PTC 2550 - Aula 20

5.2 Princípios de criptografia

(Kurose, p. 587 - 626)
(Peterson, p. 444-454)

14/06/2017

Capítulo 5 - Sumário

5.1 O que é segurança de rede?

5.2 Princípios de criptografia

5.3 Integridade de mensagem, autenticação

5.4 Tornando o e-mail seguro

5.5 Tornando conexões TCP seguras: SSL

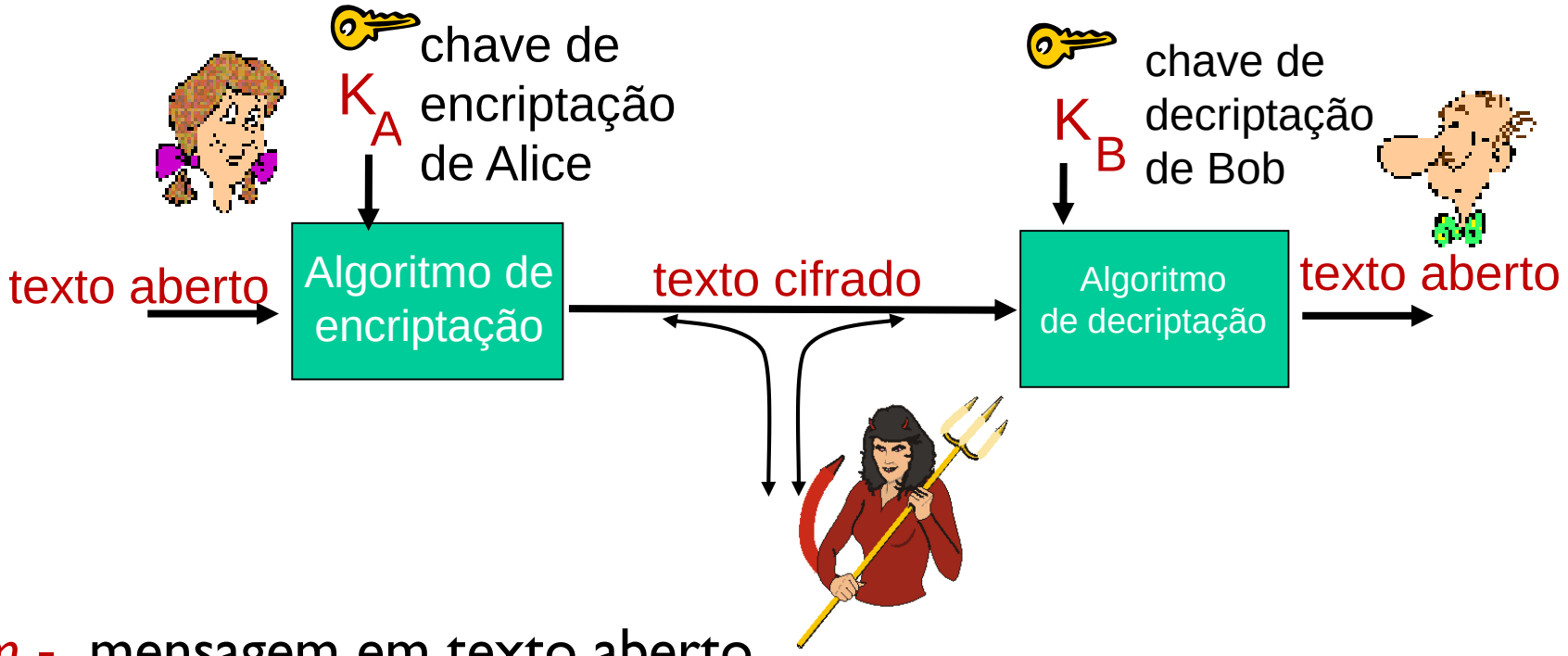
5.6 Segurança na camada de rede: IPsec

5.7 Tornando LANs sem fio seguras

5.8 Segurança operacional: *firewalls* e IDS

A linguagem da criptografia

Algoritmos de encriptação usuais são públicos e disponíveis: MD5, RSA, 3DES, AES, etc. Segredo está nas chaves…



m - mensagem em texto aberto

$K_A(m)$ - texto cifrado, criptografado com a chave K_A

$m = K_B(K_A(m))$

Algoritmos de criptografia

- 2 grandes grupos de algoritmos
  - Sistemas de chave simétrica
    - Chaves de Alice e Bob são idênticas e secretas
  - Sistemas de chave pública
    - Par de chaves é usado – uma é conhecida público (inclusive Alice e Bob) e a outra é conhecida apenas por Alice ou Bob (mas não por ambos)

Criptografia de Chave Pública

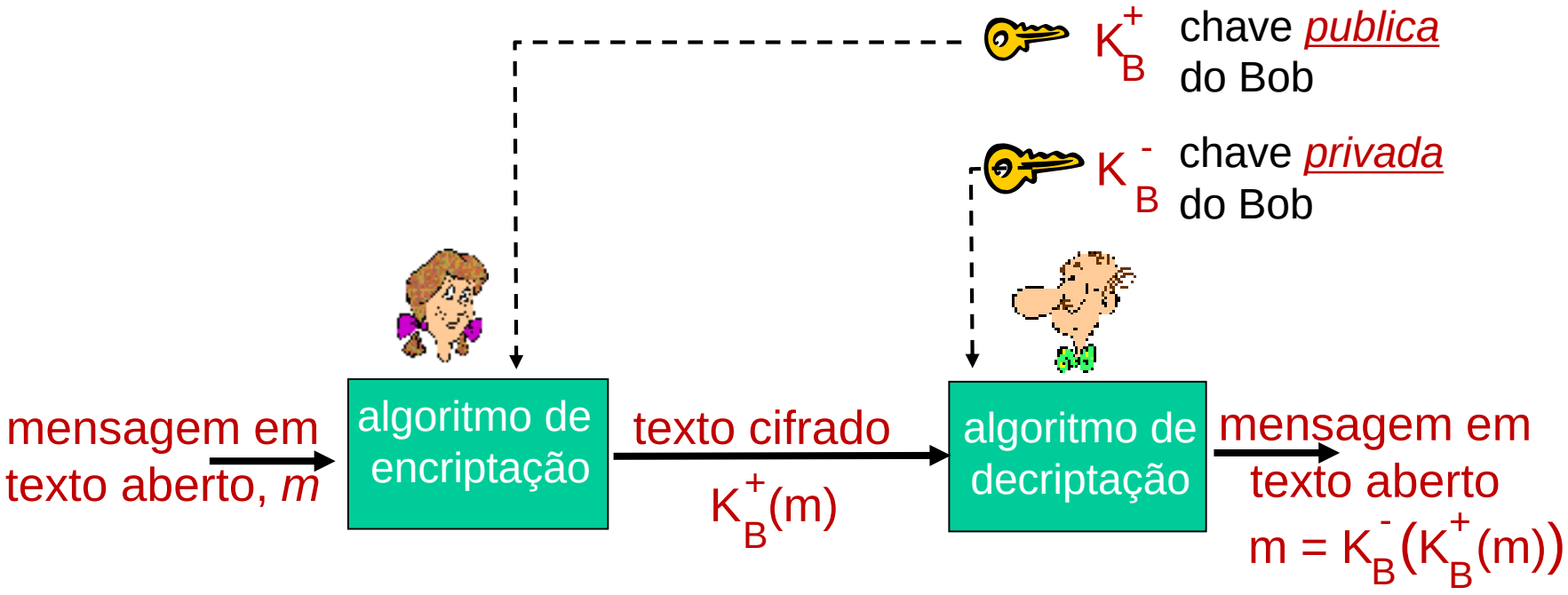
criptografia de chave simétrica

- Por mais de 2000 anos (desde Cesar até década de 1970) – única forma utilizada
- requer que transmissor e receptor saibam chave secreta compartilhada
- Q: Como concordar com uma chave para começar (particularmente se eles nunca se encontram)???

criptografia de chave pública

- abordagem radicalmente diferente [Diffie-Hellman76, RSA78 (Rivest, Shamir, Adleman (MIT), CESG],
- transmissor e receptor *não* compartilham chave secreta
- *chave de encriptação pública* conhecida por *todos*
- *chave de decriptação privada* conhecida apenas pela receptor

Criptografia de Chave Pública



Algoritmos de encriptação com chave pública

requisitos:

1. É necessário que $K_B^+(\cdot)$ e $K_B^-(\cdot)$ sejam tais que

$$K_B^-(K_B^+(m)) = m$$

2. dada a chave pública K_B^+, é *(quase)* impossível computar a chave privada K_B^-

RSA: algoritmo de Rivest, Shamir, Adelson (quase sinônimo de criptografia de chave pública)

Pré-requisito: aritmética modular

- *x mod n* = resto de *x* quando dividido por *n*
- Exemplo: *19 mod 5 = ?* 4
- Resultados:

  [(a mod n) + (b mod n)] mod n = (a+b) mod n

  [(a mod n) - (b mod n)] mod n = (a-b) mod n

  *[(a mod n) * (b mod n)] mod n = (a*b) mod n*
- Assim,

  $(a \bmod n)^d \bmod n = a^d \bmod n$
- *Exemplo: Verifique a igualdade acima para a=14, n=10, d=2*

$(a \bmod n)^d \bmod n = 4^2 \bmod 10 = 6$

$a^d = 14^2 = 196 \quad a^d \bmod 10 = 6$

RSA: preparando-se

- mensagem: sequência de bits
- sequência pode ser representada unicamente por um número
- assim, encriptar uma mensagem é equivalente a encriptar um número

exemplo:

- *m= 10010001*. Essa mensagem é unicamente representada pelo número decimal145.
- para encriptar *m*, criptografa-se o número correspondente que resulta em um novo número (o texto cifrado).

RSA: Criando as chaves pública e privada

Tarefas do receptor (Bob)

1. escolher 2 números primos p, q grandes.
   (recomendação RSA: produto tenha 1024 bits)

2. computar $n = pq$, $z = (p-1)(q-1)$

3. Escolher e (com $e<n$) que não tenha fatores comuns com z (e, z são *primos entre si*).

4. Encontrar d tal que $ed-1$ é divisível por z
   (ou seja: $ed \bmod z = 1$).

5. A chave pública é $\underbrace{(n,e)}_{K_B^+}$ e *a chave privada* é $\underbrace{(n,d)}_{K_B^-}$.

RSA: encriptação, decriptação

0. dados (n,e) e (n,d) como computados anteriormente

1. para encriptar mensagem m $(<n)$, compute

$c = m^e \bmod n$

2. para decriptar sequência de bits recebida, c, compute

$m = c^d \bmod n$

e a mágica acontece!

$$m = \underbrace{(m^e \bmod n)}_{c}{}^d \bmod n$$

Exemplo RSA:

Bob escolhe *p=3*, *q=5.* Exemplo *“toy model”*

1. *Calcule n e z.*
2. *Faça escolhas adequadas para e e d. Quais são as chaves públicas e privadas de Bob?*
3. *Encripte a sequência m=00001100.*
4. *Decripte e verifique o resultado*

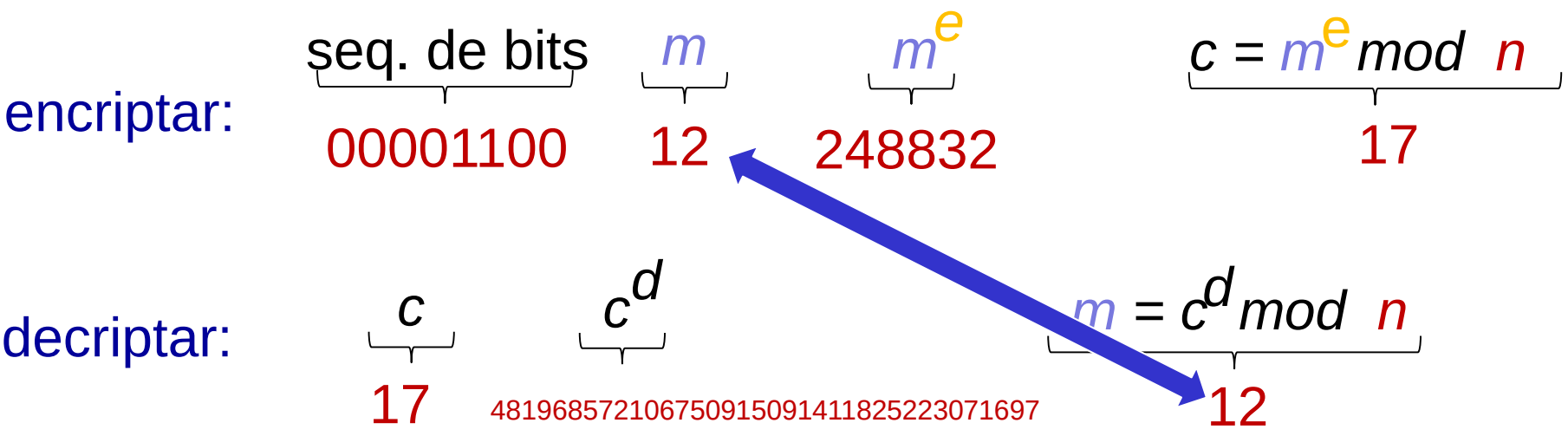
Outro exemplo RSA:

Bob escolhe $p=5$, $q=7$. Então $n=35$, $z=24$.

$e=5$ (assim e e z são primos entre si).

$d=29$ (assim $ed-1$ é divisível por z).

criptografando mensagens de 8-bits.



Perguntas: Como escolher p, q? Como escolher e, d? Como fazer exponenciação com números tão grandes?

Veja [Kaufman 1995], por exemplo.

Por que o RSA funciona?

- necessário mostrar que $c^d \bmod n = m$ em que $c = m^e \bmod n$
- Resultado: para quaisquer x e y: $x^y \bmod n = x^{(y \bmod z)} \bmod n$
  - em que n= pq and $z = (p-1)(q-1)$, p e q primos
- Assim,
-

$$
\begin{aligned}
c^d \bmod n &= (m^e \bmod n)^d \bmod n \\
&= m^{ed} \bmod n \\
&= m^{(ed \bmod z)} \bmod n \\
&= m^1 \bmod n \\
&= m
\end{aligned}
$$

RSA: outra propriedade importante

A seguinte propriedade será *muito* útil posteriormente:

$$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$$

usar a chave pública primeiro, seguida da chave privada

usar a chave privada primeiro, seguida da chave privada

resultado é o mesmo!

Por que $K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$?

Resultado segue diretamente da aritmética modular:

$$(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$$
$$= m^{de} \bmod n$$
$$= (m^d \bmod n)^e \bmod n$$

Por que o RSA é seguro?

- suponha que você conhece a chave pública de Bob *(n,e)*. Quão difícil é determinar *d*?
- Essencialmente é necessário encontrar *p* e *q* a partir de *n*.
- Daí obtém-se facilmente *z* e d

- *Notícia boa: Não se conhecem algoritmos rápidos para fatorar um número!*
- *Notícia ruim: Ainda não se sabe se existem algoritmos rápidos para fatorar um número*

RSA na prática: chaves de sessão

- exponenciação no RSA é computacionalmente custoso
- DES é no mínimo 100 vezes mais rápido do que RSA em *software* e entre 1 000 e 10 000 vezes em *hardware*
- Ideia: usar criptografia de chave pública para estabelecer conexão segura, então usar usar chave de sessão simétrica para encriptar dados

chave de sessão, K_S

- Bob e Alice usam RSA para trocar chaves simétricas K_S
- uma vez que ambos têm K_S, usam chave simétrica para criptografia