

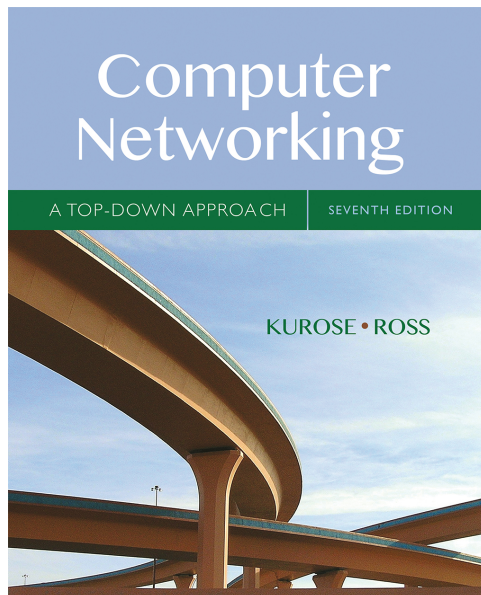
# Segurança em Redes

## Comunicações por Computador

Mestrado Integrado em Engenharia Informática

3º ano/2º semestre

2017/2018

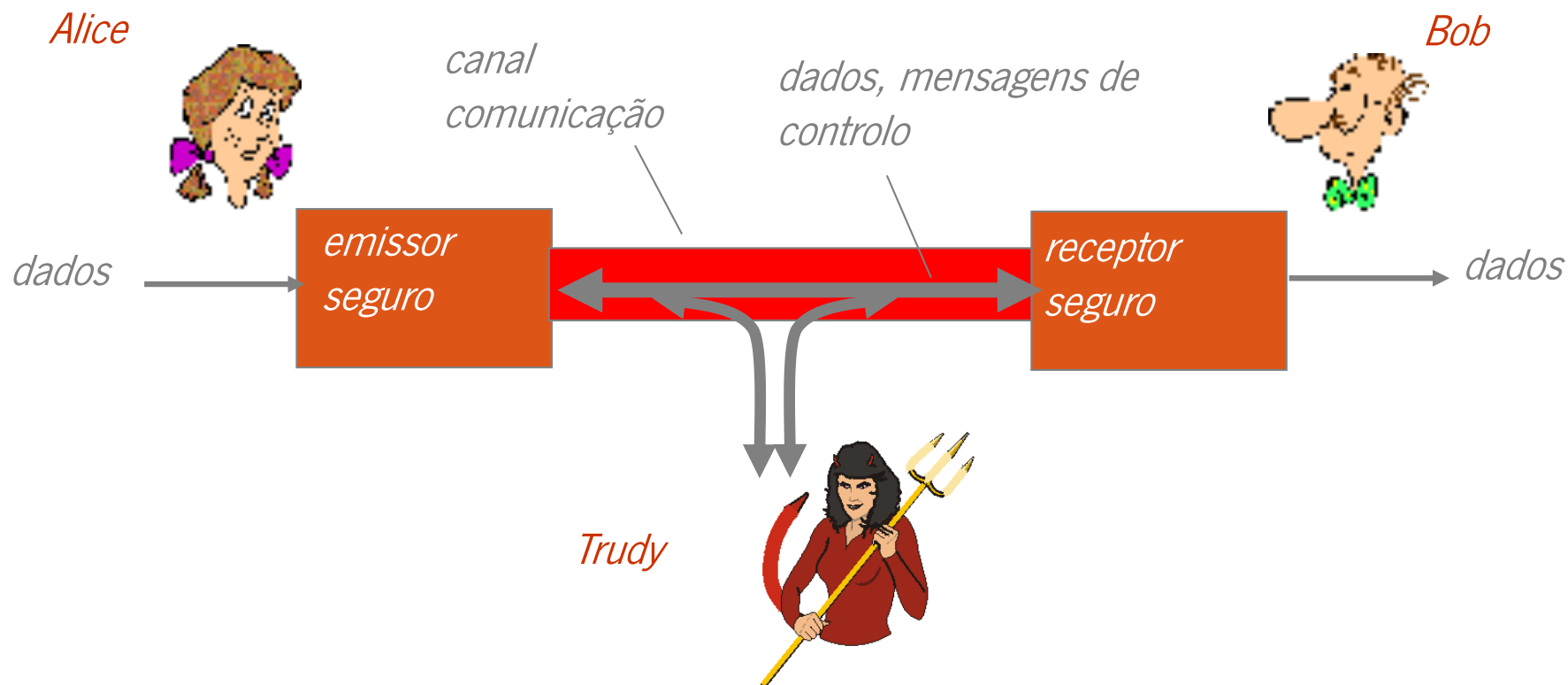


### *Capítulo 8: Security in Computer Networks*

# Actores: os amigos e os inimigos



- Personagens bem conhecidas do mundo da segurança 😊
- Alice e o Bob estão apaixonados e querem comunicar de forma segura;
- Que pode *Trudy* (a intrusa) fazer?



# Que podem fazer os “maus”?



- **espionagem**: interceptação indevida de mensagens
- **inserção** de mensagens numa conexão (comunicação)
- **disfarce**: pode fingir (*spoof*) endereços de origem nos pacotes (ou qualquer outro campo dos pacotes)
- **desviar sessões (*hijacking*)**: “tomar conta” de conexões que estão a decorrer, remover o emissor ou o receptor, colocando-se no lugar destes
- **negação de serviço**: impedir premeditadamente que um serviço seja usado por outros (ex: sobrecarregando-o de algum modo)

# Propriedades de uma comunicação segura



**Confidencialidade:** só o emissor e o receptor indicado devem “perceber” o conteúdo das mensagens

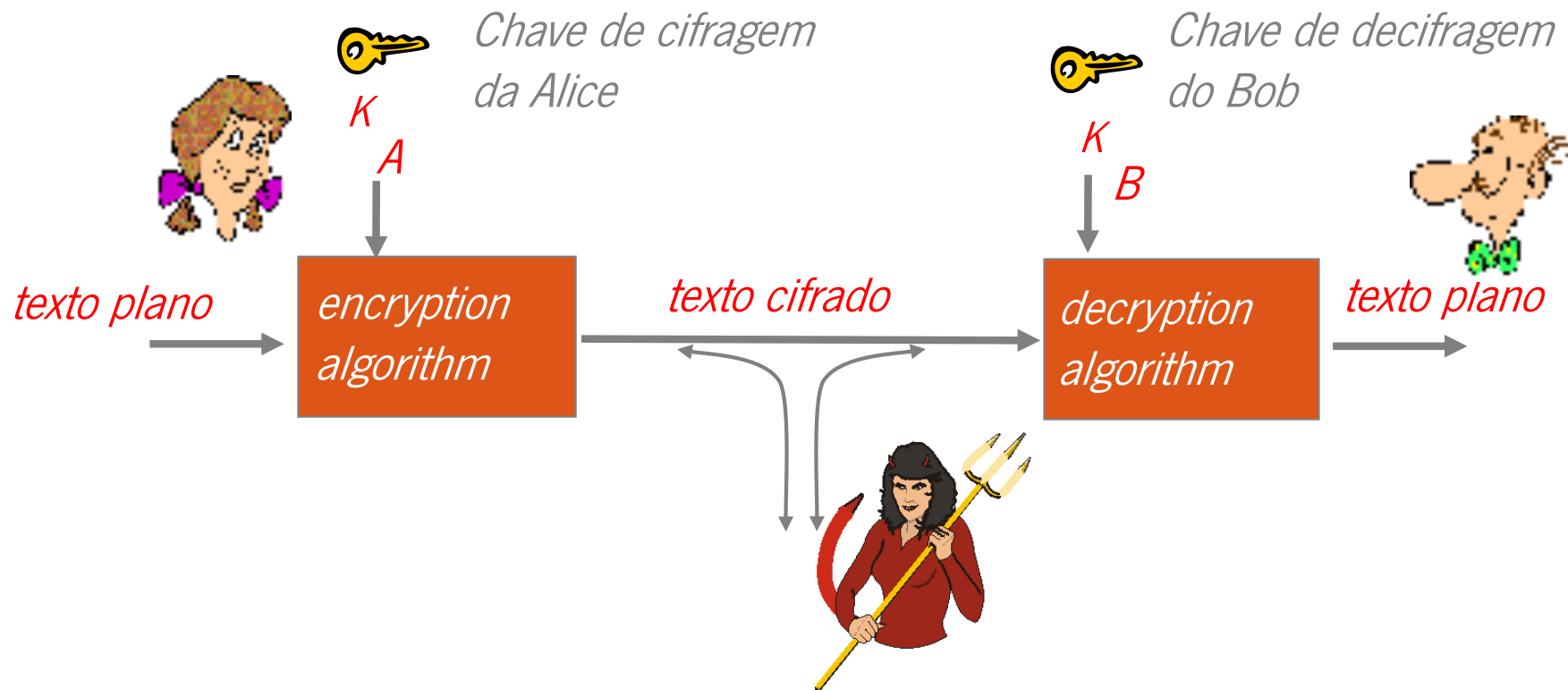
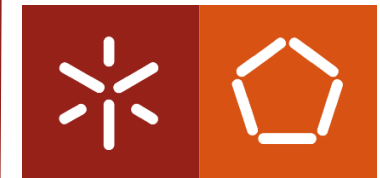
**Autenticação:** emissor e receptor pretendem confirmar a identidade um do outro

**Integridade da mensagem:** emissor e receptor querem garantir que a mensagem não foi alterada (no percurso pela rede, antes do envio ou depois da recepção) sem que tal possa ser imediatamente detectado

**Não Repúdio:** evidências que impeçam intervenientes de negar comunicação

**Acesso e Disponibilidade:** serviços devem estar acessíveis e com disponibilidade para os seus utilizadores

# A “linguagem” da criptografia



**criptografia de chave simétrica:** emissor e receptor usam a mesma chave

**criptografia de chave pública:** uma chave para cifrar (pública)  
outra para decifrar (privada)

# Criptografia de chave simétrica



**Cifra de substituição:** substituir uma coisa por outra

- cifra monoalfabética: substitui uma letra por outra

*Text plano:   abcdefghijklmnopqrstuvwxyz*



*Texto cifrado:   mnbvcxzasdfghjklpoiuytrewq*

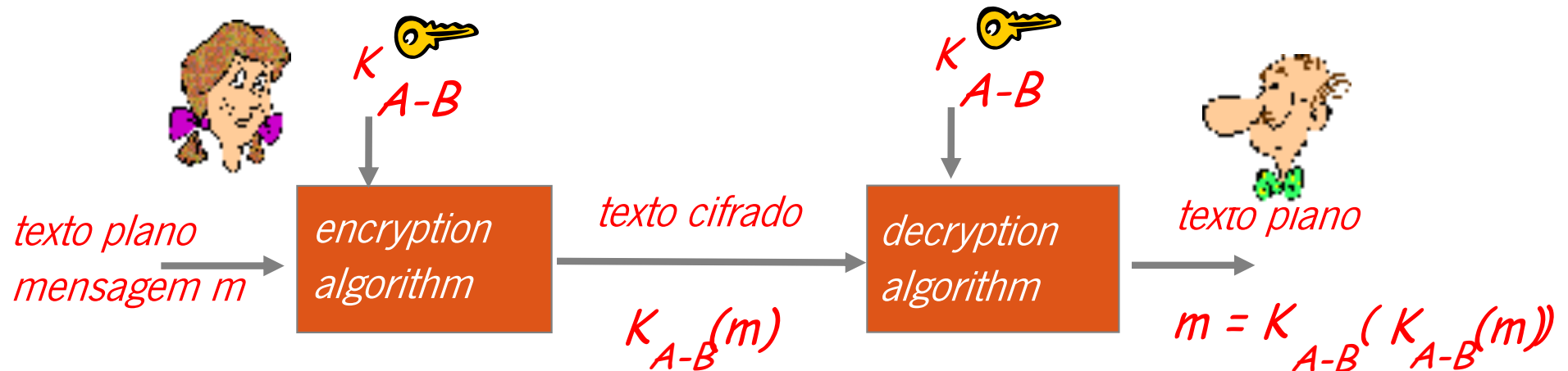
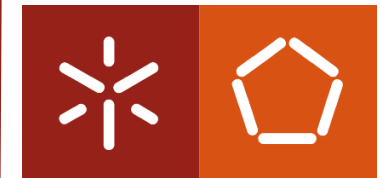
Ex.:   *Texto Plano: Alice, Amo-te. Bob.*

*Texto Cifrado: Mgsbc, Mhk-nc. Nkn.*

Q: Será fácil ou difícil quebrar esta cifra?

- *Pela força bruta (difícil?)*
- *Outro método?*

# Criptografia de chave simétrica



**Chave simétrica:** Alice e Bob conhecem a mesma chave (simétrica)  $K_{A-B}$

- Ex: conhecem o padrão de substituição do alfabeto! (ou a máquina de escrever, como a famosa **Enigma** da 2ª guerra mundial)
- **Pergunta:** Como podem eles combinar a chave?



# Algoritmos mais usados

- **DES – Data Encryption Standard**

- Chaves de 56 bits que processam blocos de 64 bits de cada vez
- Quebra-se por força bruta em **menos de 1 dia!**

- **3-DES ( 3 x DES )**

- Usa 3 chaves DES sequencialmente:  $ciphertext = E_{K3}(D_{K2}(E_{K1}(plaintext)))$
- Prevê-se que possa ser usado até ao ano 2030..

- **AES – Advanced Encryption Standard**

- Veio em 2001 para substituir o velho DES
- Processa blocos de 128 bits de cada vez
- Chaves de 128, 192 ou 256 bits de tamanho
- Pela força bruta, o que demora um segundo a quebrar no DES demorará **149 triliões de anos** no AES!!!



# Criptografia de Chave Pública



## Criptografia de chave simétrica

- exige que emissor e receptor conheçam a mesma chave secreta
- Pergunta: como podem combinar uma, se, por exemplo, não se conhecem ou nunca estiveram juntos?

## Criptografia de Chave Pública

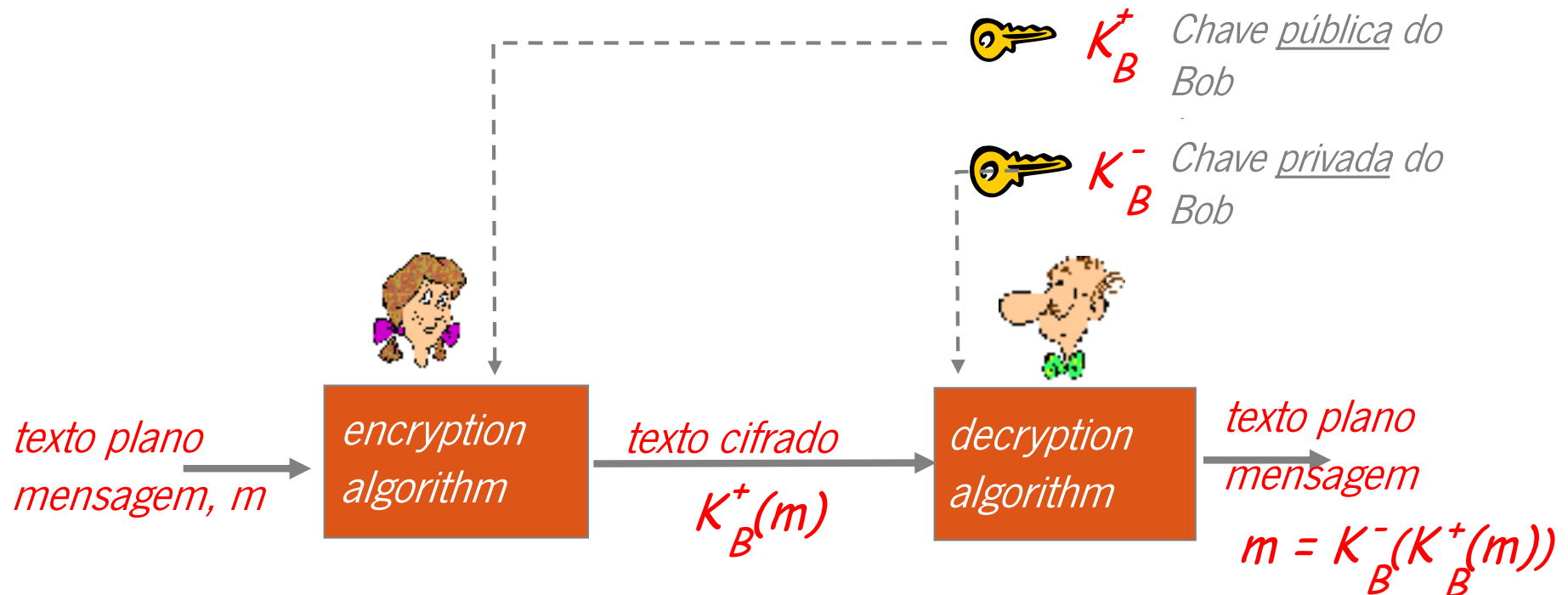
- ❑ abordagem radicalmente diferente  
*[Diffie-Hellman76, RSA78]*
- ❑ emissor e receptor não partilham nenhum segredo!
- ❑ Usa um par de chaves
- ❑ *Chave pública* conhecida por todos
- ❑ *Chave Privada* apenas conhecida pelo recetor



# Criptografia de Chave Pública



## Confidencialidade, e também integridade



Só o Bob, na posse da sua chave privada, poderá decifrar a mensagem  
Mais ninguém pode fazê-lo – total confidencialidade!  
Se não decifrar é porque não mantém a integridade

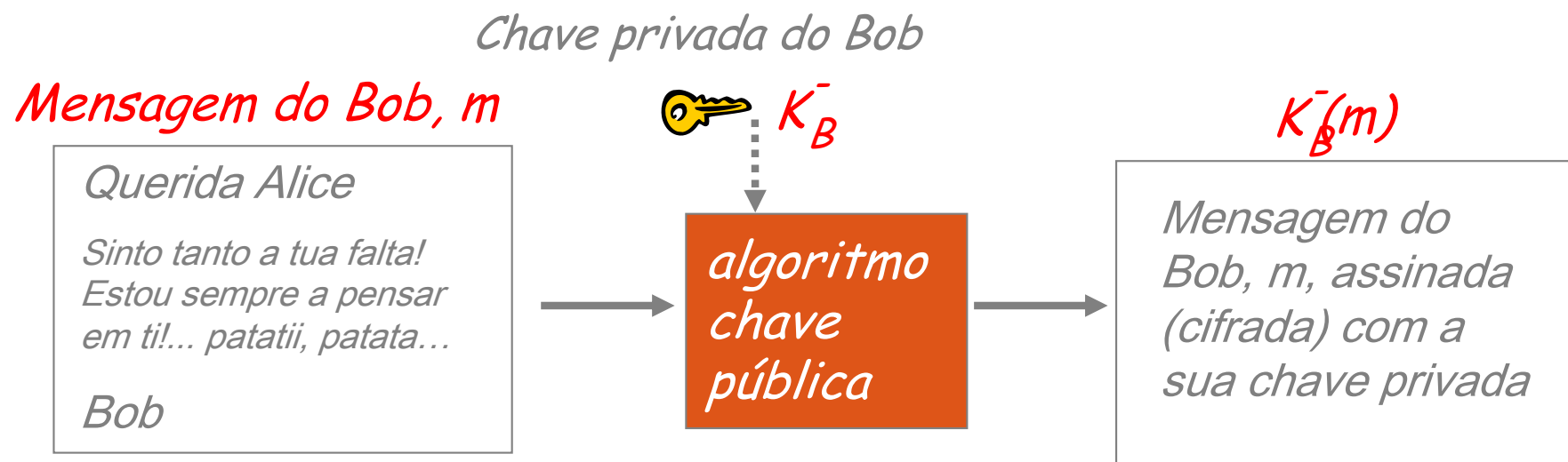
# Criptografia de Chave Pública



**Autenticação do originador (assinatura digital),  
não repúdio do originador e também integridade**

**esquema muito simples para assinar a mensagem  $m$ :**

- Não usado (por questões de desempenho)
- Bob “assina”  $m$  cifrando-a com a sua chave privada, criando assim uma mensagem assinada  $K_B(m)$



# Criptografia de Chave Pública



*Requisitos:*

- ① *necessário um par de chaves tais que*

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

- ② **deverá ser impossível obter a chave privada a partir da chave pública!**

***RSA:** Algoritmo Rivest, Shamir, Adleman*

# Criptografia de Chave Pública



A seguinte propriedade é  **muito** útil:

$$\underbrace{K_B^-(K_B^+(m))}_{\text{Usar a chave pública e depois a privada}} = m = \underbrace{K_B^+(K_B^-(m))}_{\text{Usar a chave privada e depois a pública}}$$

*Usar a chave pública  
e depois a privada*

*Usar a chave privada  
e depois a pública*

**O resultado é o mesmo!**

# Integridade e Autenticação da Origem

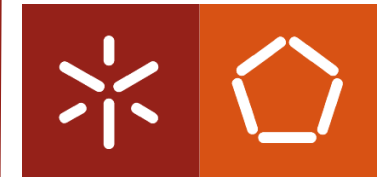


Bob recebe uma mensagem da Alice, e quer garantir que:

- a mensagem veio originalmente da Alice
- a mensagem não foi alterada (mantém-se íntegra) desde que foi enviada pela Alice até ser lida

## Função de sumariação (Hash):

- dada uma mensagem de entrada  $m$ , produz um sumário de tamanho fixo,  $H(m)$ 
  - Ex: *checksum* (soma de verificação)
- é computacionalmente improvável encontrar duas mensagens diferentes  $x, y$  tais que  $H(x) = H(y)$ 
  - de igual modo: dado um  $m = H(x)$ , (com  $x$  desconhecido), não se consegue determinar o  $x$  a partir do  $m$ .
  - Nota: isto não é verdade para o *checksum*!



# Algoritmos mais usados

---

- **MD5 – Message Digest**

- Calcula sumários de 128 bits em 4 passos
- ataques ao MD5 em 2005 mostram que talvez não seja adequado

- **SHA-1 – Secure Hash Algorithm**

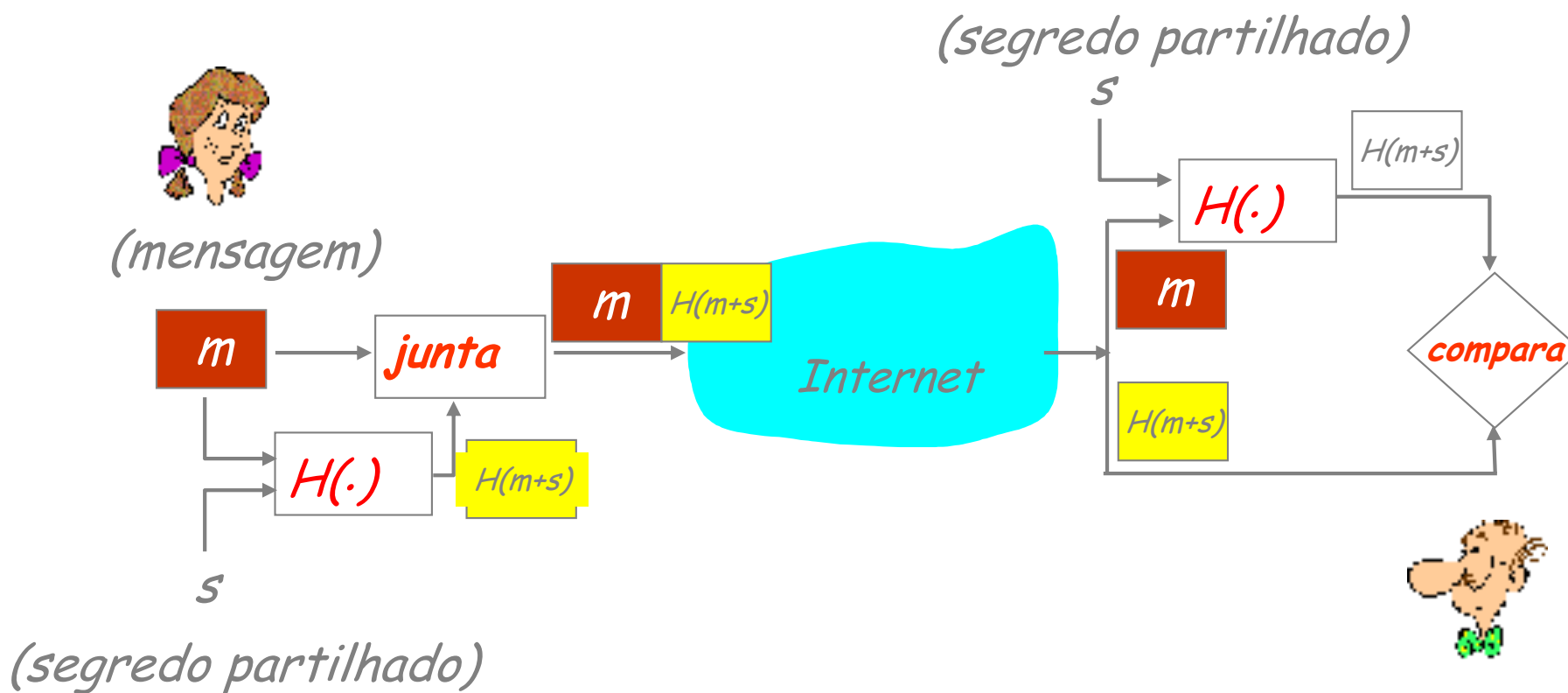
- Calcula sumários de 160 bits

# Integridade e Autenticação da Origem



## MAC – Message Authentication Code

Envia o sumário da mensagem e do segredo juntos ( $m+s$ )





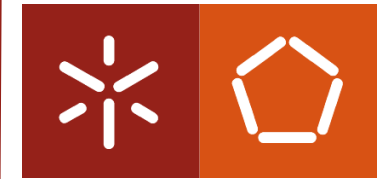
# Assinatura digital



*Integridade, autenticação de origem e não repúdio do originador*

**Uma técnica criptográfica muito semelhante à assinatura manual.**

- o emissor (Bob) assina digitalmente o documento provando que é o dono/criador do mesmo (não pode negar mais tarde!)
- **verificável, não forjável:** o receptor (Alice) consegue provar a qualquer um que foi o Bob – e não poderia ter sido mais ninguém, nem mesmo a própria Alice – que assinou o documento ou mensagem



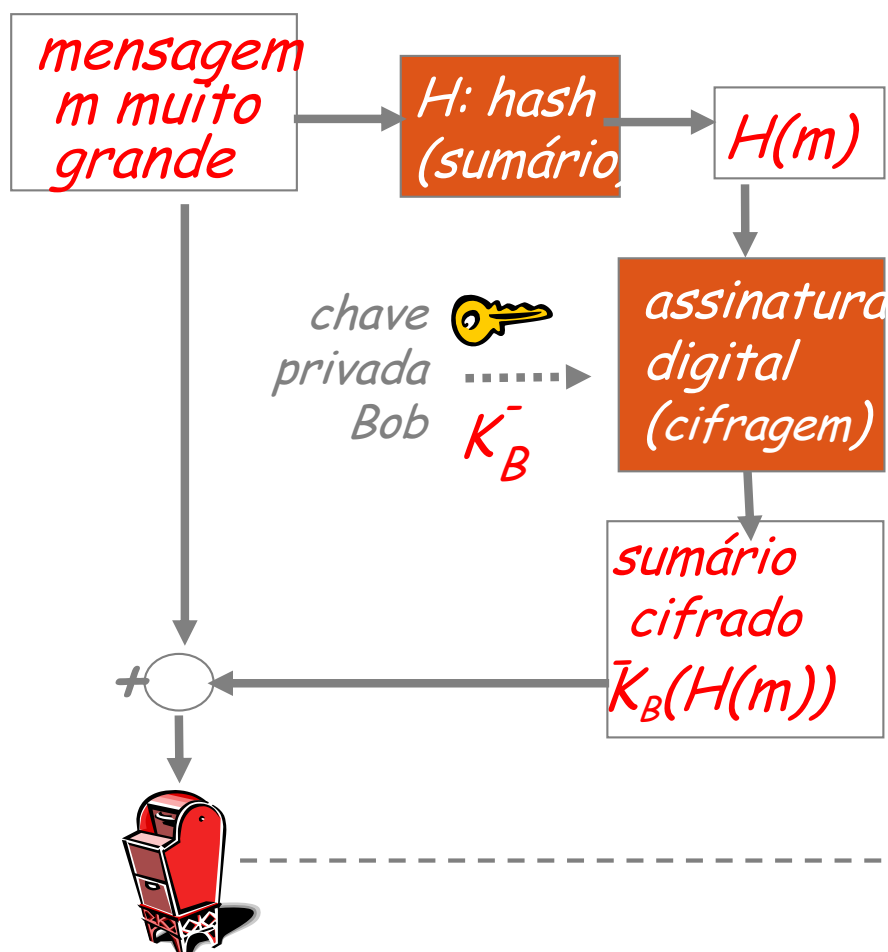
## ● Garantias

- Só o Bob pode ter assinado  $m$ , pois só ele conhece a sua chave privada
- Mais ninguém poderia ter assinado  $m$
- A mensagem que foi assinada foi  $m$  e não um  $m'$  qualquer
- Qualquer um pode verificar isso: basta pegar na chave pública de Bob e decifrar a assinatura
- Garante ainda o não repúdio – mesmo em tribunal! – pois Bob não poderá negar ter usado a sua chave privada

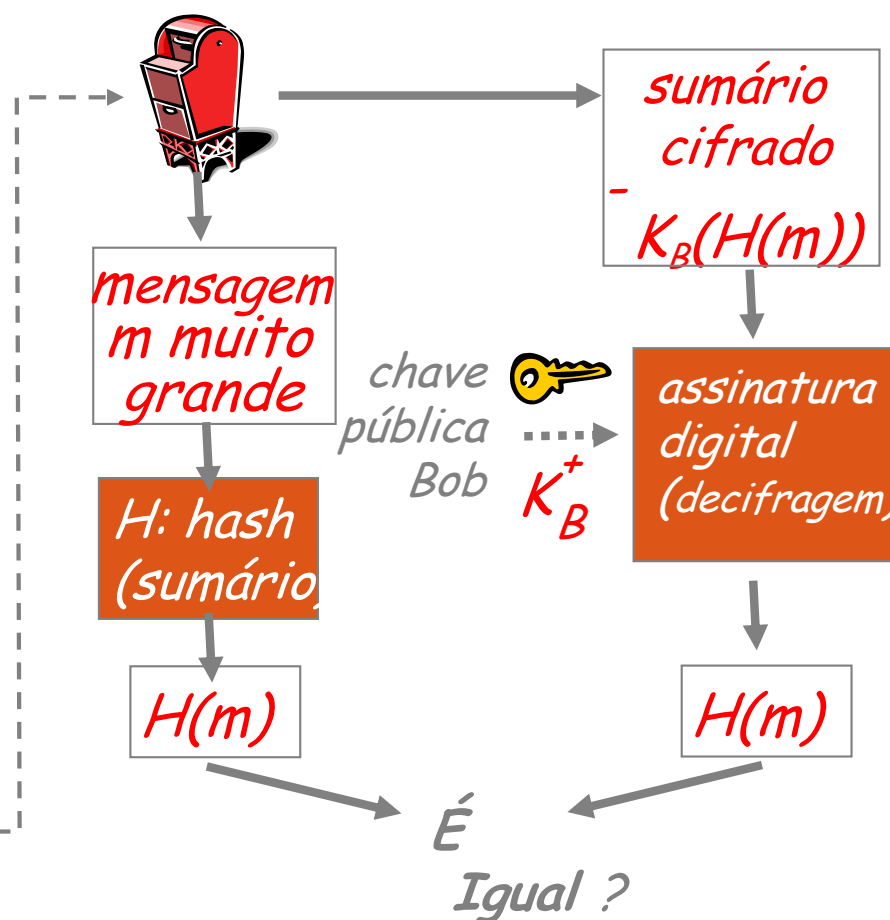
# Assinatura Digital (2)



Bob envia mensagem assinada:



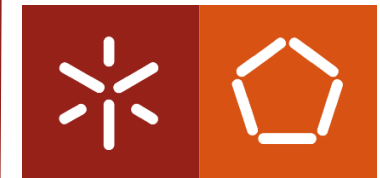
Alice verifica a assinatura:



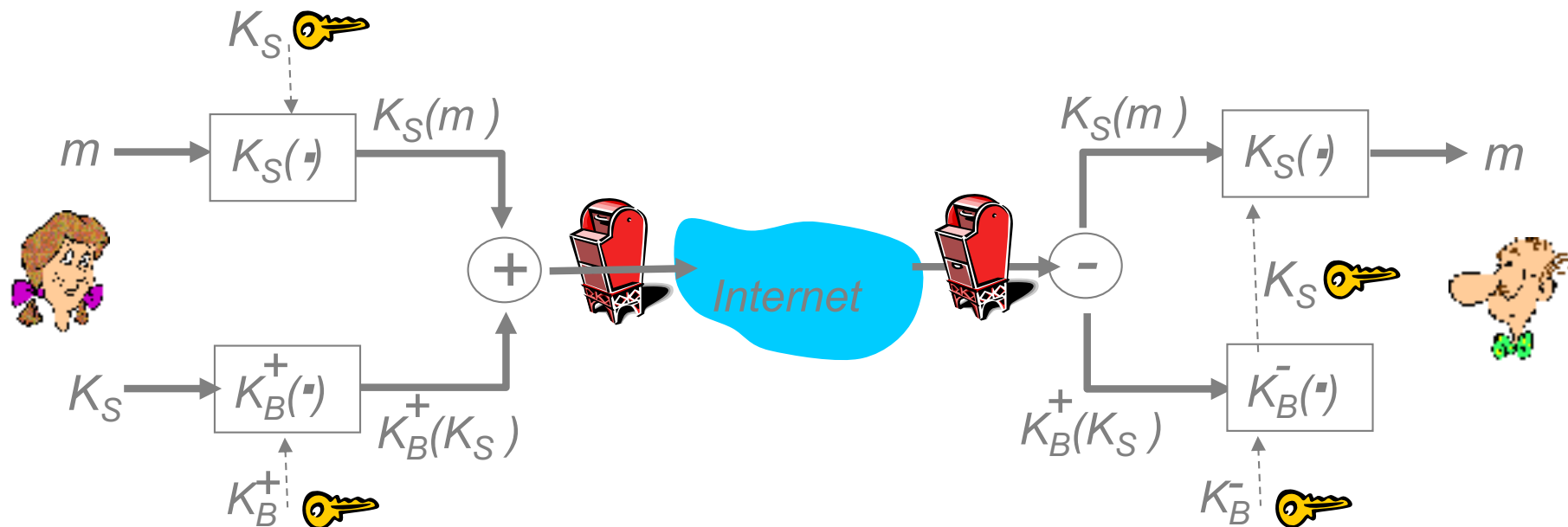


- **Usa criptografia de chave pública, que propriedades tem a seguinte comunicação:**
  - Mensagem cifrada com a chave pública do originador
  - Mensagem cifrada com a chave pública do destinatário
  - Mensagem cifrada com a chave privada do originador
  - Mensagem cifrada com a chave privada do destinatário
- **Comente a seguinte afirmação: "A assinatura digital associa o assinante ao documento assinado, garantindo integridade e não repúdio."**

# Envelope Digital



*Alice envia mensagem confidencial para Bob ("envelope digital" selado)*



*Alice:*

- Gera uma chave simétrica secreta,  $K_S$
- Cifra mensagem com  $K_S$  (por questões de eficiência)
- Cifra também a chave simétrica secreta  $K_S$  com a chave pública de Bob
- Envia ambos:  $K_S(m)$  e  $K_B(K_S)$  para Bob

# Infra-estrutura de chaves públicas (PKI)



## Problema Chaves Simétricas:

- Como é que duas entidades estabelecem um segredo (a chave secreta) usando apenas a rede?

### **Solução:**

- Centro de distribuição de chaves que seja de confiança e actua como intermediário entre as entidades

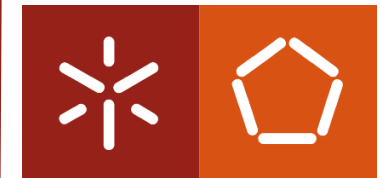
## Problema Chaves Públicas

- Quando se obtém a chave pública da Alice ou do Bob na rede (e-mail, web, etc) como sabemos que são mesmo deles e não do intruso?

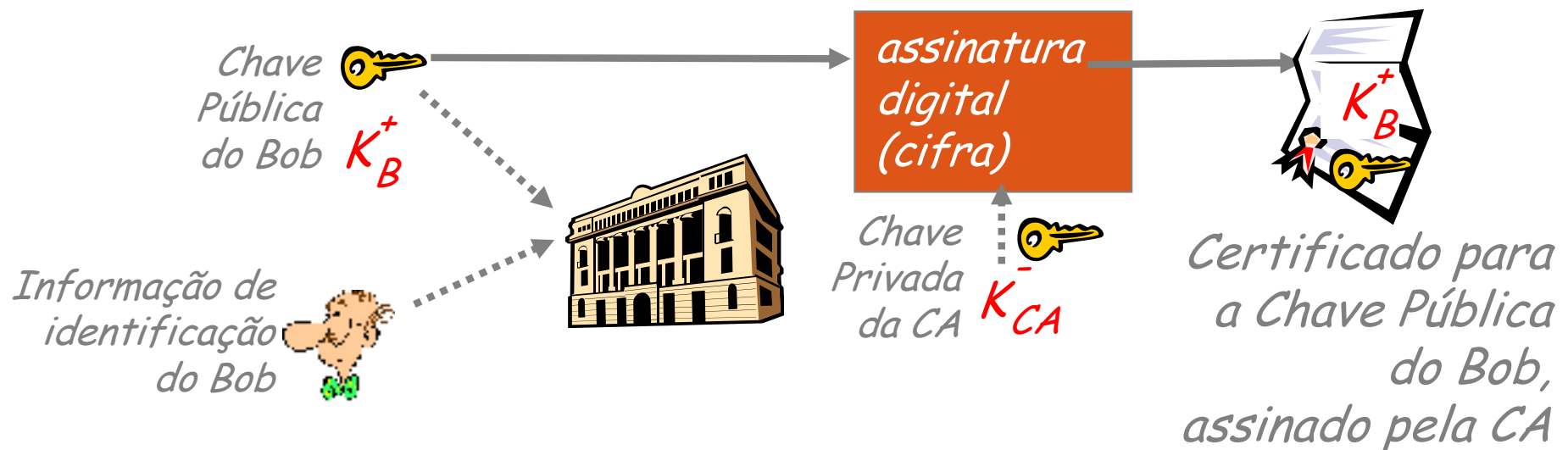
### **Solução:**

- Autoridade de Certificação (CA) de confiança (*trusted certification authority*)

# Autoridades de Certificação



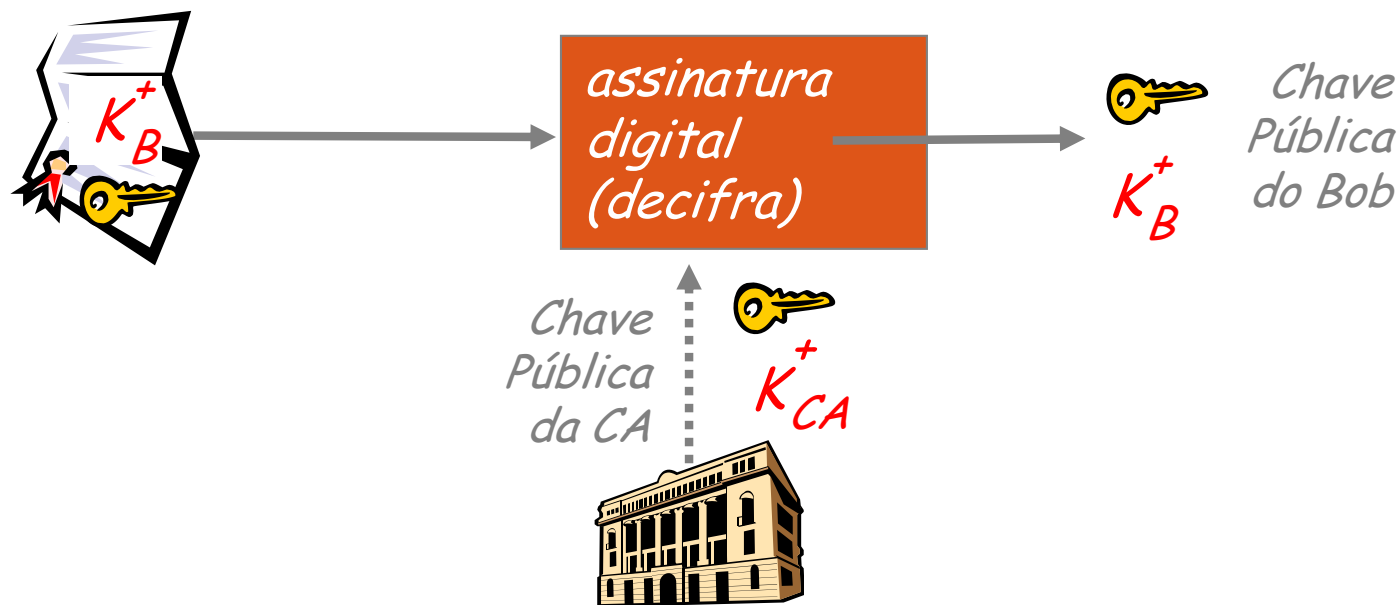
- *Autoridade de Certificação (CA): associa a chave pública a uma determinada entidade, E*
- *E (pessoa, máquina,...) regista a sua chave pública na CA*
  - *E tem de fornecer uma provada de identidade a CA*
  - *CA cria um certificado digital associando E à sua chave pública*
  - *certificado contém a chave pública de E assinada digitalmente pela CA que assim assegura que “esta é a chave pública de E”*



# Autoridades de Certificação



- Quando a Alice quer obter a chave pública do Bob:
  - *obtem o certificado do Bob (dele mesmo ou doutros sítios).*
  - *aplica a chave pública da CA ao certificado para verificar a validade do certificado e extrair de lá a chave pública do Bob*





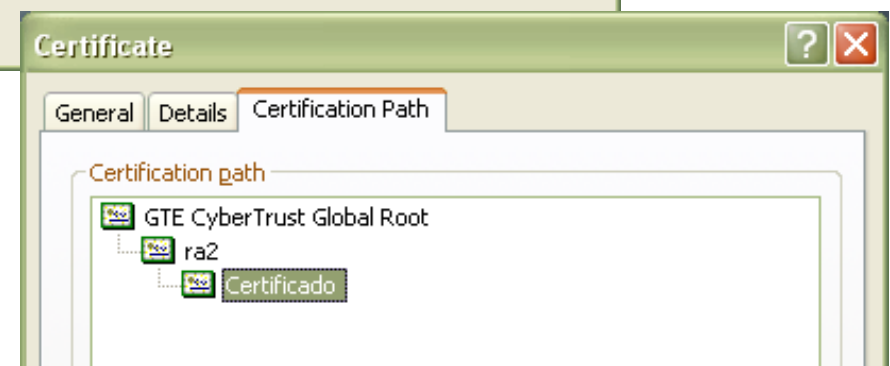
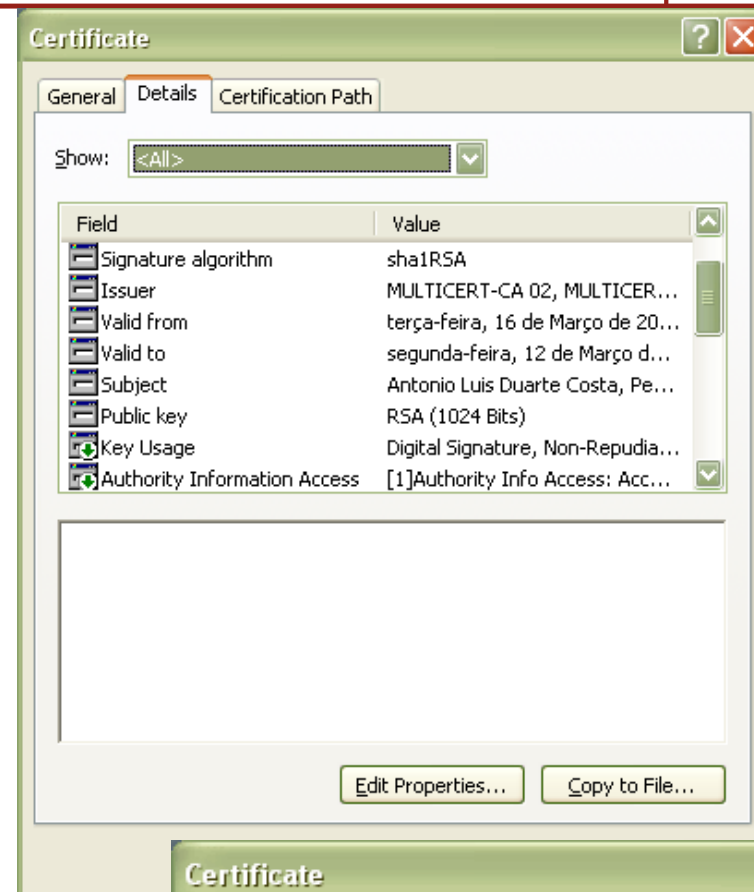
# Autoridades de Certificação



- *Problema: como confiar no CA?*
  - *A mesma coisa?... mas?... problema!*
  - *Certificados de raiz (root certificates) instalados com as máquinas (Windows, Linux, ou seja lá o que for)*
    - *Esse é o momento decisivo para a criptografia de chave pública*



# Exemplo de um Certificado



# Exemplo de uso – E-Mail



*S/MIME (Ex: OpenSSL)*  
→ Faz uso de PKI

*PGP (Ex: OpenPGP) Pretty Good Privacy*  
→ Não faz uso de PKI

## ● Fazer demo com E-mail

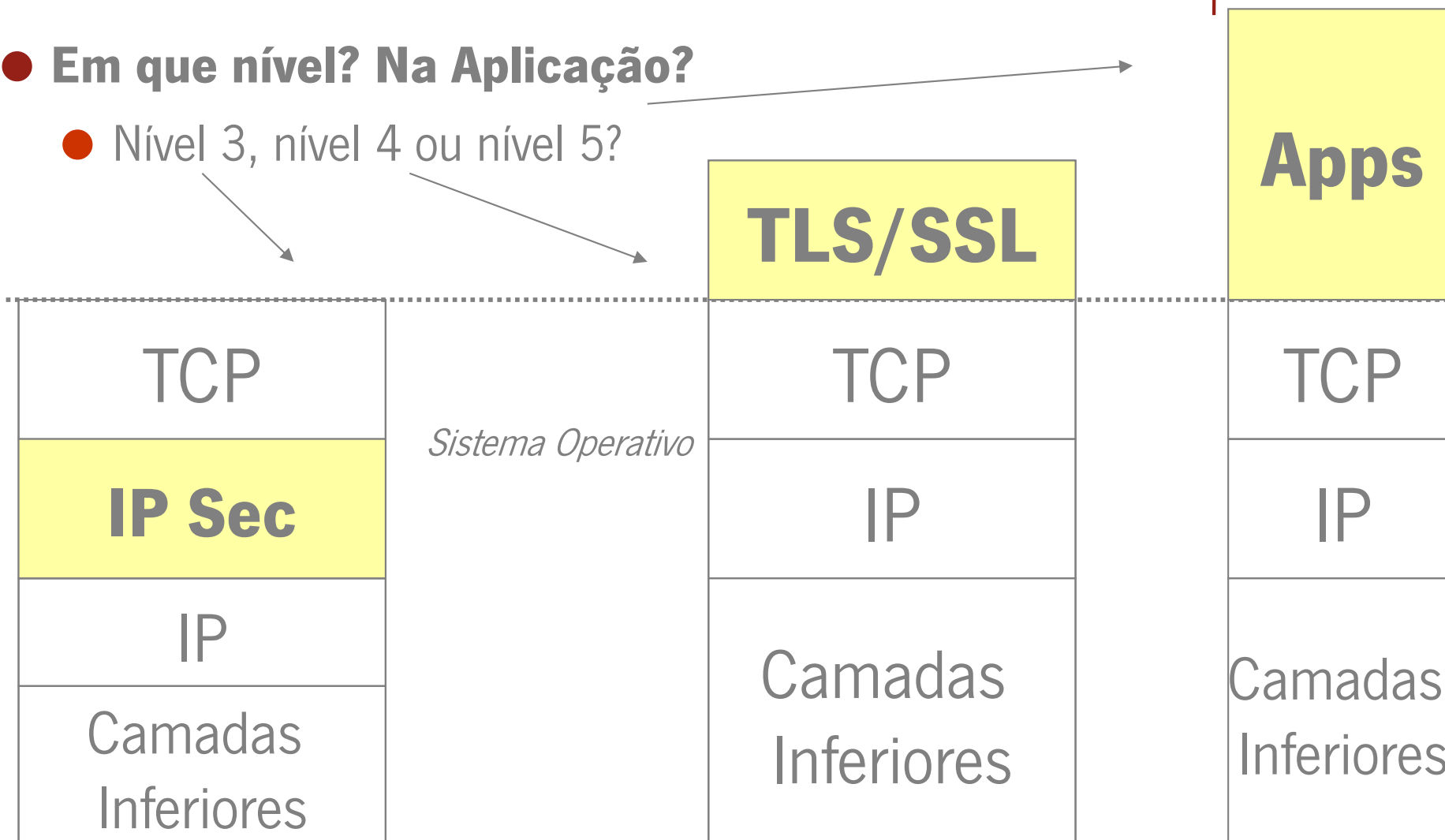
1. Adicionar um certificado válido ao cliente E-mail
2. Consultar certificados de terceiros...
3. Assinar e/ou “cifrar” uma mensagem...
4. Verificar a assinatura

# Segurança



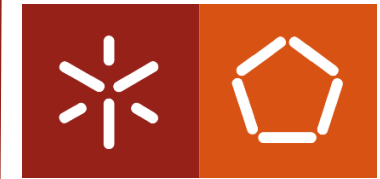
- **Em que nível? Na Aplicação?**

- Nível 3, nível 4 ou nível 5?



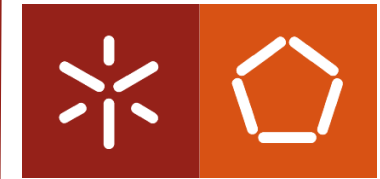


- **A nível 4 (SSL/TLS) as aplicações fazem interface com SSL e não com o TCP:**
  - Como o TCP não participa em nada, não consegue discernir pacotes inseridos maliciosamente na stream, desde que estejam correctos (checksum) e passa-os ao SSL...
- **A nível 3, as aplicações continuam a interagir com o TCP:**
  - As aplicações não precisam ser modificadas
  - Mas o nível IP só sabe com que IP está a trocar dados e não com que utilizador...
- **Também é possível a nível 5 (aplicação), mantendo compatibilidade com aplicações existentes:**
  - Exemplo: PGP ou S/MIME (compatível MAIL) sobre SMTP
  - Soluções específicas para uma dada aplicação...



- **Nível 4: Secure Sockets Layer (SSL)**

- Segurança ao nível de transporte para qualquer aplicação TCP
- Usado, por exemplo, no acesso a servidores HTTP, IMAP, SMTP
- Serviços de segurança:
  - Autenticação do servidor e, opcionalmente, do cliente
  - Confidencialidade dos dados
- **Autenticação do servidor:**
  - Cliente SSL conhece chaves públicas de autoridades de certificação de sua confiança (CA)
  - Obtem certificado do servidor emitido por uma CA sua conhecida
  - Extrai chave pública do certificado depois de verificada validade



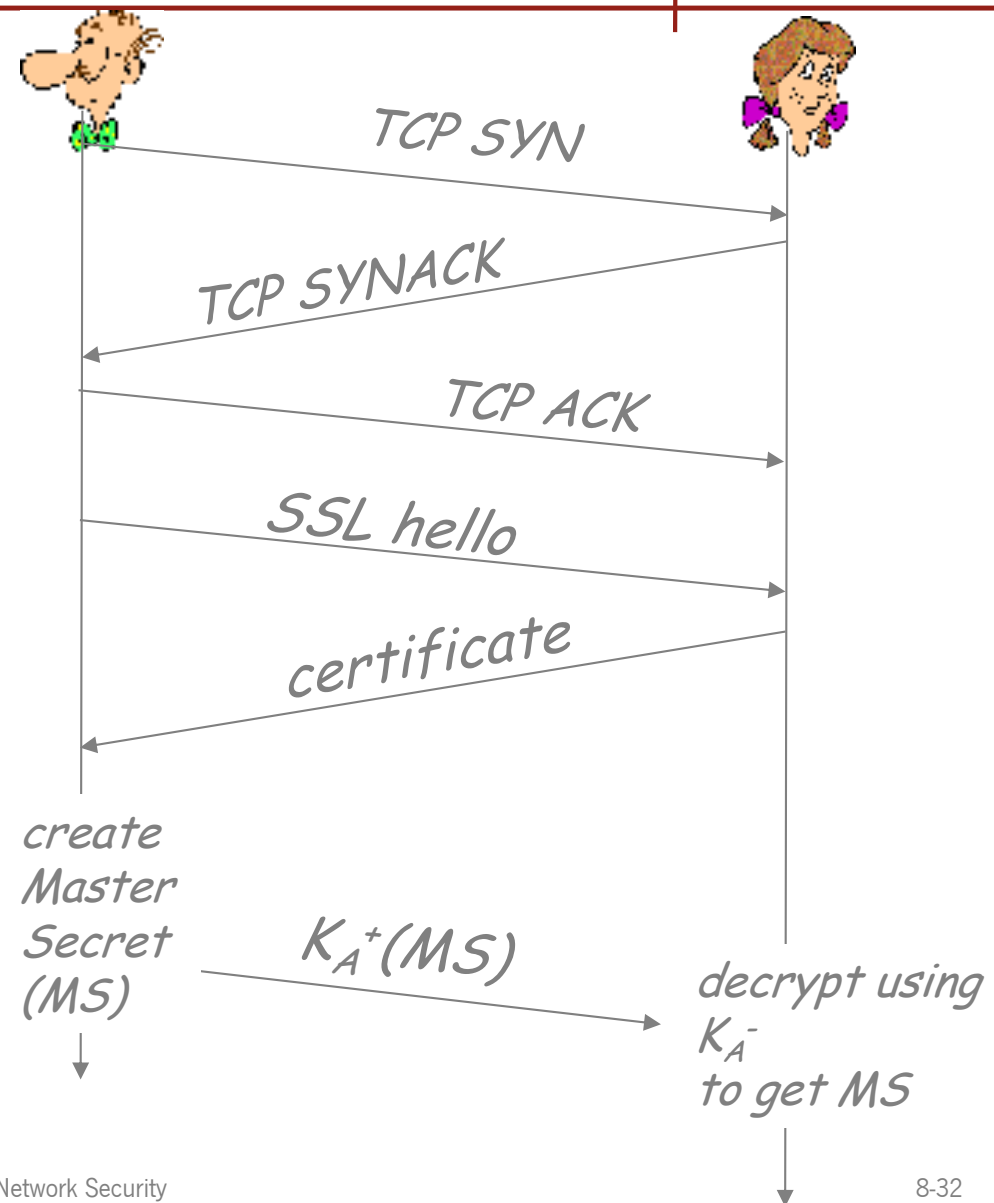
- **Nível 4: Secure Sockets Layer (SSL)**
  - **Confidencialidade (cifragem dos dados da sessão)**
    - Cliente SSL gera chave de sessão, cifra-a com a chave pública do servidor e envia-a ao servidor
    - Servidor decifra a chave de sessão usando a sua chave privada
    - Ambos – cliente e servidor – na posse da chave de sessão, podem cifrar todos os dados trocados...
  - **Autenticação do cliente** pode ser feita com base em certificados do cliente
- **SSL serviu de base ao TLS (Transport Layer Security) do IETF**
  - As diferenças não são significativas, mas suficientes para impedir a interoperabilidade

# Exemplo SSL: 3 fases



## 1. Handshake inicial:

- Bob estabelece conexão TCP com Alice
- Autentica Alice usando o certificado assinado por uma CA
- Cria chave mestra, encripta-a (usando a chave pública da Alice), e envia-a à Alice
  - Incompleto: a troca de um “nonce” não está ilustrada aqui!!





# Exemplo SSL: 3 fases



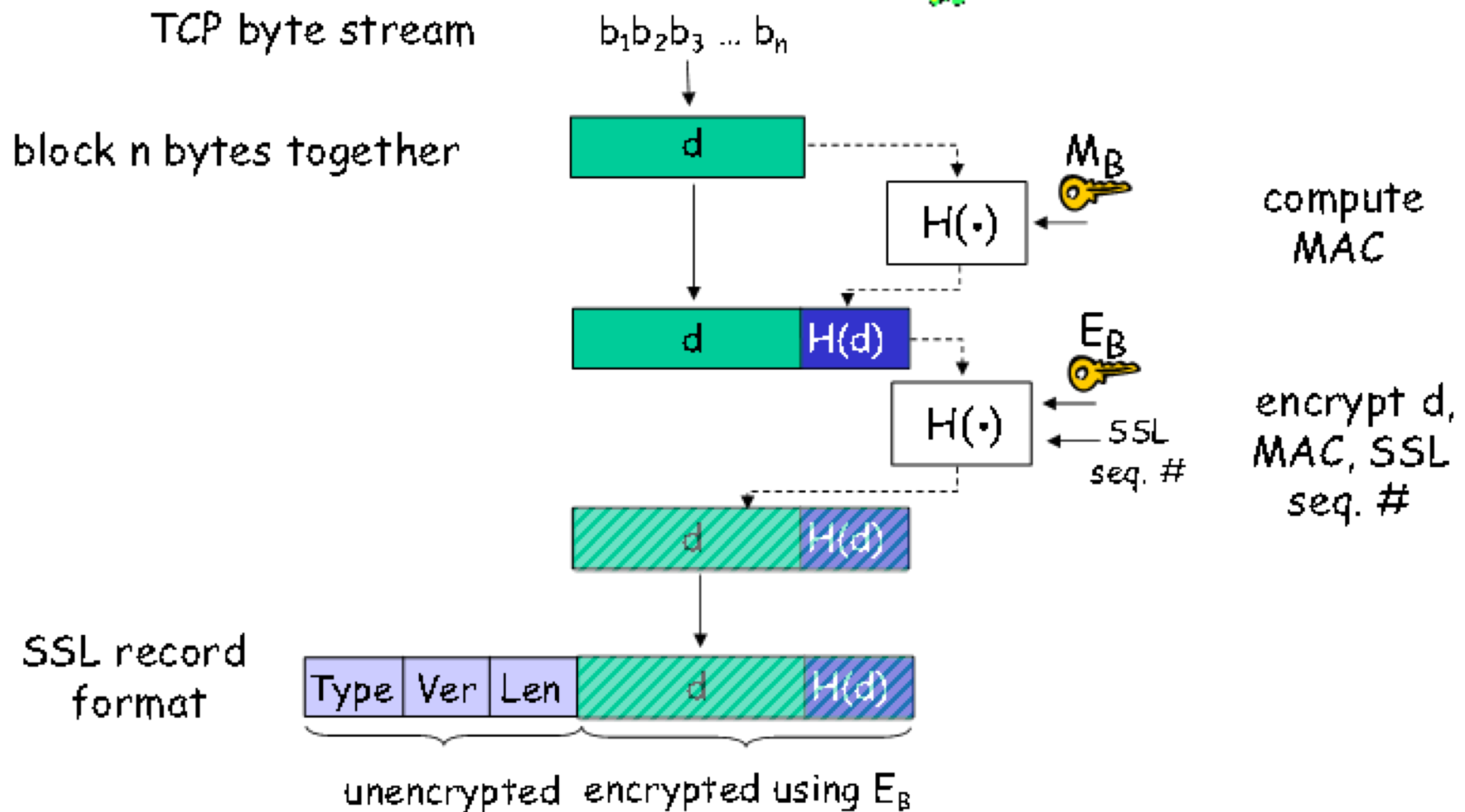
## 2. Cálculo das chaves:

- Alice e Bob usam a chave mestra (MS) para gerar 4 chaves:
  - $E_B$ : Bob→Alice chave de cifragem de dados
  - $E_A$ : Alice→Bob chave de cifragem de dados
  - $M_B$ : Bob→Alice chave MAC (*Message Authentication Code*)
  - $M_A$ : Alice→Bob chave MAC (*Message Authentication Code*)
- Os algoritmos (cifragem e MAC) são negociados entre a Alice e o Bob
- Porquê 4 chaves?

# Exemplo SSL: 3 fases



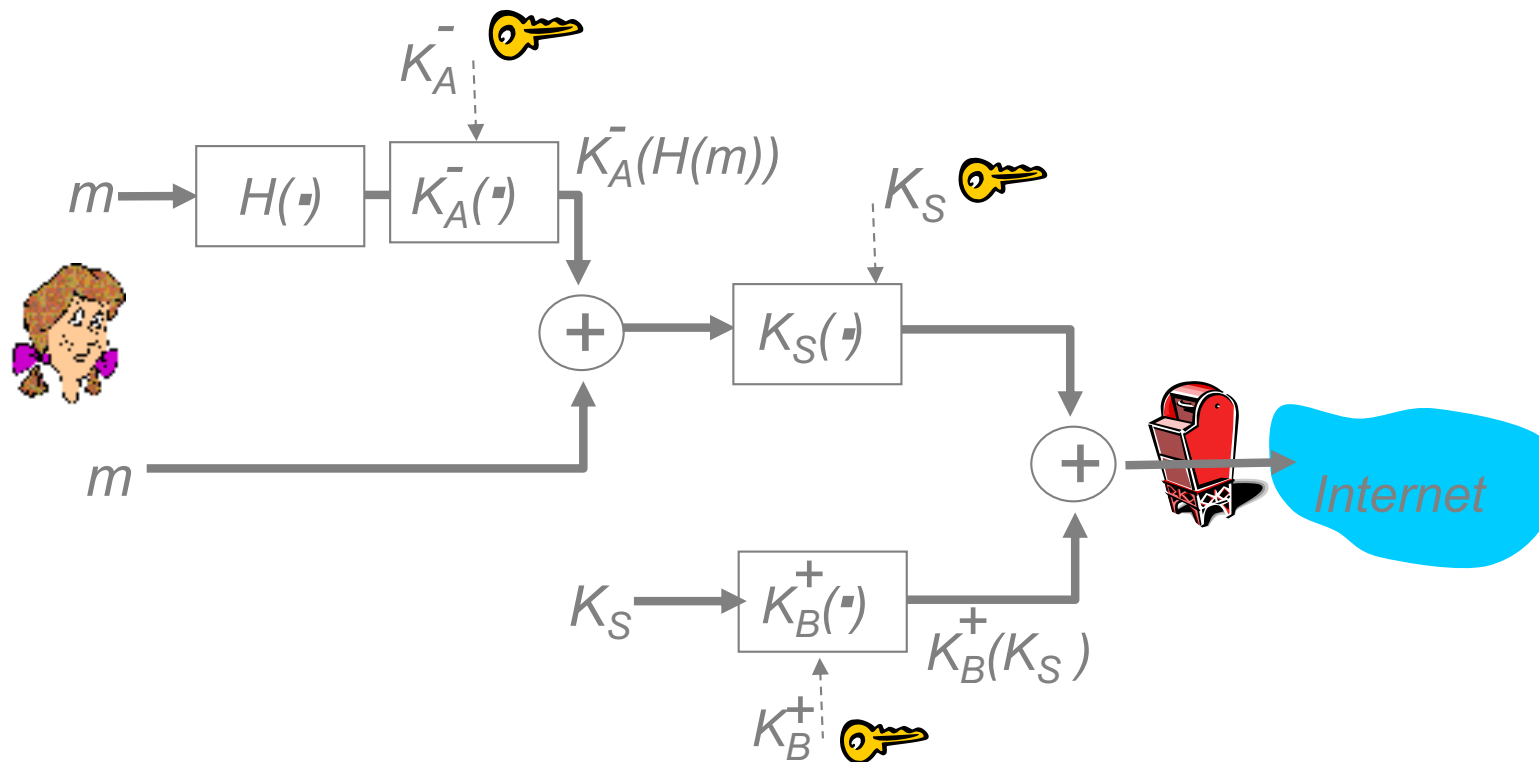
## 3. Transferência dados



# E-Mail seguro (assinado e cifrado)



*Para dar todas as garantias: confidencialidade, integridade, autenticação e não repúdio do originador*



*Alice usa três chaves: a sua chave privada, a chave pública do Bob, uma chave simétrica secreta gerada no momento*

# Segurança IP (IPSec)



- **Segurança no nível 3: IPsec**

- Confidencialidade: host que origina o pacote IP cifra os dados nele contidos (inclui segmentos TCP, datagramas UDP, ICMP, etc..)
- Autenticação de origem e integridade: host de destino consegue validar autenticidade do IP de origem

- **Dois protocolos:**

- Authentication Header (AH) – autenticação de origem e integridade
- Encapsulation Security Payload (ESP) – ou só confidencialidade ou confidencialidade + autenticação de origem

- **Obrigatórios nas implementações IPv6**

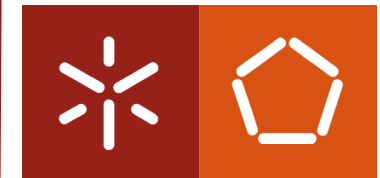
- **Opcionais nas implementações IPV4**

# IPSec: usando as opções do IPv6

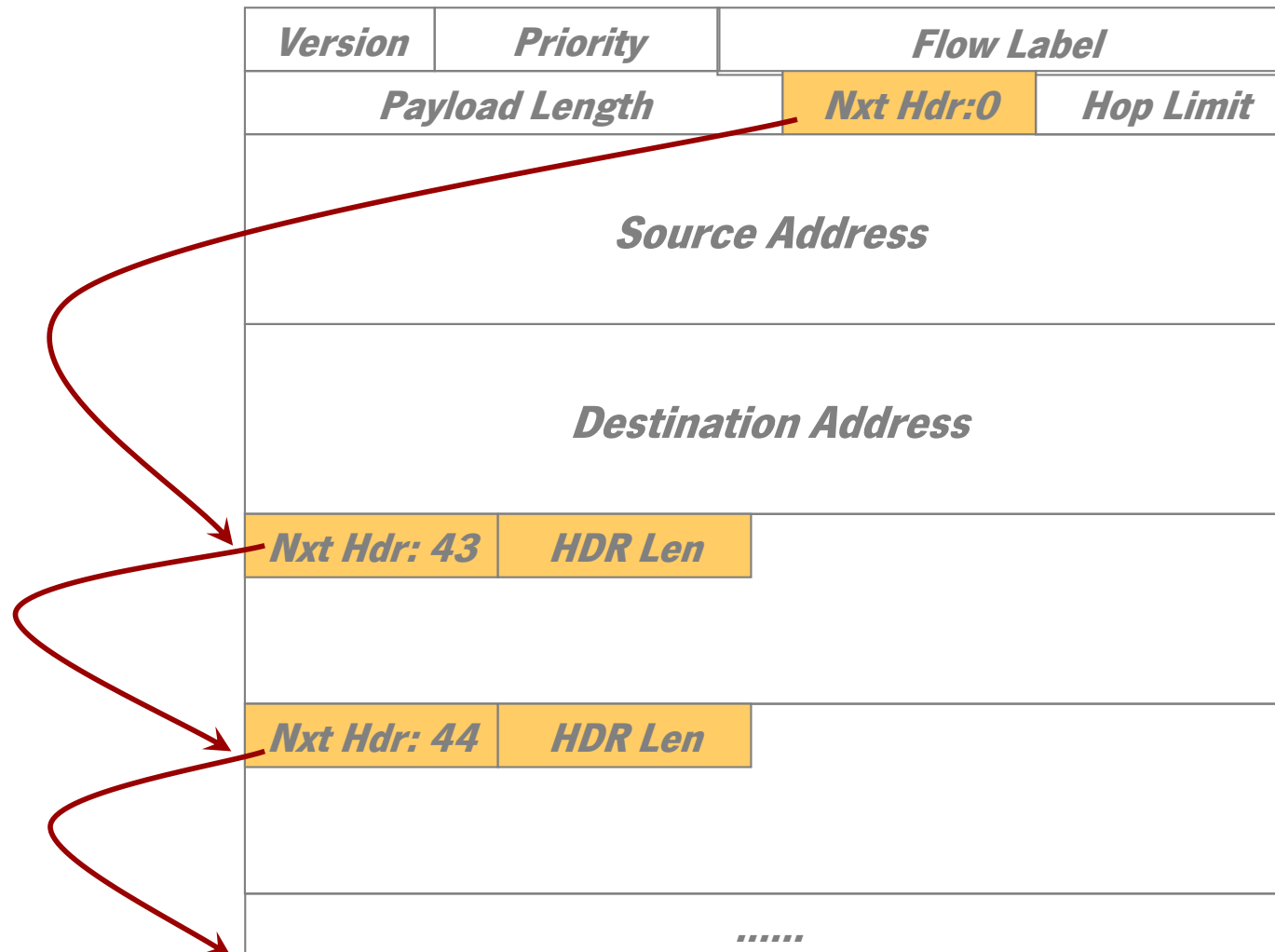


## ● Extensões ao cabeçalho (a ordem é importante):

- 0 - hop-by-hop Option Header
- 43 - Routing Header
- 44 - Fragmentation Header
- **51 - Authentication Header**
- **50 - Encapsulation Security Payload Header**
- 59 - No Next Header
- 60 - Destination Options Header
- 135 – Mobility Header
- Camadas superiores: 6 – TCP, 17 – UDP, 58 – ICMPv6

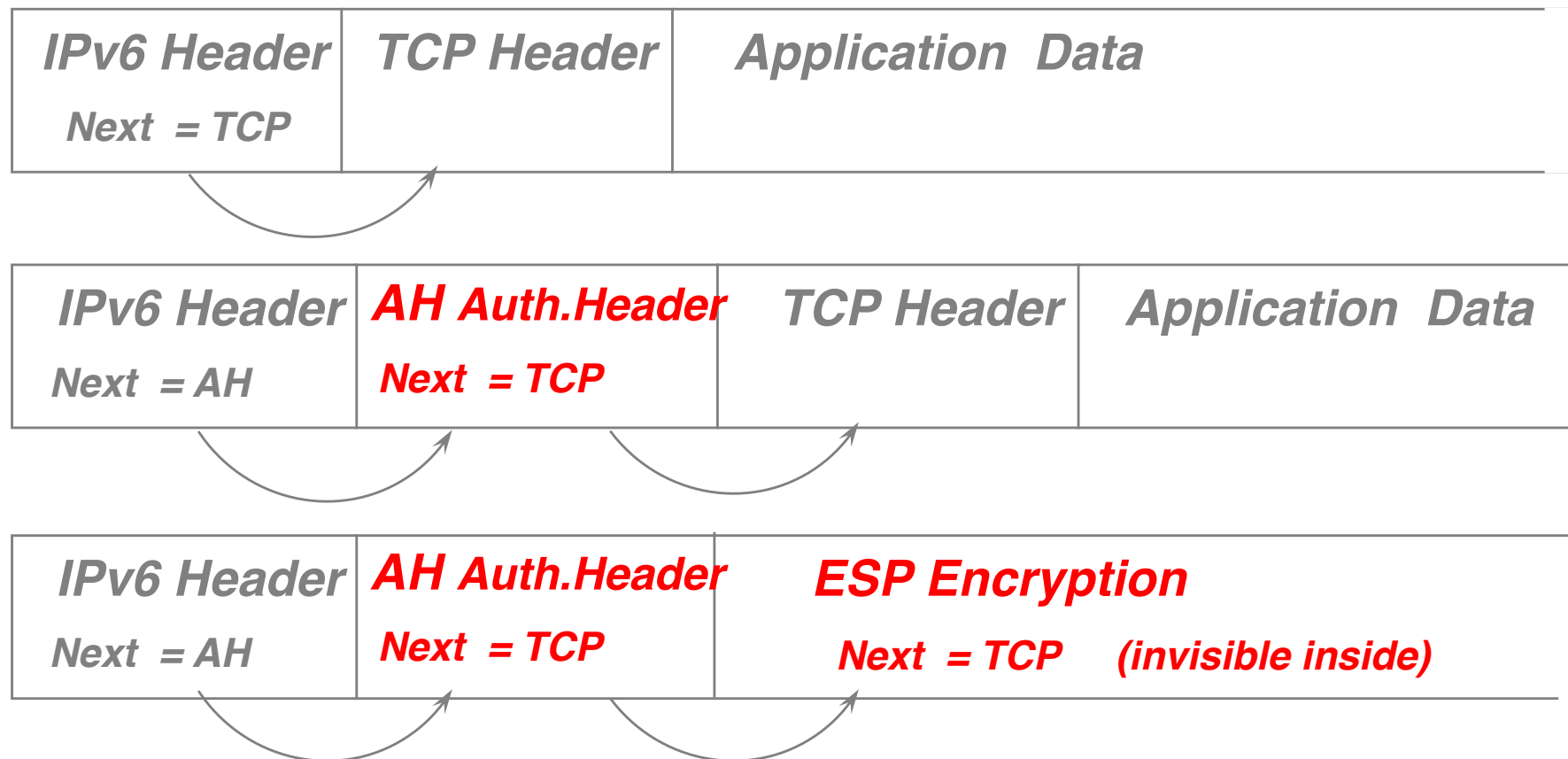
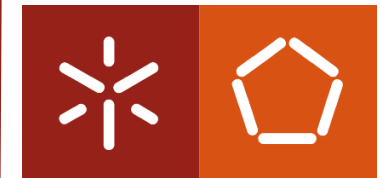


# Extensões ao cabeçalho IPv6



Fonte: Prof.a Ana Benso <http://www.inf.pucrs.br/~benso/>

# Extensões ao cabeçalho IPv6



# IPSec – Autenticação



- **Authentication Header (AH - Opções IPv6)**

Next Header	Payload Len	Reserved
Security Parameter Index (SPI)		
Sequence Number		
Authentication Data (variable Length)		





# IPSec – Autenticação

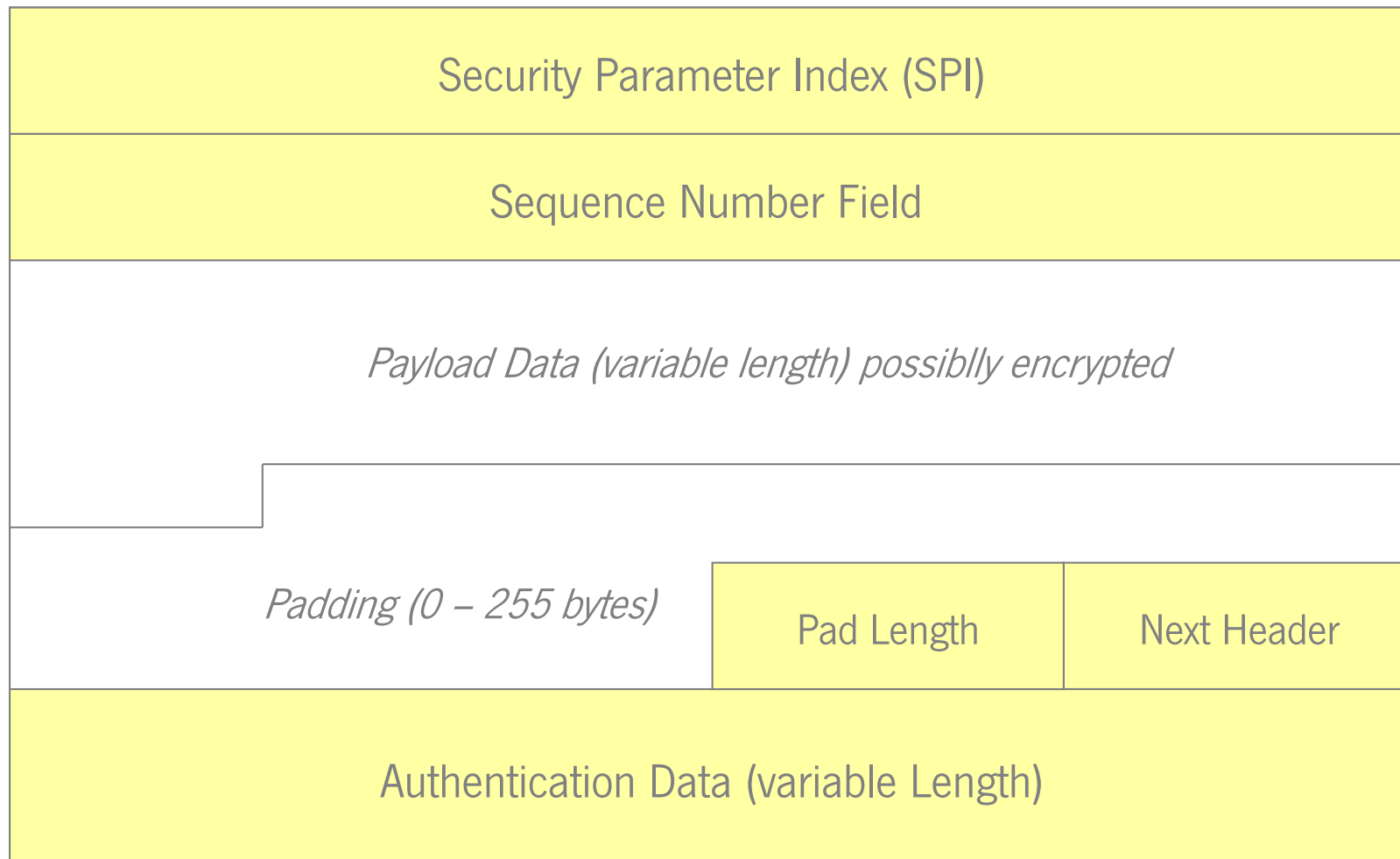
## ● Authentication Header (AH)

- O número de sequência (crescente) evita ataques por repetição
- Exemplo baseado no Message Digest 5:
  - Primeiro coloca-se a chave secreta com no mínimo 128 bits (identificável a partir do campo SPI)
  - De seguida coloca-se o datagrama IP completo...
  - ... Excepto, claro, o campo de autenticação propriamente dito e todos os campos do cabeçalho que podem ser modificados em trânsito ao longo do percurso (ex: Hop)
  - Após o datagrama IP acrescenta-se de novo a chave secreta
  - A mensagem assim construída é então sumariada com o algoritmo MD5;
  - Os 128 bits resultantes são colocados no campo “Authentication data” do cabeçalho de autenticação

- Garante integridade e autenticação de origem



- **Encryption Security Payload Header (ESP)**





# IPSec – Confidencialidade

## ● Encryption Security Payload Header (ESP – Opções IPv6)

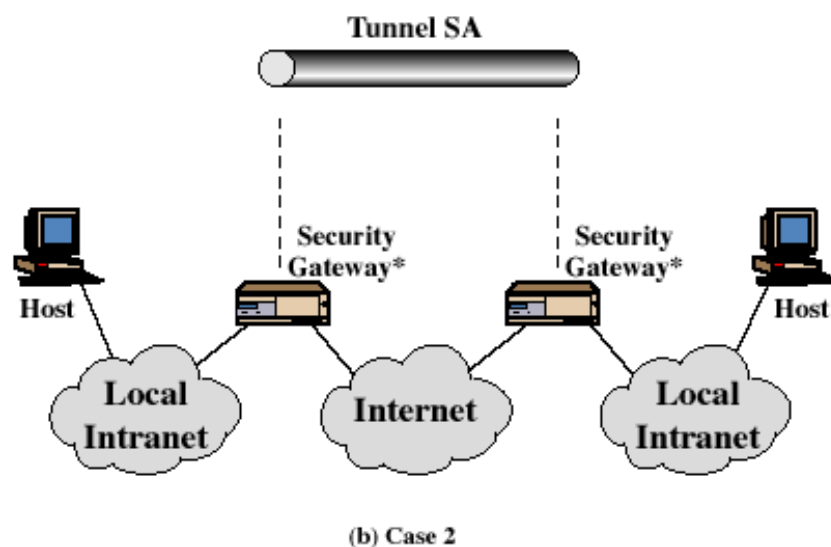
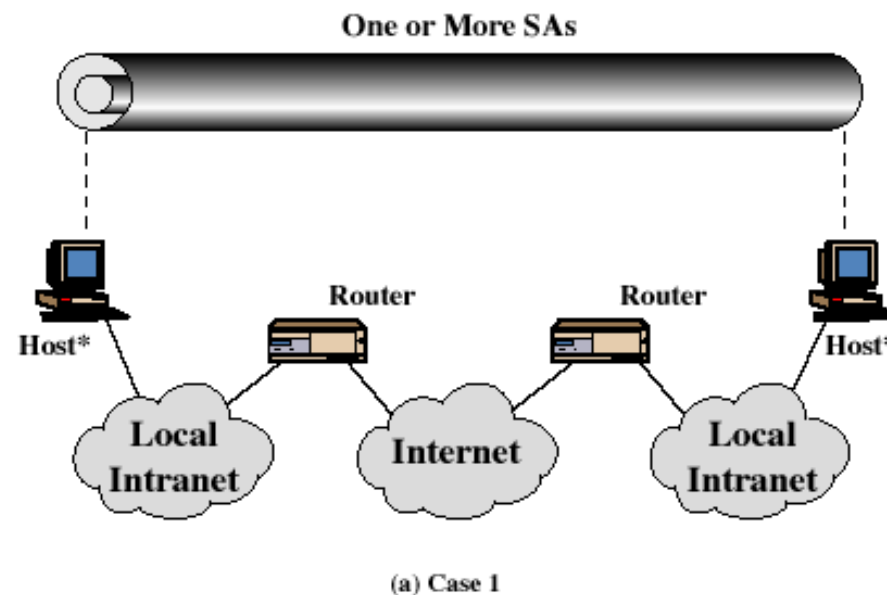
- O conteúdo cifrado não é observável nem modificável: garantia de integridade e também de confidencialidade
- Os campos *SPI*, *sequence number* e *authentication data* têm o mesmo significado que no cabeçalho de autenticação, mas aqui a autenticação é opcional;
- Só são cifrados (usando um determinado algoritmo) os dados deste cabeçalho em diante...
  - O cabeçalho básico IPv6 não é coberto – nem poderia! Cifrar endereços de origem e de destino? Como encaminhar então?

# Segurança



- Conexões entre hosts podem ser feitas em qualquer dos modos (túnel ou transporte):

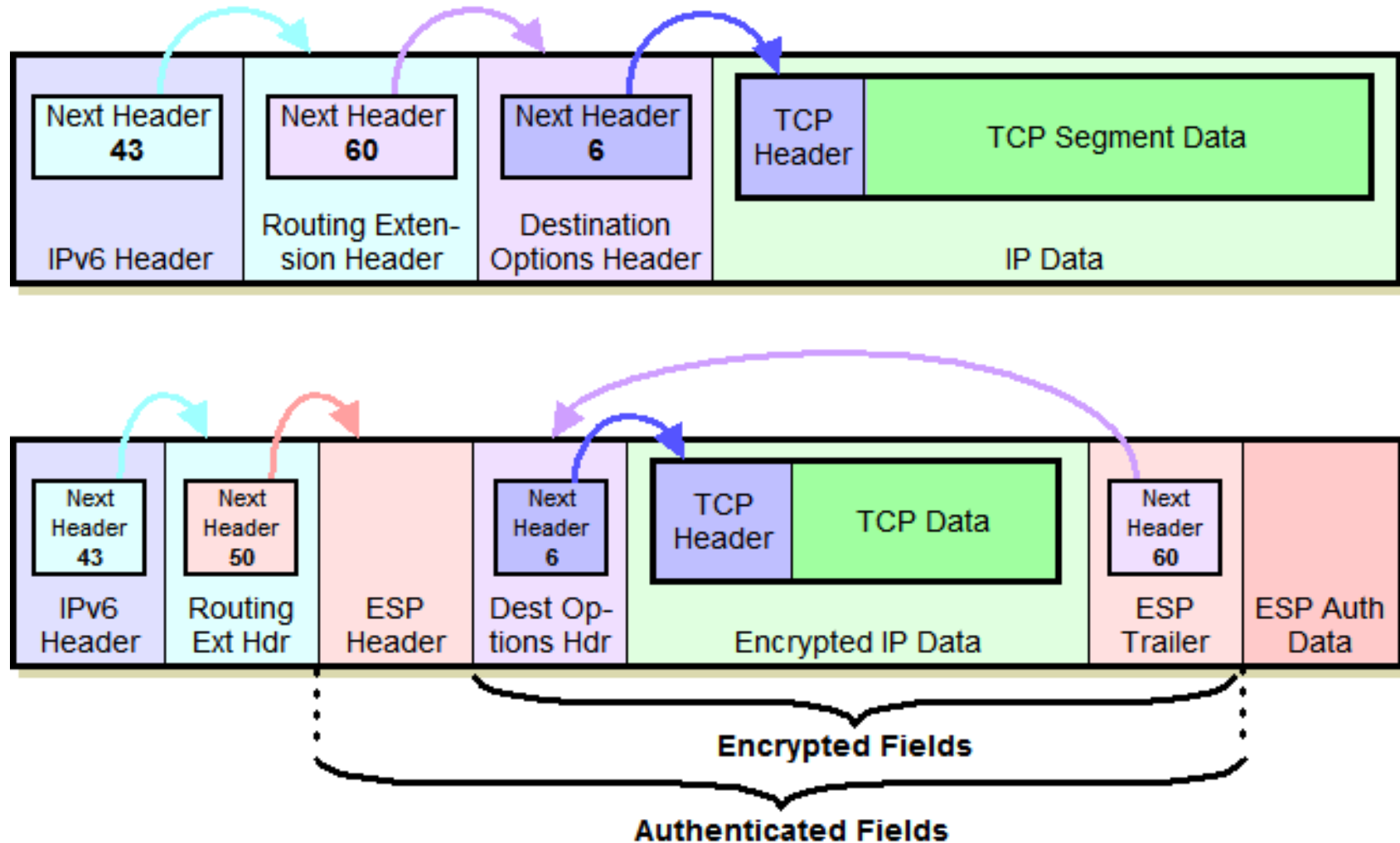
- Entre routers, apenas em modo túnel:





# Opções do IPv6 – Confidencialidade

## ● Modo transporte:

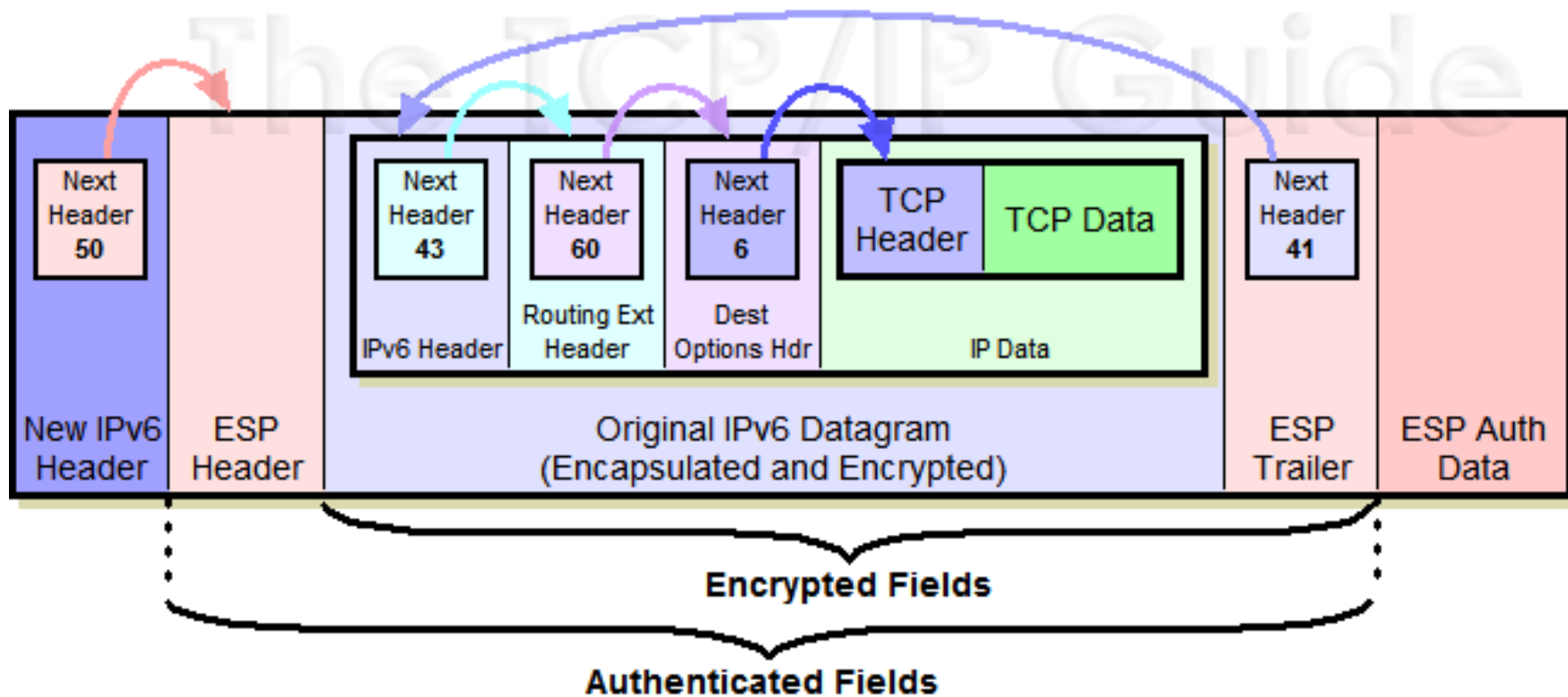


Fonte: The TCP/IP Guide <http://www.tcpipguide.com/>, (Adaptado)



# Opções do IPv6 – Confidencialidade

- **Modo túnel:**



Fonte: The TCP/IP Guide <http://www.tcpipguide.com/>, (Adaptado)



- **Garantias de segurança do IPsec**

	AH	ESP (encryption only)	ESP (encryption plus authentication)
Connectionless integrity	✓		✓
Data origin authentication	✓		✓
Rejection of replayed packets	✓	✓	✓
Confidentiality		✓	✓
Limited traffic flow confidentiality		✓	✓

- **Duas formas de integridade: Connectionless e anti-replay (pacote a pacote ou com noção de sequência de pacotes)**



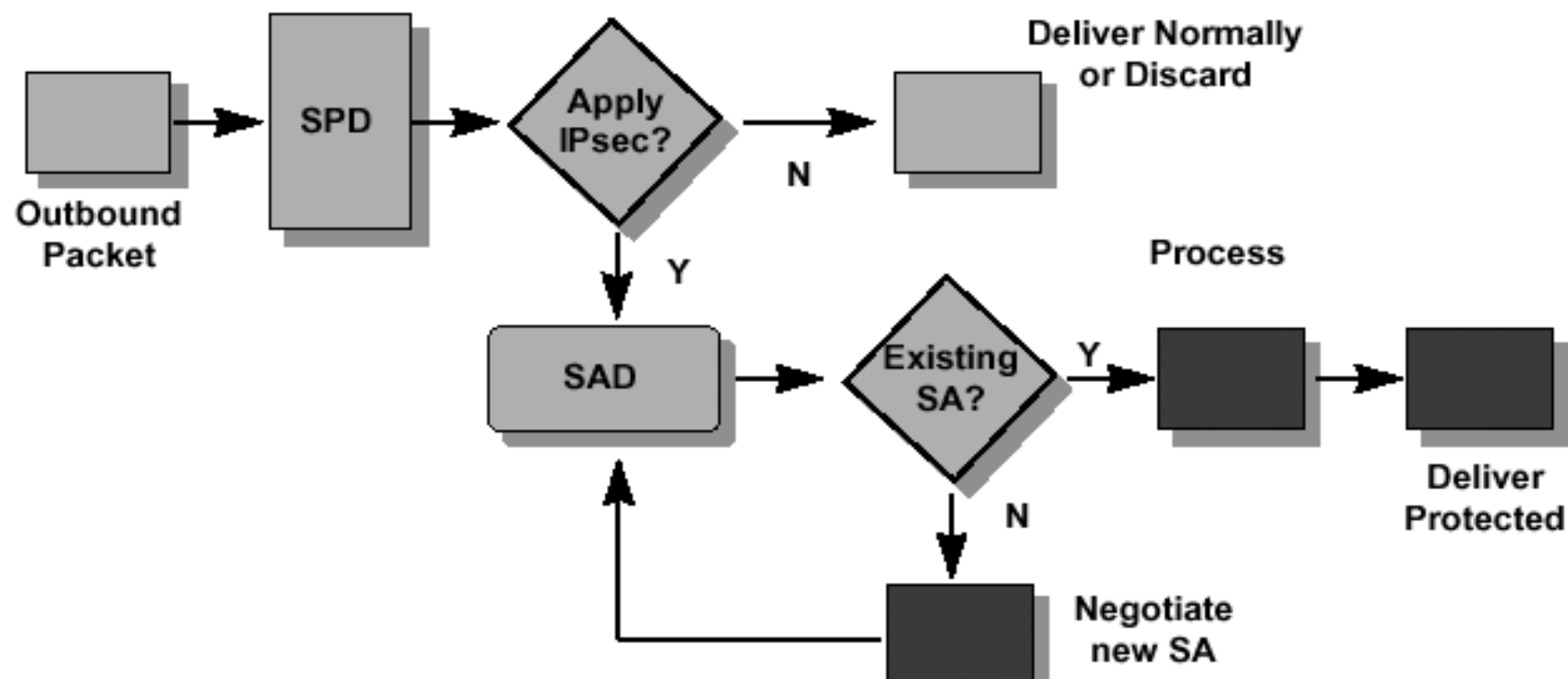
- **IPsec, em termos muito simples**

- Dois sistemas A e B pretendem comunicar de forma segura entre si
- Primeiro estabelecem uma associação de segurança (SA - Security Association) entre si, que pode ser visto com um canal lógico:
  - SA: define conjunto de parâmetros, algoritmos e conjuntos de chaves, negociados entre A e B e armazenados por ambas as partes numa base de dados de associações (SAD)
  - Cada SA é unidireccional e identificada por
    - SPI – Security Parameter Index (índice na BDA)
    - Endereço IP de origem (só unicast)
    - Protocolo (AH ou ESP)
  - Uma ligação com AH e ESP, bidireccional, implica 4 SAs!!
  - Funcionamento (exclusivo) em modo túnel ou modo transporte
- Depois de estabelecida a associação todo o tráfego trocado nessa direcção é sujeito à protecção definida (AH e/ou ESP)





- **Processamento do tráfego de saída:**





- **Processamento do tráfego de entrada:**

