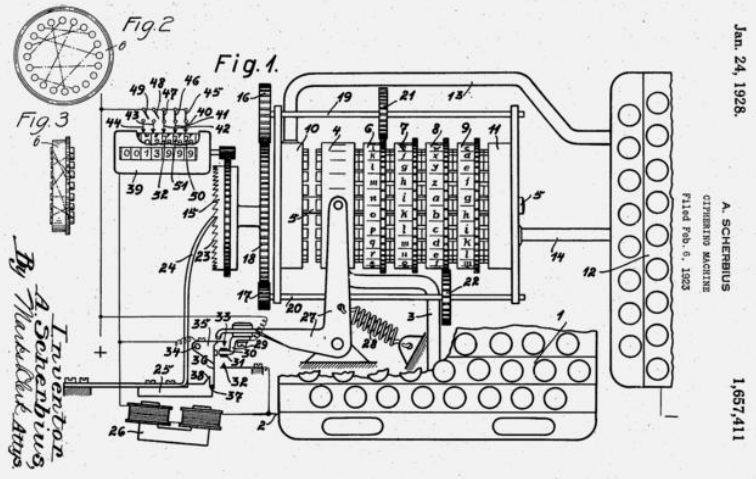
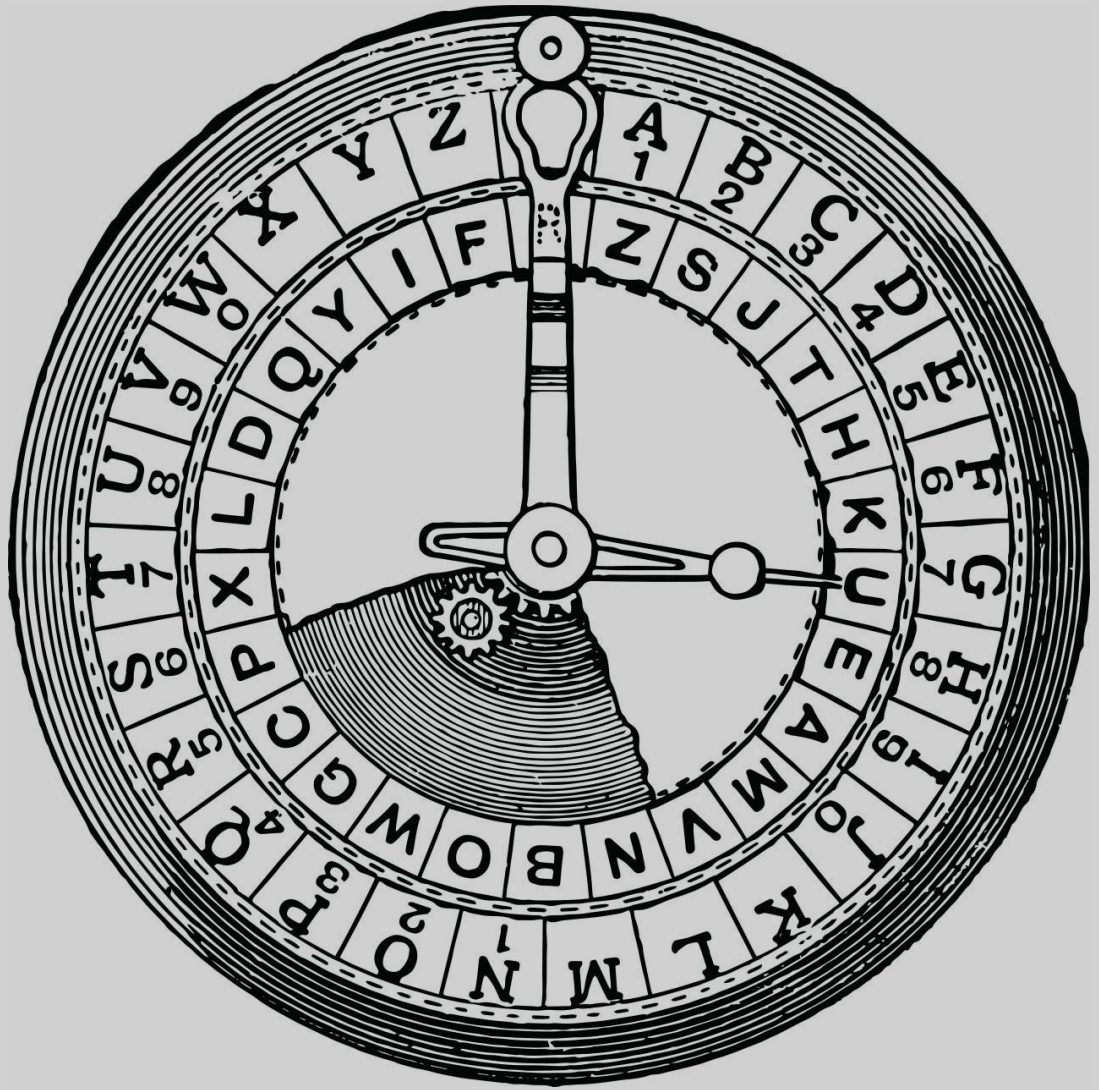
Segurança Computacional

Fundamentos da Criptografia

Nicollas Mocelin Sdroievski



Na última aula



Probabilidade condicional

- A *probabilidade condicional de* E_1 *dado* E_2 é definida a seguir:

$$\Pr[\, E_1 \mid E_2 \,] = \frac{\Pr[E_1 \wedge E_2]}{\Pr[E_2]}$$

- Segue que $\Pr[E_1 \wedge E_2] = \Pr[\, E_1 \mid E_2 \,] \cdot \Pr[E_2]$

- **Teorema A.8 (Teorema de Bayes).** Se $\Pr[E_2] \neq 0$, então

$$\Pr[\, E_1 \mid E_2 \,] = \frac{\Pr[\, E_2 \mid E_1 \,] \cdot \Pr[E_1]}{\Pr[E_2]}$$

Lei da probabilidade total

- Sejam $E_1, \ldots, E_n$ eventos tal que $\Pr[E_1 \vee \cdots \vee E_n] = 1$ e $\Pr[E_i \wedge E_j] = 0$ para $i \neq j$
  - $E_1, \ldots, E_n$ formam uma partição do espaço de todos os eventos possíveis
- Então para qualquer evento F

$$\Pr[F] = \sum_{i=1}^{n} \Pr[F \mid E_i] \cdot \Pr[E_i]$$

Segredo Perfeito

- O texto cifrado c não revela nada sobre a mensagem original m

$$\Pr[\, M = m \mid C = c \,] = \Pr[M = m]$$

- Para mensagens m com $\Pr[M = m] > 0$, isso equivale a

$$\Pr[\, C = c \mid M = m \,] = \Pr[C = c]$$

- Alternativamente, para todo texto cifrado c e par de mensagens m, m'

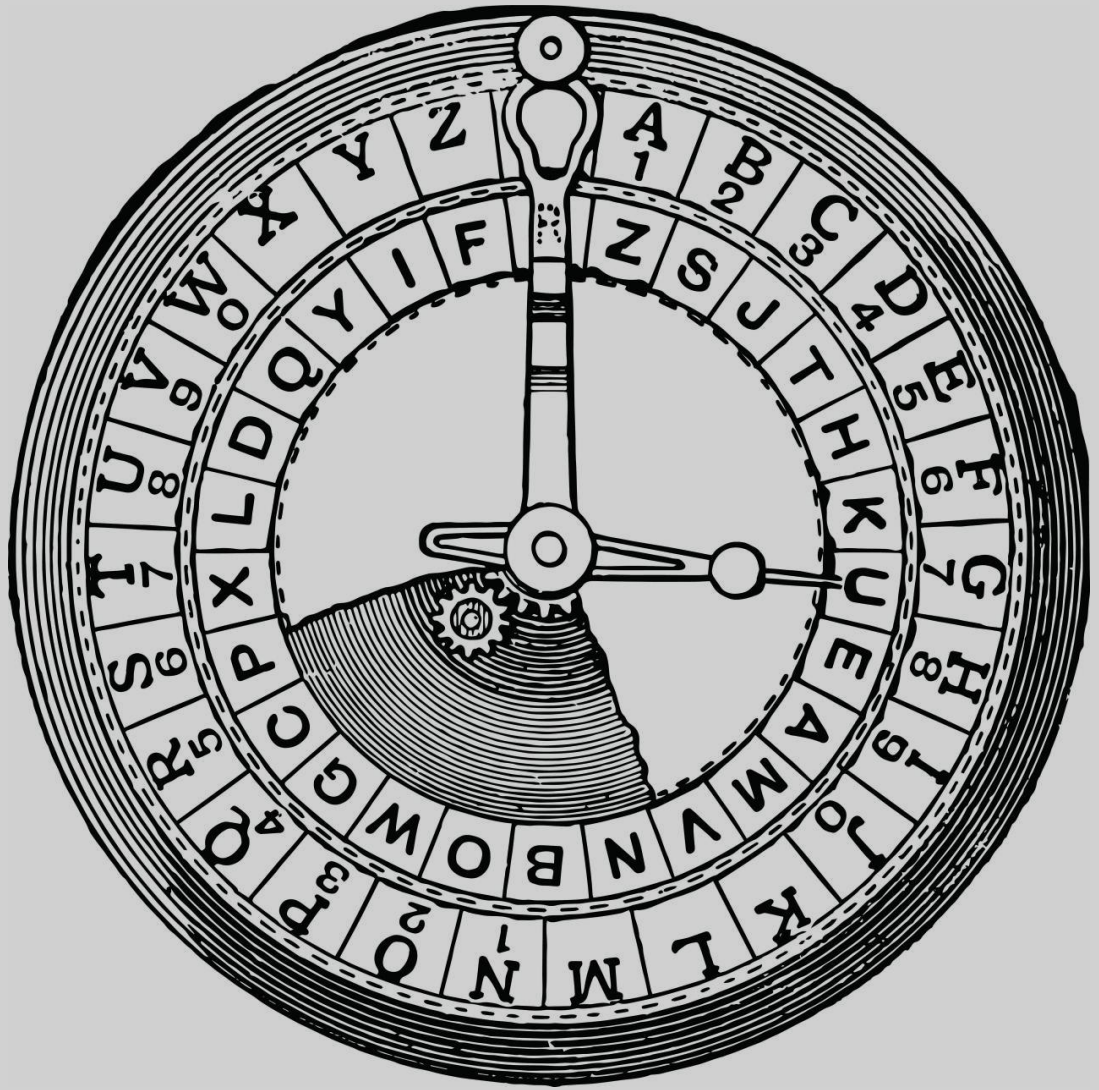
$$\Pr[\mathrm{Enc}_K(m) = c] = \Pr[\mathrm{Enc}_K(m') = c]$$

Limitações

- **Tamanho da chave:** se $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ é uma cifra perfeitamente segura com espaço de mensagens $\mathcal{M}$ e espaço de chaves $\mathcal{K}$, então $|\mathcal{K}| \geq |\mathcal{M}|$

- **Reuso da chave:** é impossível preservar segredo perfeito ao reutilizar a chave
  - Exercício 2.16 - a) do livro
  - Principal culpado: Dec é determinístico, então conseguimos perceber que as duas mensagens são iguais se os textos cifrados também são

Experimentos



Experimentos/Jogos de segurança

- Vamos definir segurança de cifras Π a partir de experimentos
- Lembrando que uma definição de segurança possui duas partes
  - Um modelo de ameaça (essencialmente, o poder do adversário)
  - Um objetivo de segurança (o que significa quebrar a segurança do protocolo)
- Por enquanto, adversários $\mathcal{A}$ que apenas interceptam o texto cifrado
- Queremos, porém, que a segurança não dependa da mensagem específica que foi cifrada
  - Como fazer isso?
  - Deixamos o próprio adversário “escolher” a mensagem

O experimento $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}$

Desafiante

Adversário $\mathcal{A}$



$m_0, m_1 \in \mathcal{M}$

$k \leftarrow \mathrm{Gen}$

$b \leftarrow \{0,1\}$

$c \leftarrow \mathrm{Enc}_k(m_b)$

$b' =$ chute para b



- A saída do experimento é 1 se $b' = b$ e 0 caso contrário
- Escrevemos $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi} = 1$ se a saída do experimento é 1
  - e dizemos que $\mathcal{A}$ obtém sucesso neste caso
- **Observação:** no livro, o desafiante é implícito

O experimento $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}$

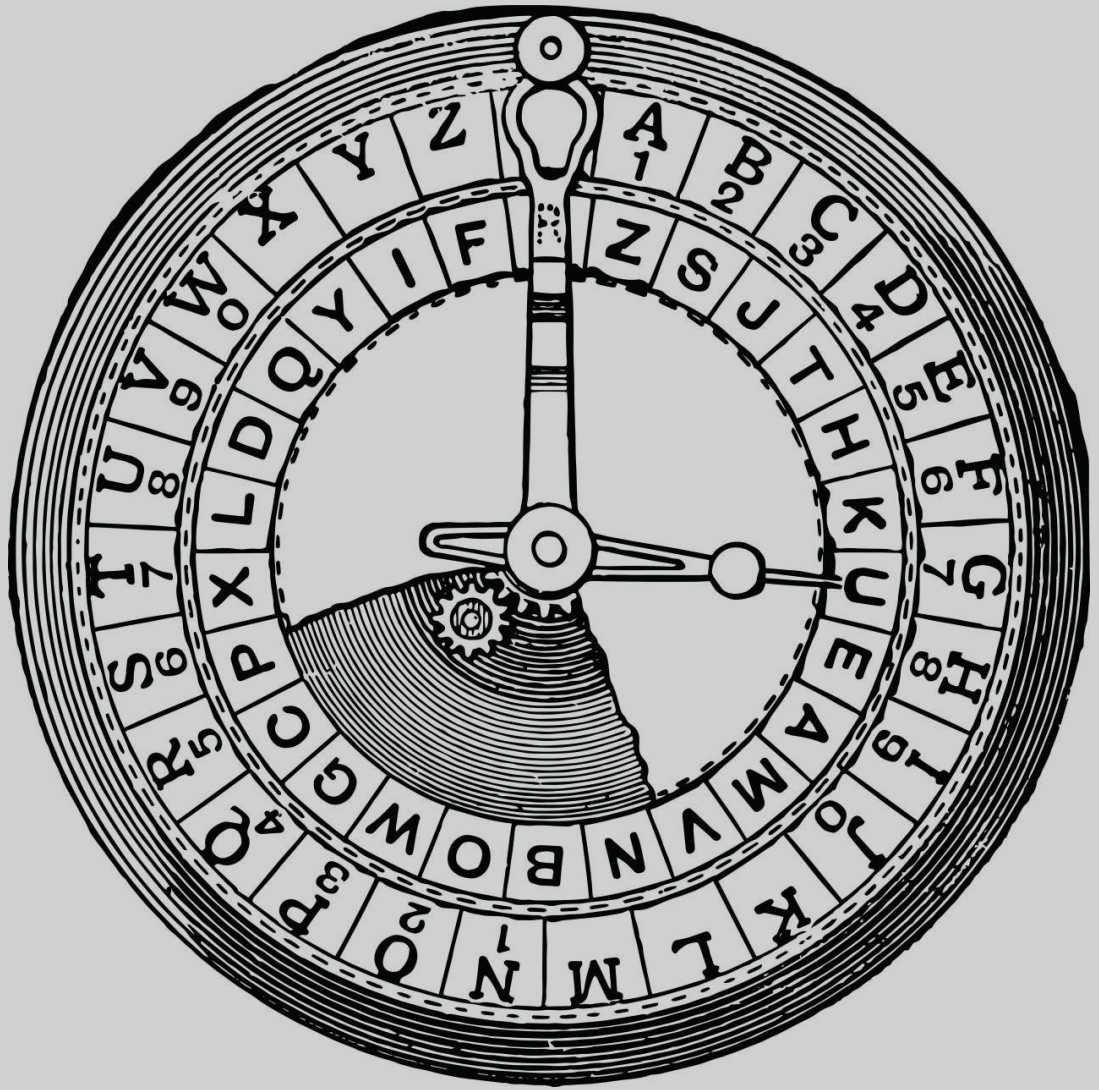
- É trivial para $\mathcal{A}$ obter sucesso com probabilidade $1/2$

- **Definição.** Uma cifra $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ com espaço de mensagens $\mathcal{M}$ é **perfeitamente indistinguível** se para todo $\mathcal{A}$ vale que:

$$\Pr\left[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi} = 1\right] = \frac{1}{2}$$

- **Lema 2.6.** Uma cifra Π é perfeitamente secreta se e somente se é perfeitamente indistinguível.

Segurança computacional



Segurança computacional

- **Segredo perfeito:** nenhuma informação da mensagem original é vazada, nem mesmo para um adversário com poder computacional ilimitado
- Podemos relaxar em dois pontos:
  - Podemos deixar vazar apenas um *pouco* de informação
  - E nos preocupar apenas com adversários que possuem algum tipo de *limitação computacional*
- Exemplo:
  - cifra que vaza informação com probabilidade no máximo 2^{-60} para "bisbilhoteiros" que investiram 200 anos de esforço computacional

Segurança concreta

- Um protocolo é (t, ε)-seguro se qualquer adversário rodando em tempo no máximo t só consegue sucesso ao "quebrar" a segurança do protocolo com probabilidade no máximo ε
- Assume-se que cifras modernas possuem segurança quase ótima
  - Chaves de tamanho n
    - 2^n possibilidades
  - Adversários rodando em tempo t conseguem "quebrar" a segurança da cifra com probabilidade apenas $ct/2^n$, para c constante

Exemplo

- Por simplicidade, $c = 1$
- Hoje, $n = 60$ é adequado contra desktops
  - 2^{60} ciclos em um processador de 4 GHz levam aproximadamente 9 anos
  - Um supercomputador faz o mesmo em 1 minuto, porém
- $n = 80$ seria mais prudente
  - 2 anos para um supercomputador realizar 2^{80} operações
- Hoje, comum usar chaves de tamanho $n = 128$ ou maiores
  - Assumindo que um adversário consegue recuperar uma mensagem cifrada com probabilidade 2^{-60} em um ano
  - Espera-se que uma mensagem seja recuperada a cada 100 bilhões de anos

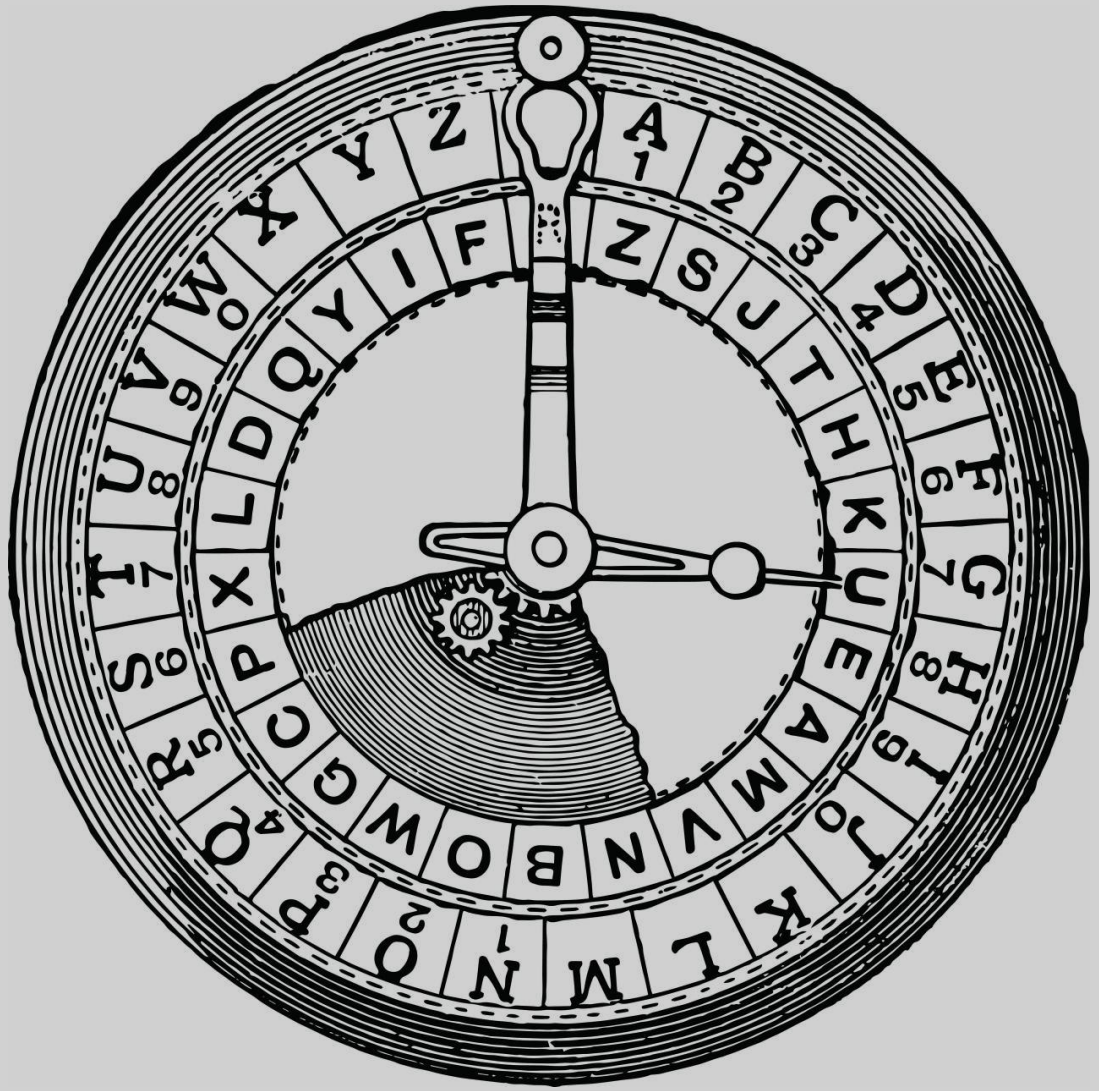
Segurança assintótica

- Todos os envolvidos recebem um parâmetro de segurança n
  - Tanto as partes “honestas” como o adversário
- Parametriza o “nível de segurança” do protocolo criptográfico
  - Tempo de execução do adversário é (polinomial) em termos de n
  - Assim como a probabilidade de sucesso
    - Que deverá ser “negligenciável”
  - O tempo de execução das partes “honestas” também deve ser polinomial
    - Porém, o adversário pode rodar, no geral, por mais tempo
- Como um exemplo simples, pode-se tomar n como o tamanho da chave
  - Chaves maiores são mais seguras contra ataques de força bruta
  - Porém tornam todo o protocolo um pouco mais lento
- Usaremos este tipo de definição

Exemplo

- Considere um protocolo em que
  - As partes honestas levam tempo $10^6 \cdot n^2$
  - Um adversário rodando em tempo $10^8 \cdot n^4$ consegue quebrar a segurança do protocolo com probabilidade no máximo $2^{-n/2}$
- Com $n = 80$ e computadores de 2 GHz
  - Partes honestas levam 3.2 segundos
  - Em 3 semanas, o adversário tem probabilidade apenas 2^{-40} de quebrar a segurança do protocolo
- Com $n = 160$ e processadores de 8 GHz
  - Partes honestas ainda levam 3.2 segundos
  - Para atingir probabilidade de sucesso de 2^{-80}, o adversário agora precisa de 13 semanas

Um pouco de complexidade



O modelo computacional

- Algoritmos A (máquinas RAM, máquinas de Turing, etc.) aleatorizados e de tempo polinomial
  - A cada passo computacional, o algoritmo pode "lançar uma moeda" e obter um bit aleatório
  - Existe um polinômio p tal que para toda entrada x, a computação de $A(x)$ leva tempo no máximo $p(|x|)$
- Questão técnica: as vezes o parâmetro de segurança é dado como entrada em unário, como 1^n

Mais sobre algoritmos aleatorizados

- Considere um algoritmo aleatorizado A
- A saída $A(x)$ é uma variável aleatória que depende das moedas internas de A
- Podemos "puxar essas moedas pra fora" e enxergar A como um algoritmo determinístico
  - Neste caso, A deve receber bits aleatórios como entrada
  - Denotamos por $A(x, r)$
  - Útil ao descrever probabilidades que dependem da aleatoriedade de A
  - **Exemplo:**

$$\Pr_{r \leftarrow \{0,1\}^n}[A(x, r) = 1] \geq \frac{1}{2}$$

Funções negligenciáveis

- **Definição 3.4.** Uma função f dos números naturais para números reais não-negativos é **negligenciável** se para todo polinômio positivo p existe um N tal que para todos os inteiros $n > N$, vale que

$$f(n) \leq \frac{1}{p(n)}$$

- Definições alternativas:
  - Para todo polinômio p e todo valor de n suficientemente grande, $f(n) < \frac{1}{p(n)}$
  - Para toda constante c existe N tal que para todo $n > N$, $f(n) < n^{-c}$

Exemplos

- 2^{-n} é negligenciável
  - pois para qualquer c e n suficientemente grande, $2^{-n} < n^{-c}$

$$2^{-n} < n^{-c} \Leftrightarrow 2^n > n^c \Leftrightarrow n > c \cdot \log n$$

- $2^{-\sqrt{n}}$ é negligenciável
  - pois para qualquer c e n suficientemente grande, $2^{-\sqrt{n}} < n^{-c}$

$$2^{-\sqrt{n}} < n^{-c} \Leftrightarrow 2^{\sqrt{n}} > n^c \Leftrightarrow \sqrt{n} > c \cdot \log n$$

- $n^{-\log n}$ é negligenciável
  - pois para qualquer c e n suficientemente grande, $n^{-\log n} < n^{-c}$

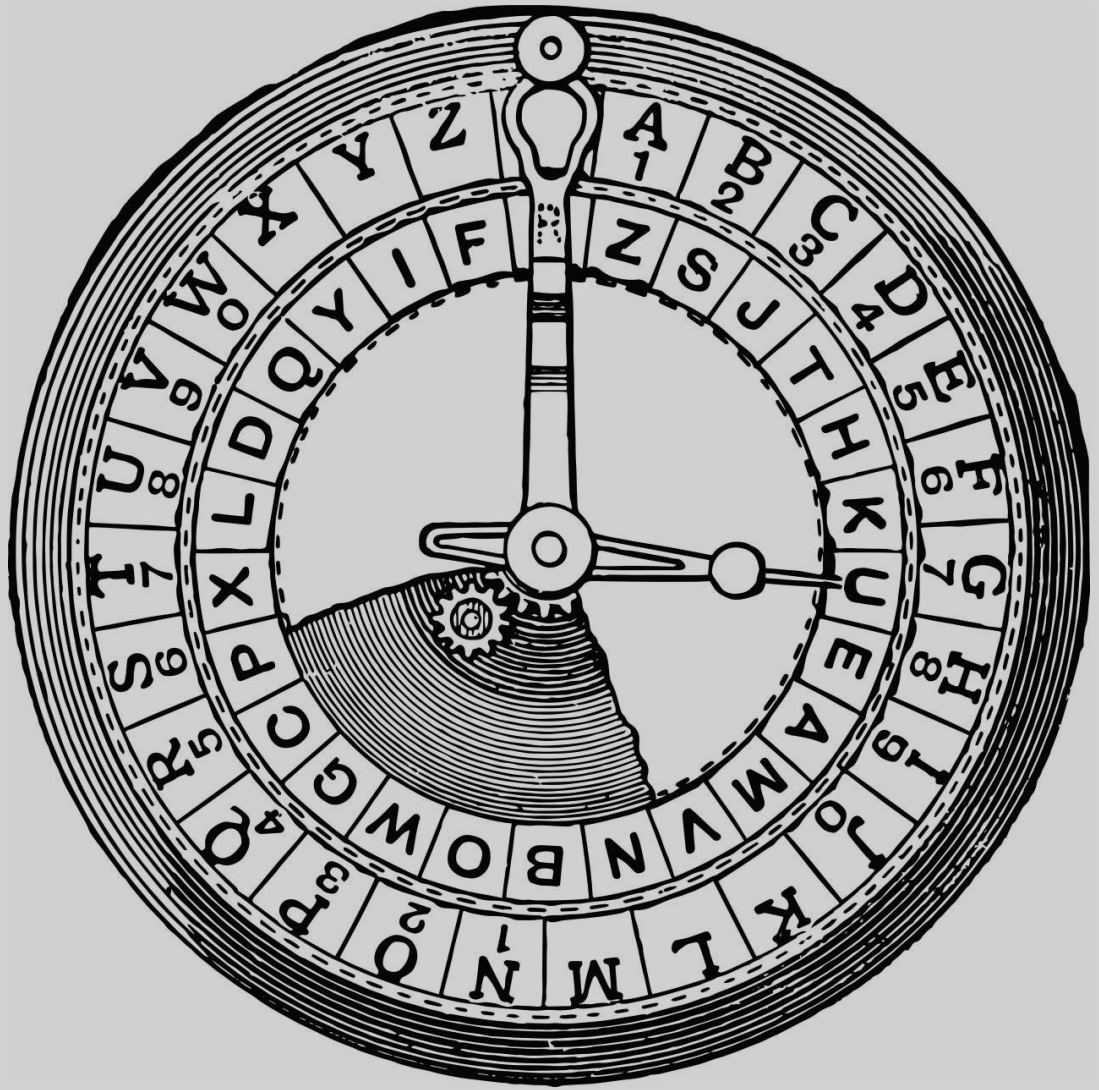
$$n^{-\log n} < n^{-c} \Leftrightarrow n^{\log n} > n^c \Leftrightarrow \log n > c$$

Alguns resultados

- **Proposição 3.6.** Sejam negl_1 e negl_2 duas funções negligenciáveis. Então,
  1. A função negl_3 definida como $\mathrm{negl}_3(n) = \mathrm{negl}_1(n) + \mathrm{negl}_2(n)$ é negligenciável
  2. Para qualquer polinômio positivo p, a função negl_4 definida por $\mathrm{negl}_4(n) = p(n) \cdot \mathrm{negl}_1(n)$ é negligenciável

- Evento ocorre com probabilidade negligenciável em um experimento
  - Repetimos o experimento uma quantidade polinomial de vezes
  - Probabilidade do evento ocorrer em alguma destas repetições?
  - Ainda negligenciável

Cifras simétricas com segurança computacional



Definição

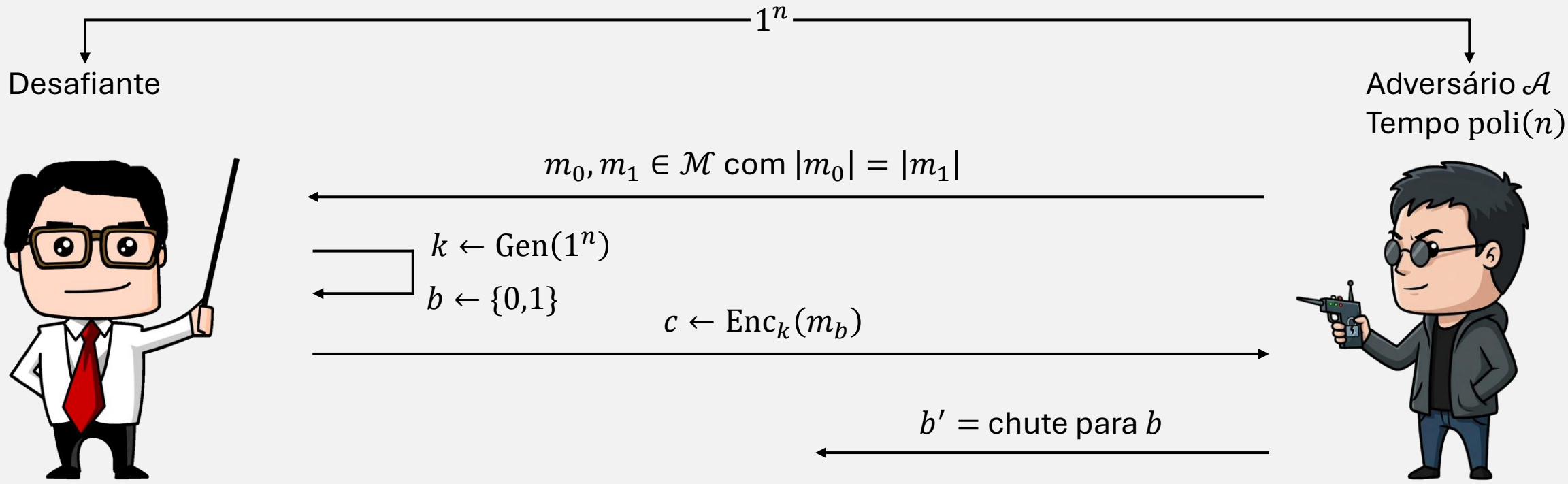
- **Definição 3.7.** Uma cifra de chave privada é uma tripla de algoritmos polinomiais $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ tal que:
  1. O algoritmo probabilístico Gen recebe entrada 1^n e tem como saída uma chave k.
     - Escrevemos $k \leftarrow \mathrm{Gen}(1^n)$ para indicar a saída do algoritmo.
     - Sem perda de generalidade, assumimos que $|k| \geq n$ para $k \leftarrow \mathrm{Gen}(1^n)$.
  2. O algoritmo probabilístico Enc recebe como entrada uma chave k e uma mensagem $m \in \{0,1\}^*$, e tem como saída um texto cifrado c.
     - Escrevemos $c \leftarrow \mathrm{Enc}_k(m)$ para indicar a saída do algoritmo.
  3. O algoritmo determinístico Dec recebe como entrada uma chave k e um texto cifrado c, e tem como saída uma mensagem m ou erro.
     - Escrevemos $m := \mathrm{Dec}_k(m)$ para indicar a saída do algoritmo (caso não haja erro).
     - Erros são denotados pelo símbolo $\perp$.

Mudanças no experimento $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}$

- Parâmetro de segurança n é parte da entrada para o Desafiante e o Adversário $\mathcal{A}$
  - Agora o experimento é $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)$
  - Ambos Desafiante e $\mathcal{A}$ executam em tempo polinomial em n
- $\mathcal{A}$ pode obter sucesso com probabilidade negligenciavelmente maior que $1/2$
- $\mathcal{A}$ agora precisa produzir duas mensagens do mesmo tamanho
  - O que isso significa em relação à garantias de segurança?
  - Não estamos mais pedindo que a cifra esconda o tamanho da mensagem (caso permita a cifragem de mensagens de tamanhos diferentes)

O experimento $\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n)$

- Mesmo experimento, mas ambos desafiante e adversário recebem o parâmetro de segurança n como entrada
- E o adversário é limitado a tempo polinomial em n



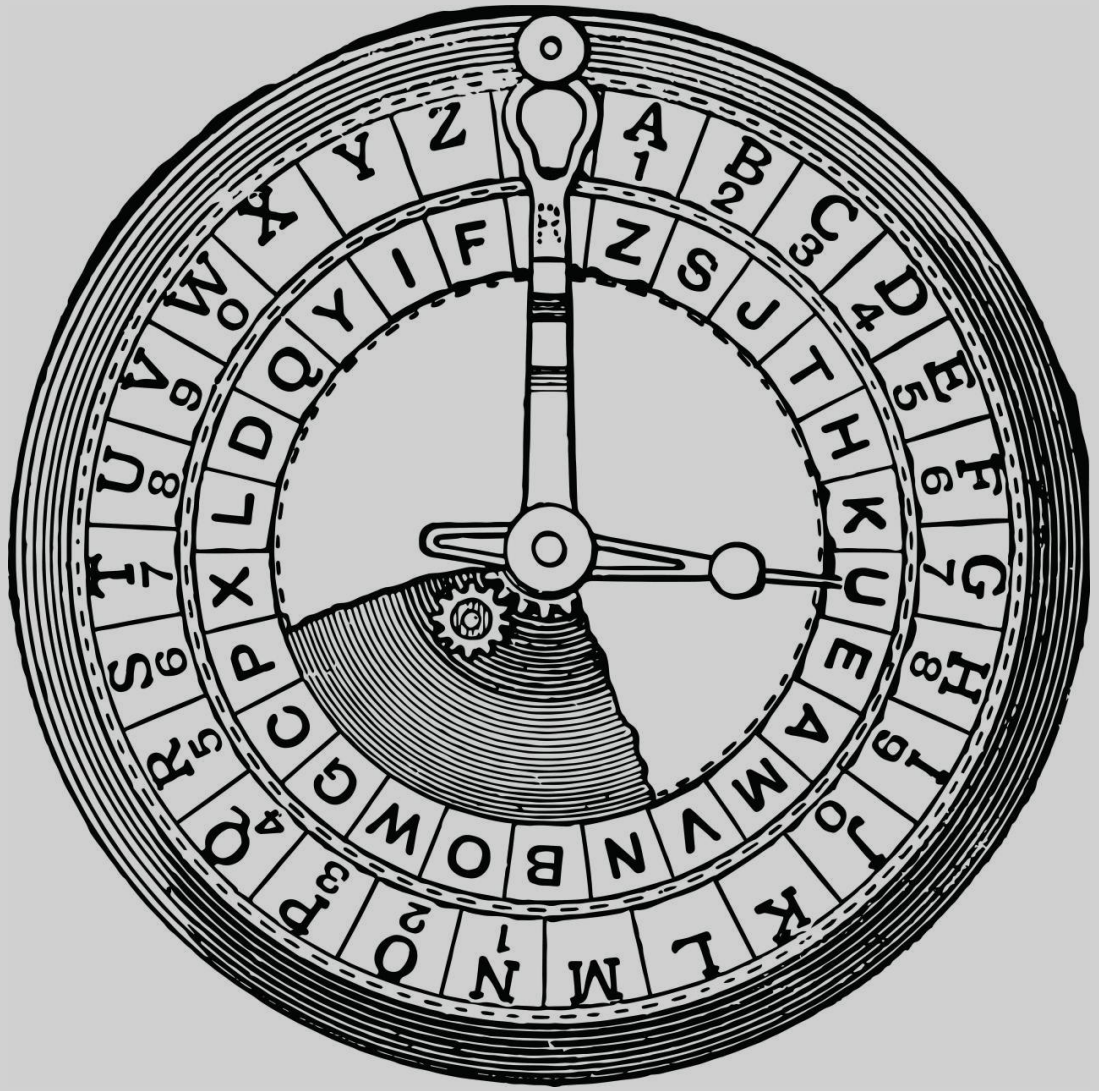
Definição de segurança

- **Definição 3.8.** Uma cifra de chave privada $\Pi = (\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ possui cifragens indistinguíveis na presença de um bisbilhoteiro, ou é EAV-segura, se para todos os algoritmos probabilísticos polinomiais $\mathcal{A}$ existe uma função negligenciável negl tal que, para todo n,

$$\Pr\left[\mathrm{PrivK}^{\mathrm{eav}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \mathrm{negl}(n),$$

onde as probabilidades são sobre a aleatoriedade de $\mathcal{A}$ e do Desafiante (para escolher a chave, o bit b, e de Enc , caso necessário)

Para fechar



Para fechar

- Hoje
  - Experimentos
  - Segurança computacional
  - Funções negligenciáveis
  - Definição de segurança contra “bisbilhoteiros”
- Próxima aula
  - Como provar que um protocolo criptográfico é seguro?
  - Reduções