra RSA

Fundamentos da Criptografia

Nicollas Mocelin Sdroievski



Relembrando



Definição

- **Definição 8.9.** Um *grupo* é o um conjunto $\mathbb{G}$ junto de uma operação binária $\circ$ para os quais o seguinte é verdade:
  - **(Fechamento).** Para todos $g, h \in \mathbb{G}$, $g \circ h \in \mathbb{G}$.
  - **(Identidade).** Existe um elemento *identidade* $e \in \mathbb{G}$ tal que, para todo $g \in \mathbb{G}$, $e \circ g = g = g \circ e$.
  - **(Inversos).** Para todo $g \in \mathbb{G}$, existe um elemento $h \in \mathbb{G}$ tal que $g \circ h = e = h \circ g$. Este elemento h é chamado de *inverso* de g.
  - **(Associatividade)**. Para todos $g_1, g_2, g_3 \in \mathbb{G}$, $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$
- Vamos lidar com grupos finitos e abelianos, que obedecem a seguinte propriedade adicional
  - **(Comutatividade).** Para todos $g, h \in \mathbb{G}$, $g \circ h = h \circ g$.
- A ordem de um grupo finito $\mathbb{G}$, denotada por $|\mathbb{G}|$, é o número de elementos de $\mathbb{G}$

Fatoração e divisão

- **Teorema fundamental da aritmética.** Todo inteiro maior que 1 pode ser representado de maneira única (exceto pela ordem dos fatores) como um produto de números primos

$$N = \prod_i p_i^{e_i}$$

Onde cada p_i é um número primo distino e $e_i \geq 1$ para todo i

- **Proposição 8.1.** Seja a um inteiro e b um inteiro positivo. Então existem inteiros únicos q, r tal que $a = qb + r$ e $0 \leq r < b$.

MDC (GCD) e companhia

- O *maior divisor comum* (MDC ou GCD) de dois inteiros a e b, denotado por $\gcd(a, b)$ é o maior inteiro c tal que $c \mid a$ e $c \mid b$
  - Se $\gcd(a, b) = 1$, dizemos que a e b são relativamente primos

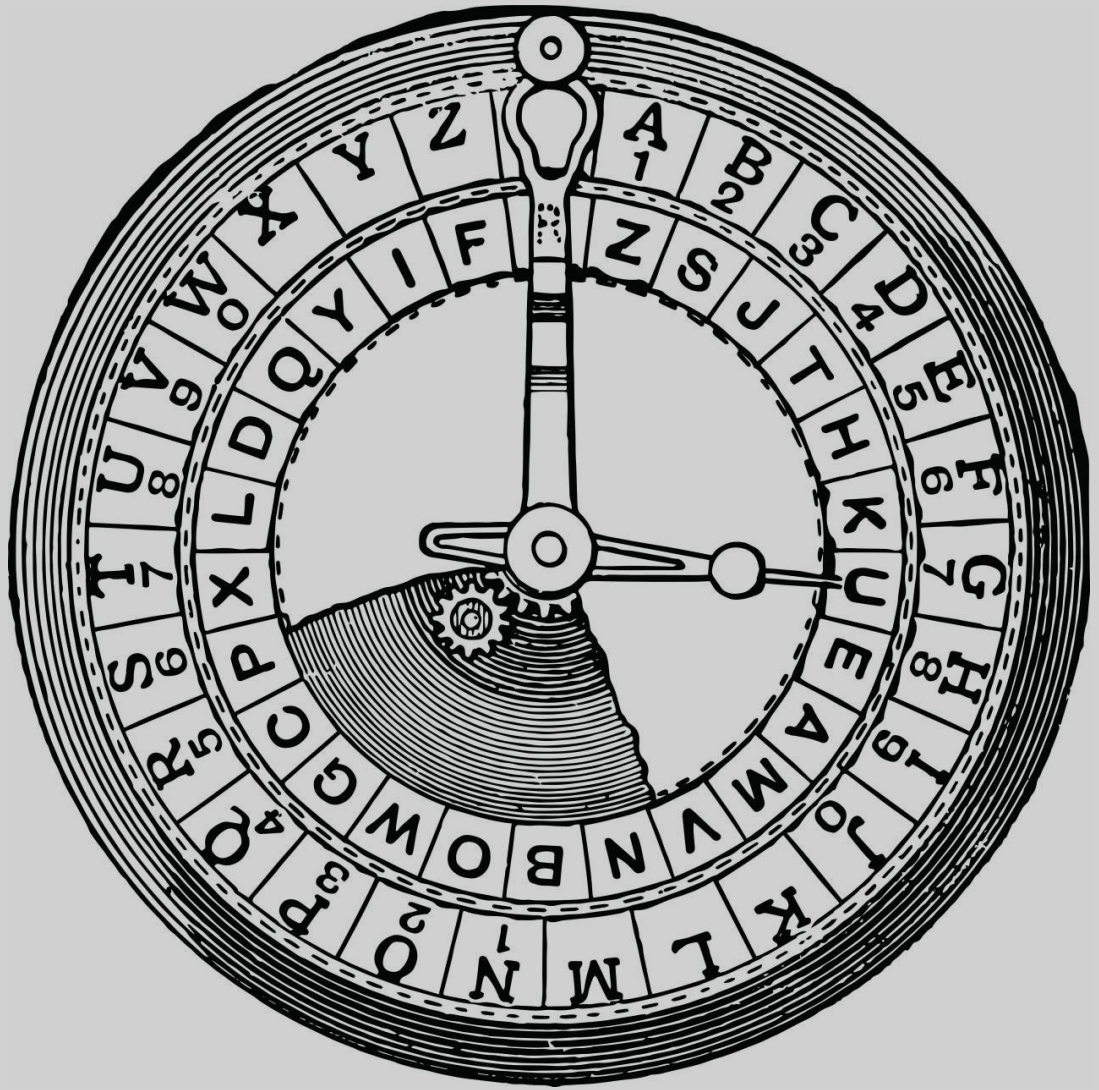
- **Proposição 8.2.** Sejam a e b inteiros positivos. Então existem $X, Y \in \mathbb{Z}$ tal que $Xa + Yb = \gcd(a, b)$. Além disso, $\gcd(a, b)$ é o menor inteiro positivo que pode ser escrito desta maneira.

Divisão modular

- Se para um inteiro b, existe um inteiro c tal que $bc = 1 \bmod N$, dizemos que b é invertível módulo N
  - E que c é um inverso (multiplicativo) de b módulo N
  - Se c é um inverso de b módulo N, o mesmo vale para $[c \bmod N]$
  - Se b é invertível, existe um único inverso de b no intervalo $\{1, \dots, N-1\}$

- **Proposição 8.7.** Sejam b, N inteiros com $b \geq 1$ e $N > 1$. Então b é invertível módulo N se e somente se $\gcd(b, N) = 1$
  - Além disso, é possível computar inversos de maneira eficiente (usando o algoritmo estendido de Euclides)

O grupo $\mathbb{Z}_N^*$



Definição

- **Lembrando:** um elemento $x \in \mathbb{Z}_N$ possui inverso multiplicativo se e somente se $\gcd(x, N) = 1$ (ou seja, se é coprimo a N)
- **Proposição 8.18.** $\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N \mid \gcd(x, N) = 1\}$ é um grupo abeliano para qualquer $N > 1$.
- **Demonstração:**
  - **Identidade.** Sempre vale que $1 \in \mathbb{Z}_N^*$, para qualquer $N > 1$
- **Inverso.** O lembrete acima garante que todo elemento $x \in \mathbb{Z}_N^*$ possui inverso multiplicativo.
  - **Associatividade e Comutatividade** são fáceis de verificar.
  - **Fechamento.** Sejam $x, y \in \mathbb{Z}_N^*$, então xy admite o inverso $y^{-1} \cdot x^{-1}$, o que implica que $\gcd(xy, N) = 1$.

Tamanho de $\mathbb{Z}_N^*$

- $\phi(N) \doteq |\mathbb{Z}_N^*|$ (a função phi ou totiente de Euler)
- Sabemos que para p primo, $\phi(p) = p - 1$
  - Todo elemento do conjunto $\{1,2, \dots, p-1\}$ é coprimo a p
- E $\mathbb{Z}_N^*$ para N composto?
  - Para nossos objetivos, $N = pq$ para primos p e q
  - Neste caso $\phi(N) = (p-1)(q-1)$
  - Se $a \in \{1, \dots, N-1\}$ não é coprimo a p ou q, então $p \mid a$ ou $q \mid a$
  - Divisíveis por p: $p, 2p, 3p, \dots, (q-1)p$
  - Divisíveis por q: $q, 2q, 3q, \dots, (p-1)q$
  - Elementos que sobraram
    $(N-1) - (q-1) - (p-1) = pq - p - q + 1 = (p-1)(q-1)$

Exemplo

- $N = 15 = 5 \cdot 3$
- $\mathbb{Z}_N^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$
- $|\mathbb{Z}_{15}^*| = 8 = 4 \cdot 2 = \phi(15)$
- O inverso de 8 em $\mathbb{Z}_{15}^*$ é 2, pois $2 \cdot 8 = 16 = 1 \bmod 15$
- Como $\gcd(7,8) = 1$, é possível verificar que
  - $\{1^7, 2^7, 4^7, 7^7, 8^7, 11^7, 13^7, 14^7\} = \{1, 8, 4, 13, 2, 11, 7, 14\} \bmod 15$
  - Veremos por que no próximo slide

Alguns resultados

- Lembrando que, para $m = |\mathbb{G}|$ e $g \in \mathbb{G}$, sempre vale que $g^m = 1$
- **Corolário 8.21.** Seja $N > 1$ e $a \in \mathbb{Z}_N^*$. Então

$$a^{\phi(N)} = 1 \bmod N$$

Corolário 8.22. Sejam $N > 1$ e $e > 0$. Defina $f_e : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ como $f_e(x) = [x^e \bmod N]$. Se e é coprimo a $\phi(N)$, então f_e é uma permutação, e f_d, para $d = e^{-1} \bmod \phi(N)$, é seu inverso.

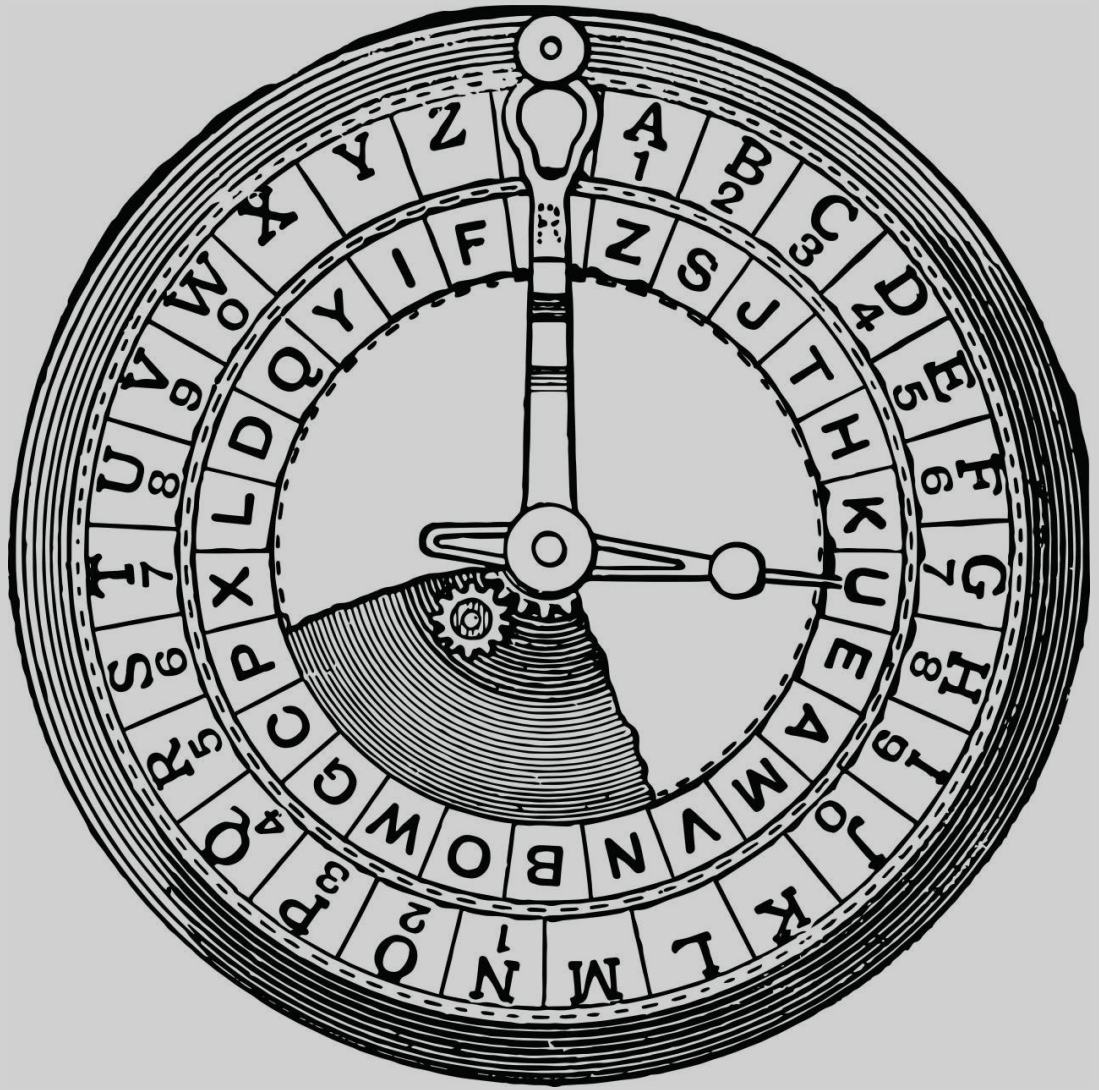
Demonstração: Basta mostrar que f_d como no enunciado é a função inversa de f_e

$$f_d\big(f_e(x)\big) = [x^{ed} \bmod N] = \left[x^{ed \bmod \phi(N)} \bmod N\right] = x$$

Algoritmos

- Existem algoritmos eficientes (polinomiais) para:
  - Gerar um número primo p dentro de um intervalo
    - Com probabilidade baixa de falha
  - Testar se um número é primo
    - Na prática, com probabilidade baixa de falha
  - Amostrar elementos aleatórios de $\mathbb{Z}_N^*$, dado N
    - Com probabilidade baixa de falha
  - Computar inversos em $\mathbb{Z}_N^*$ (usando o algoritmo estendido de Euclides)
  - Computar $[x^e \bmod N]$

Fatoração e o problema RSA



Fatoração é fácil

- Imagine o seguinte experimento

Desafiante



$x, y \leftarrow$ inteiros aleatórios com n bits

$N := x_1 \cdot x_2$

N

x_1', x_2'

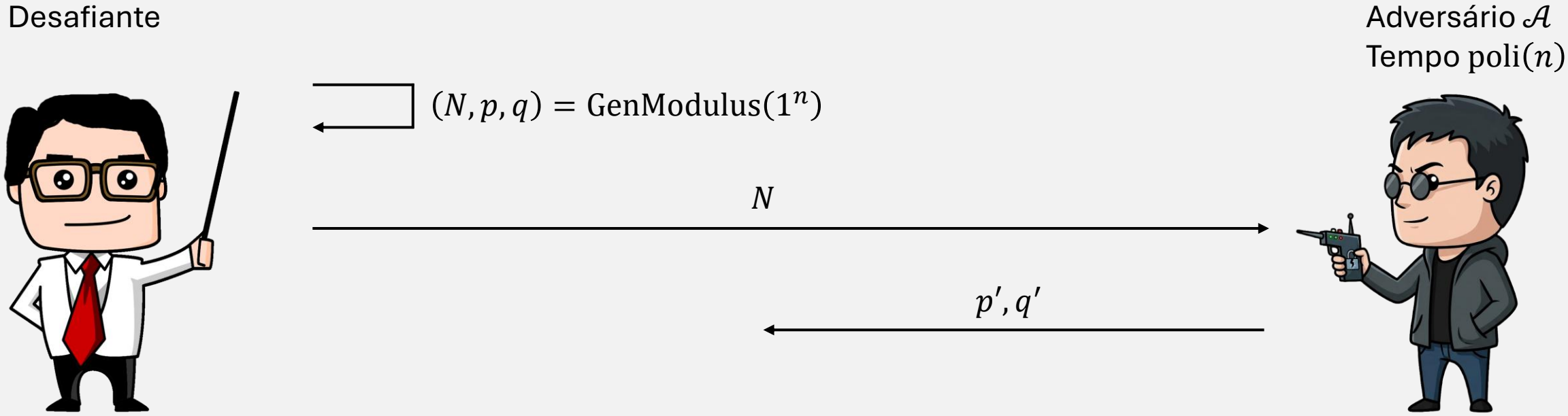
Adversário $\mathcal{A}$
Tempo poli(n)



- $\mathcal{A}$ vence se $x_1' \cdot x_2' = N$
- Existe um adversário polinomial $\mathcal{A}$ que vence com probabilidade $\geq 3/4$

Fatoração é difícil: $\text{Factor}_{\mathcal{A},\text{GenModulus}}(n)$

- Seja GenModulus um algoritmo que, na entrada 1^n, produz (N, p, q), tal que $N = pq$ e p, q são primos com n bits
  - GenModulus pode falhar com probabilidade negligenciável



Definição de dificuldade

- $\text{Factor}_{\mathcal{A},\text{GenModulus}}(n) = 1$ sse $p' \cdot q' = N$

- **Definição 8.45.** O problema da fatoração é difícil em relação ao algoritmo GenModulus se para todo algoritmo probabilístico polinomial $\mathcal{A}$, existe uma função negligenciável negl tal que

$$\Pr\left[\text{Factor}_{\mathcal{A},\text{GenModulus}}(n) = 1\right] \leq \text{negl}(n)$$

O problema RSA

- Fatoração difícil ⇒ existem funções unidirecionais
  - Porém não leva à cifras “práticas”
- **RSA:** problema relacionado (potencialmente mais fácil) que leva à cifras práticas
- O Corolário 8.22 nos mostra que, para N e $e > 2$ coprimo a $\phi(N)$, a operação de exponenciação por e é uma permutação
- Definimos $\left[y^{1/e} \bmod N\right]$ como o elemento único $x \in \mathbb{Z}_N^*$ tal que $x^e = y \bmod N$
- Informalmente, o problema RSA envolve computar $\left[y^{1/e} \bmod N\right]$ para um inteiro N de fatoração desconhecida

Experimento $\text{RSA-inv}_{\mathcal{A},\text{GenRSA}}(n)$

- A seguir, GenRSA é um algoritmo que, na entrada 1^n, produz N que é o produto de dois primos com n bits, além de e, d tal que $\gcd\big(e, \phi(N)\big) = 1$ e $ed = 1 \bmod \phi(N)$ (fácil quando se conhecem $pq = N$)

Desafiante

Adversário $\mathcal{A}$
Tempo $\text{poli}(n)$



$(N, e, d) \leftarrow \text{genRSA}(1^n)$

$y \leftarrow \mathbb{Z}_N^*$

(N, e, y)

x



- $\text{RSA-inv}_{\mathcal{A},\text{GenRSA}}(n) = 1$ sse $x^e = y \bmod N$

Dificuldade

- **Definição 8.46.** O problema RSA é difícil em relação ao algoritmo GenRSA se para todo algoritmo probabilístico polinomial $\mathcal{A}$, existe uma função negligenciável negl tal que

$$\Pr\left[\text{RSA-inv}_{\mathcal{A},\mathrm{GenRSA}}(n)\right] \leq \mathrm{negl}(n)$$

- É possível construir um algoritmo GenRSA de maneira eficiente a partir do algoritmo $\mathrm{GenModulus}$ que vimos antes

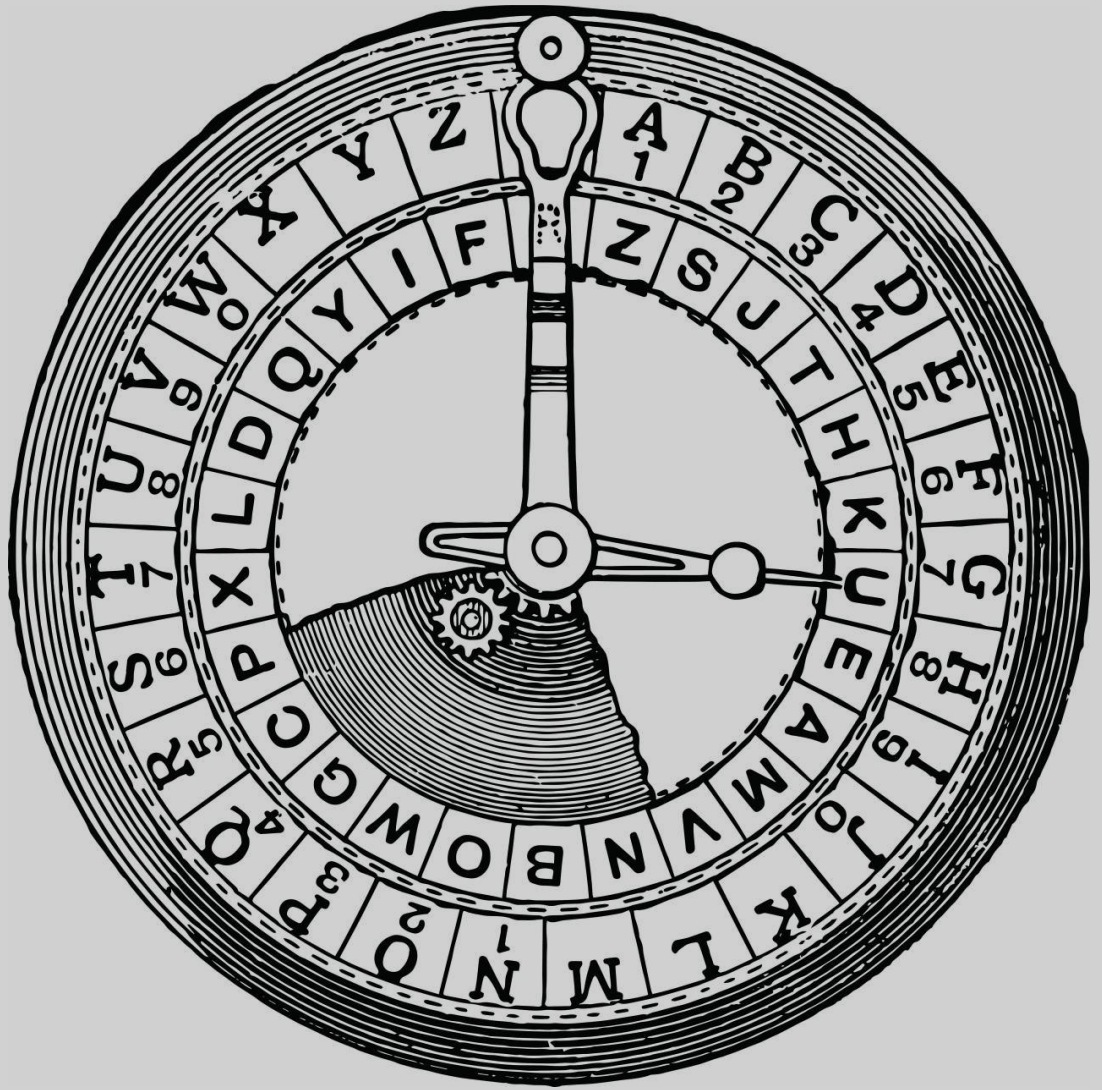
Algoritmo GenRSA

- GenRSA
  - Entrada: parâmetro de segurança 1^n
  - Saída: N, e, d tal que tal que $\gcd\big(e, \phi(N)\big) = 1$ e $ed = 1 \bmod \phi(N)$
  1. $(N, p, q) \leftarrow \mathrm{GenModulus}(1^n)$
  2. $\phi(N) \coloneqq (p-1)(q-1)$
  3. Escolha $e > 1$ tal que $\gcd\big(e, \phi(N)\big) = 1$
  4. Compute $d \coloneqq [e^{-1} \bmod \phi(N)]$
  5. Retorne N, e, d

Exemplo

- Digamos que $\mathrm{GenModulus}$ produza $(N, p, q) = (143, 11, 13)$
  - Então $\phi(N) = 120$
- Precisamos escolher e que é coprimo a 120
  - $e = 7$ basta
- Agora computamos $d = [e^{-1} \bmod \phi(N)]$
  - Com o algoritmo estendido de Euclides, encontramos $d = 103$
- O algoritmo tem como saída $(143, 7, 103)$
- **Observação:** é comum usar $e = 3$, porém temos que ser mais cuidadosos com a escolha dos números primos p e q

Cifra RSA “básica”



Discussão e ideia

- Ainda bastante utilizada
  - Porém recomenda-se a troca gradual por esquemas baseados nos problemas de Diffie-Hellman com grupos de curvas elípticas
- Ideia
  - Sejam $(N, e, d) \leftarrow \mathrm{GenRSA}(1^n)$
  - Lembrando que exponenciação por e é uma permutação
  - A chave pública é $\langle N, e \rangle$ e a chave privada é $\langle N, d \rangle$
  - Para cifrar $m \in \mathbb{Z}_N^*$, basta fazer $c \coloneqq [m^e \bmod N]$
  - Para recuperar m, basta fazer $m \coloneqq [c^d \bmod N] = [m^{ed} \bmod N] = [m^1 \bmod N]$

Construção 11.26

- Seja GenRSA como vimos anteriormente. Defina a cifra de chave pública abaixo:
  - Gen: ao receber 1^n como entrada, execute $\mathrm{GenRA}(1^n)$ para obter N, e e d. A chave pública é $\langle N, e \rangle$ e a chave privada é $\langle N, d \rangle$
  - Enc: ao receber a chave pública $pk = \langle N, e \rangle$ e uma mensagem $m \in \mathbb{Z}_N^*$ como entrada, retorna o texto cifrado

$$c \coloneqq [m^e \bmod N]$$

  - Dec: ao receber a chave privada $sk = \langle N, d \rangle$ e um texto cifrado $c \in \mathbb{Z}_N^*$, retorna a mensagem

$$m \coloneqq [c^d \bmod N]$$

Exemplo

- Digamos que GenRSA produza $(N, e, d) = (391, 3, 235)$
  - Neste caso, $\phi(391) = 352$ pois $391 = 17 \cdot 23$
  - A chave pública é $(391, 3)$
  - A chave privada é $(391, 235)$
- Vamos cifrar $m = 158 \in \mathbb{Z}_{391}^*$
  - $c \coloneqq [158^3 \bmod 391] = 295$
- Para decifrar
  - $m = [295^{235} \bmod 391] = 158$

Discussão sobre segurança

- Se o problema da fatoração é difícil, então é difícil de computar a chave privada a partir da chave pública
  - Necessário, mas não suficiente para segurança
- O esquema que vimos não é CPA, nem EAV-seguro
  - Determinístico
- Ataques para casos específicos
  - Quando a mensagem é um número menor que $\sqrt[e]{N}$, a exponenciação não precisa passar pelo módulo, e tirar a "e-ésima raiz" fica fácil
  - Mensagens com informação parcial conhecida
  - Mensagens relacionadas como m e $m + \delta$
  - Enviar a mesma mensagem para remetes diferentes (que usam o mesmo e)

Um caso em que a cifra é segura

Desafiante

Adversário $\mathcal{A}$
Tempo poli(n)



$(N, e, d) \leftarrow \text{genRSA}(1^n)$

$m \leftarrow \mathbb{Z}_N^*$

$c := [m^e \bmod N]$

(N, e, c)

m'

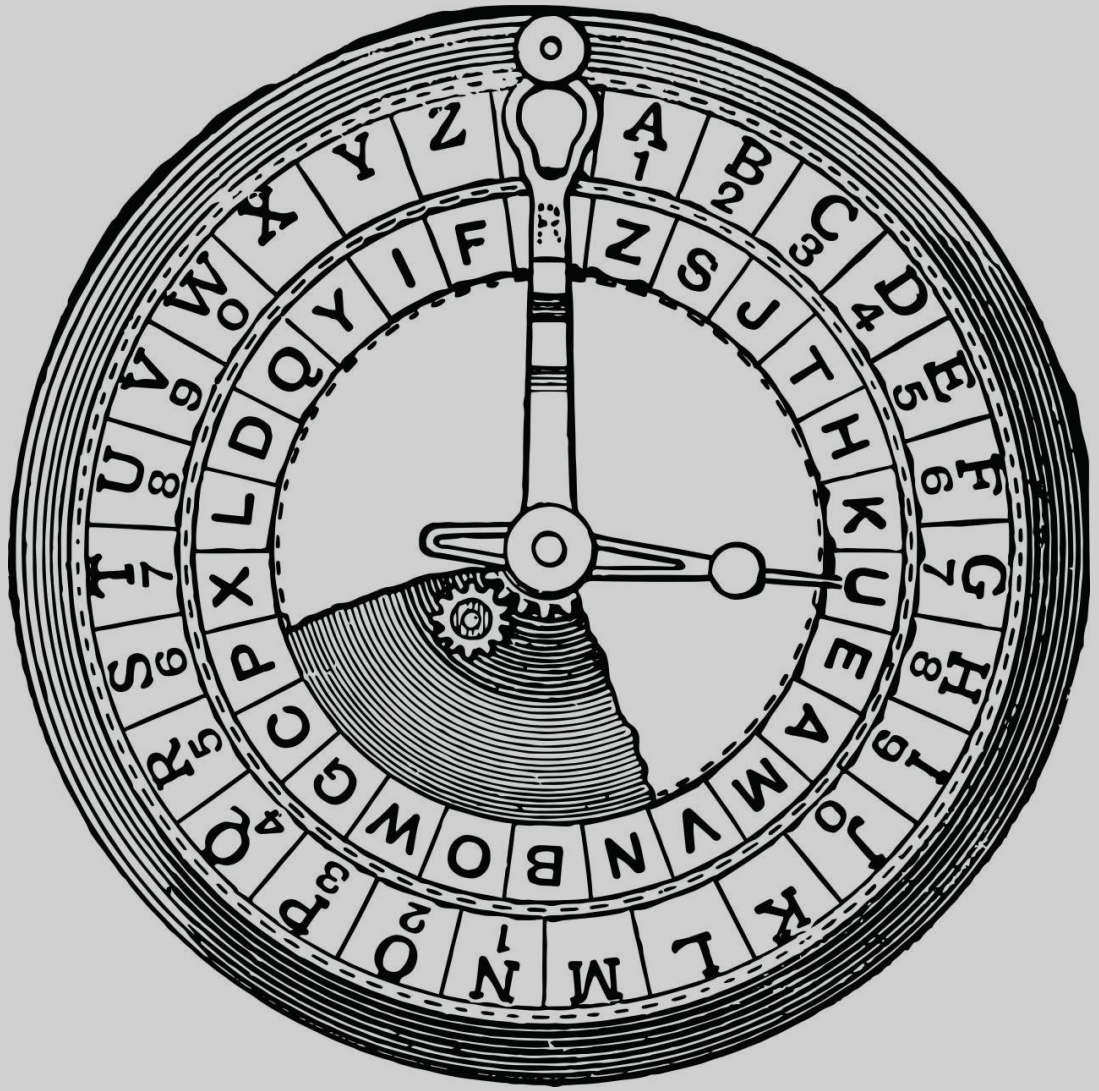


- Quando a mensagem m é escolhida de maneira aleatória, podemos concluir que o adversário só produz $m' = m$ com probabilidade negligenciável
  - Assumindo a dificuldade do problema RSA em relação a genRSA

Como consertar a segurança?

- Temos que tornar o esquema probabilístico
- Ideia geral: padding
  - Antes de cifrar a mensagem, adiciona um padding aleatório (que seja fácil de remover depois)
  - Um problema: se o padding é feito sem cuidado, leva a ataques tanto no estilo CPA como no estilo CCA (padding oracle)
  - Uma construção CCA-segura conhecida: OAEP (Optimal Asymmetric Encryption Padding)
    - Transforma $m \in \{0,1\}^*$ em um elemento de $\mathbb{Z}_N^*$ de maneira complicada (e aleatória)
    - Em particular, usando duas funções hash modeladas como oráculos aleatórios em uma rede de Feistel (capítulo 6 do livro, que pulamos)

Para fechar



Para fechar

- Hoje
  - Fatoração
  - RSA
    - Cifra RSA básica
- Próxima aula
  - Assinaturas digitais
    - Versão de chave pública de MACs