Stream ciphers 2 +

PSEUDO RANDOM GENERATORS

↳ Como fazer o OTP prático? (Problema das chaves longas)

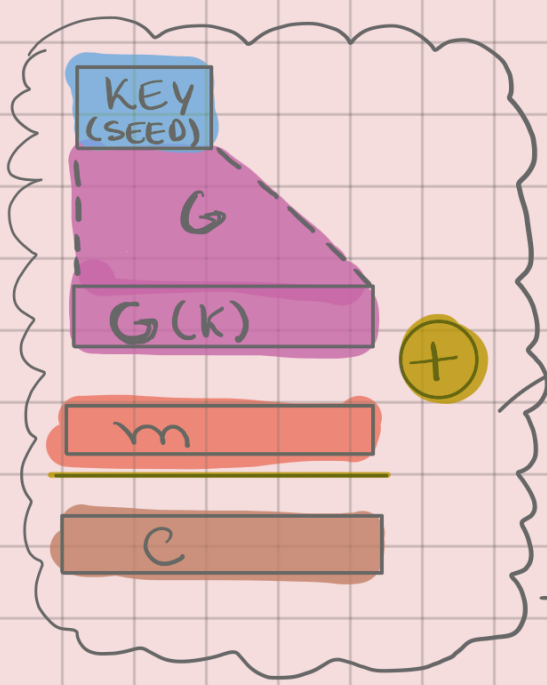
↳ Ideia: substituir a chave aleatória por pseudo-aleatória

PRG

* DEF: É uma função $G: \{0,1\}^s \rightarrow \underbrace{\{0,1\}^n}_{\text{OUTPUT}}$, $n >> s$

seed

somente é a única coisa aleatória, ainda se mantendo um algoritmo determinístico.



$$C = E(K,m) := m \oplus G(K)$$
$$D(K,C) := C \oplus G(K)$$

* Mas... Esse padrão não tem um segredo perfeito. Sua chave é menor que a mensagem.
* Precisamos de uma nova definição de segurança e novas propriedades.

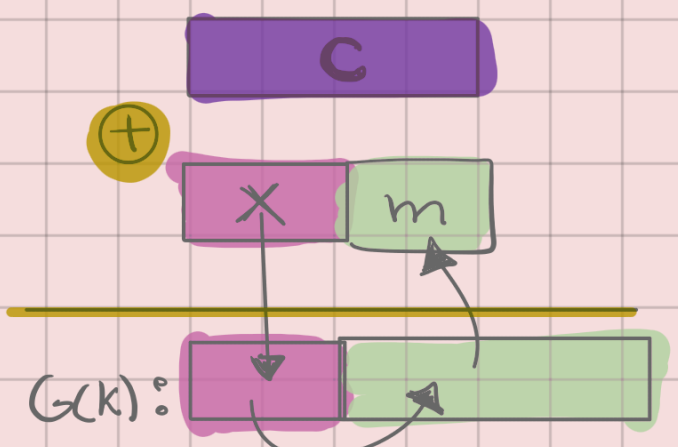
PRG DEVE SER IMPREVISÍVEL → Vi não consegue prever.

$$\exists i : G(K)\big|_{1,\dots,i} \xrightarrow{\text{Alg.}} G(K)\big|_{i+1,\dots,n}$$

→ dados os primeiros termos, obtém lógica de todos os restantes.

Então:

Exemplo SMTP: começ sempre com "X", atacante já sabe um prefixo, o que se faz:



pode prever o resto

Questão: · $G: K \rightarrow \{0,1\}^n$ para todo K: $XOR(G(k)) = 1$
· G é imprevisível?

R: Não, dado n-1 bits é possível prever o último bit.

PRG FRACO → não utilizado em criptografia.

* Gerador congruente linear: 3 parâmetros: 2 inteiros: A e B
1 primo: p

↳ r[0] ≡ seed

loop:
r[i] ← a.r[i-1]+b mod p
output são alguns bits de r[i]
i++

↳ fácil de prever.

* glibc random(): → nunca utilizar essa biblioteca para criptografia

r[i] ← (r[i-3] + r[i-3]) % 2^{32}
output r[i] >> 1

Insignificância e não-Insignificância

neg — non-neg

* Na prática: • "ε" é um escalar e
  - ε non-neg: $\varepsilon \geq 1/2^{30}$
  - ε neg: $\varepsilon \leq 1/2^{80}$

* Na teoria: • ε é uma função $\varepsilon : \mathbb{Z}^{\geq 0} \to \mathbb{R}^{\geq 0}$ e
  - ε non-neg: $\exists d : \varepsilon(\lambda) \geq 1/\lambda^d$ ($\varepsilon \geq 1/$ polinômio)
  - ε neg: $\forall d, \lambda \geq \lambda_d : \varepsilon(\lambda) \leq 1/\lambda^d$ ($\varepsilon \leq 1/$ polinômio)

p/ muitos λ

Exemplos:

* $\varepsilon(\lambda) = 1/2^\lambda$: non.
* $\varepsilon(\lambda) = 1/\lambda^{1000}$: non-neg.
* $\varepsilon(\lambda) = \begin{cases} 1/2^\lambda \text{ para } \lambda \text{ impar} \\ 1/\lambda^{1000} \text{ para } \lambda \text{ par} \end{cases}$ } non-neg.