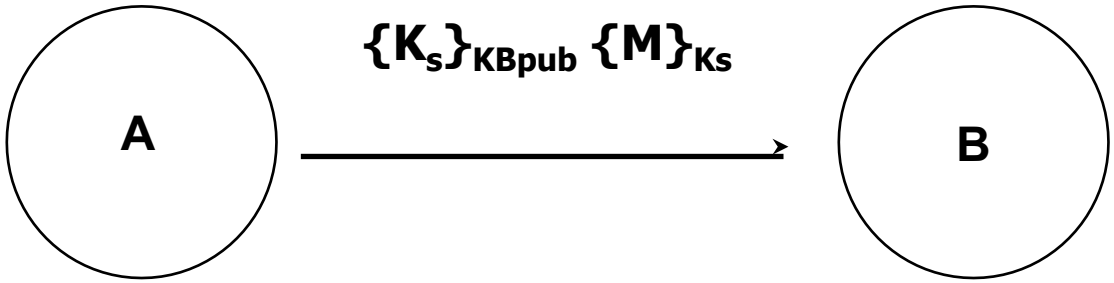
Segurança em Sistemas Distribuídos

Ricardo Costa – rcosta@estg.ipp.pt

Utilização conjunta dos métodos

- A criptografia assimétrica é 100 a 1000 vezes mais pesada que a criptografia simétrica.
- Por esta razão, os dois métodos são utilizados em conjunto.
- Como?
  - As chaves públicas são utilizadas para **autenticação** e para **acordar** uma **chave de sessão**
  - Criptografia simétrica entre os dois principais.
- Exemplo:



Notações mais frequentes

K_A	Chave secreta da Alice
K_B	Chave secreta do Bob
K_{AB}	Chave secreta partilhada entre Alice e Bob
K_{privA}	Chave privada da Alice (só a Alice conhece)
K_{pubA}	Chave pública da Alice (acessível a todos)
$\{M\}_K$	Mensagem M cifrada com K
$[M]_K$	Message M assinada com K

Controlo de integridade com funções de síntese seguras

- É possível usar as funções de síntese para controlar a integridade num canal de dados não seguro.
- O método consiste em usar MACs (”Message Authentication Codes”) que são assinaturas, computacionalmente fáceis de calcular, baseadas em chaves secretas.
- O método funciona assim:
  1. A e B estabelecem uma chave secreta K só conhecida por ambos. K pode ser trocada aquando da autenticação, por exemplo.
  2. Para A enviar a mensagem M a B:

     Calcula h =

     H(M+K) Envia M,h
  3. Ao receber “M,h”, B calcula h’=H(M+K) e verifica integridade da mensagem se h=h’

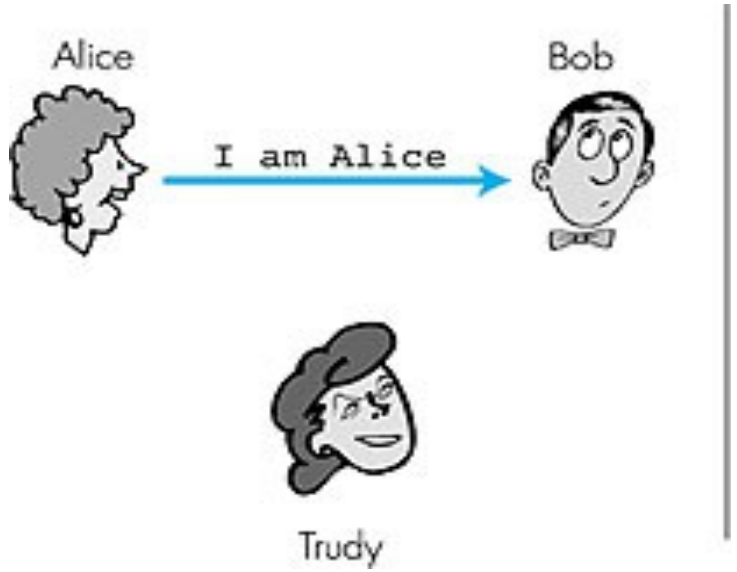
     Porque é que garante autenticação e integridade?

     Só A conhece K, logo só A consegue gerar H(M+K) – autentica emissor de M e garante que a mensagem não foi modificada

NOTA: Não se pretende garantir confidencialidade

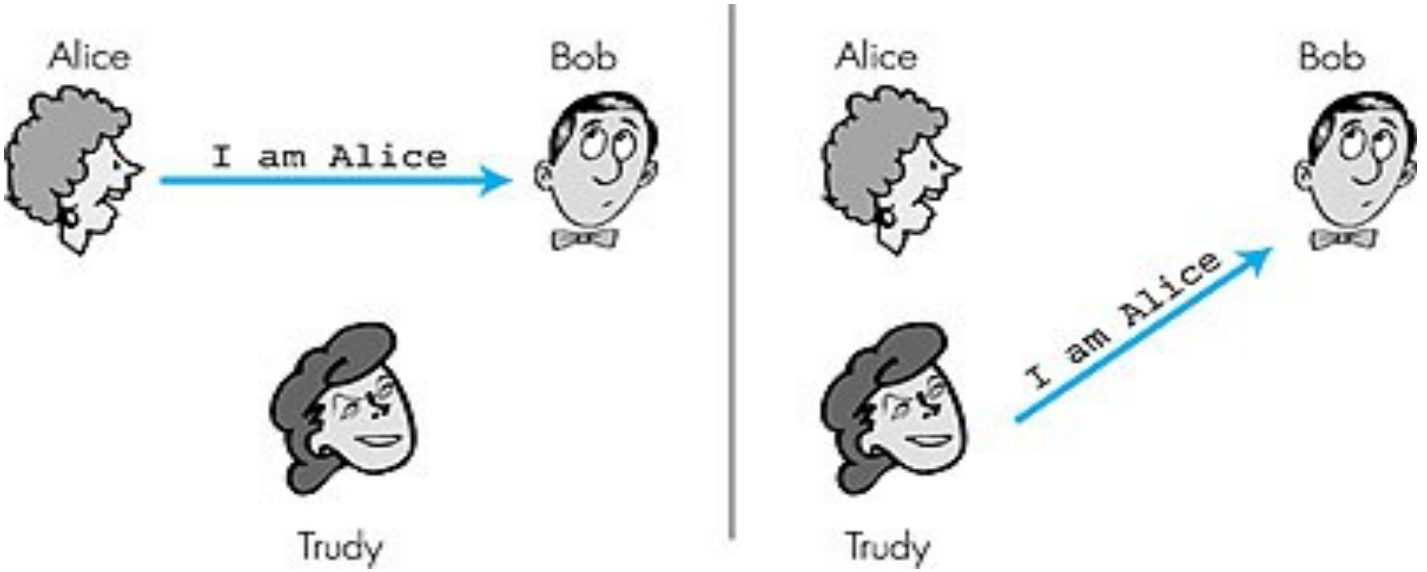
Autenticação

- **Objetivo**: Bob quer que a Alice lhe "demonstre" a sua identidade
- **Protocolo 1**: Alice diz simplesmente "I am Alice"



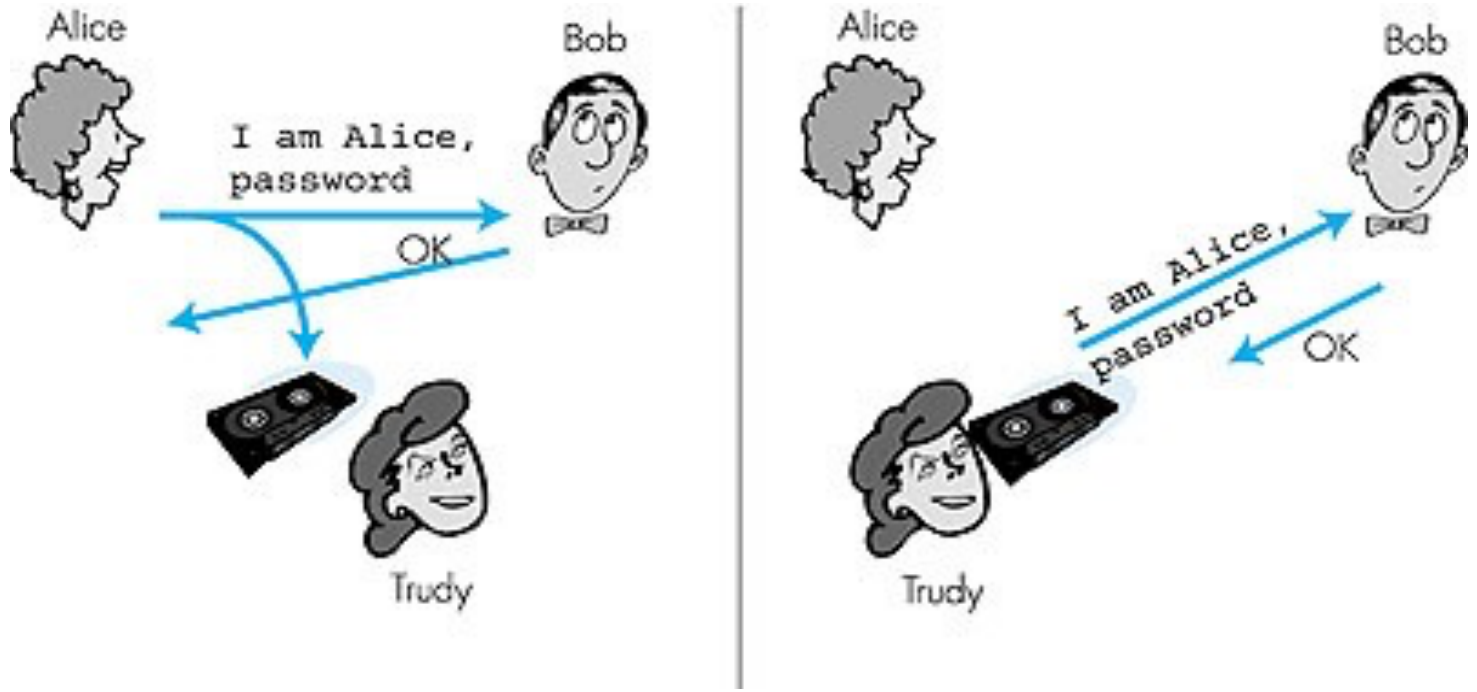
Autenticação

- **Objetivo**: Bob quer que a Alice lhe "demonstre" a sua identidade
- **Protocolo 1**: Alice diz simplesmente "I am Alice"



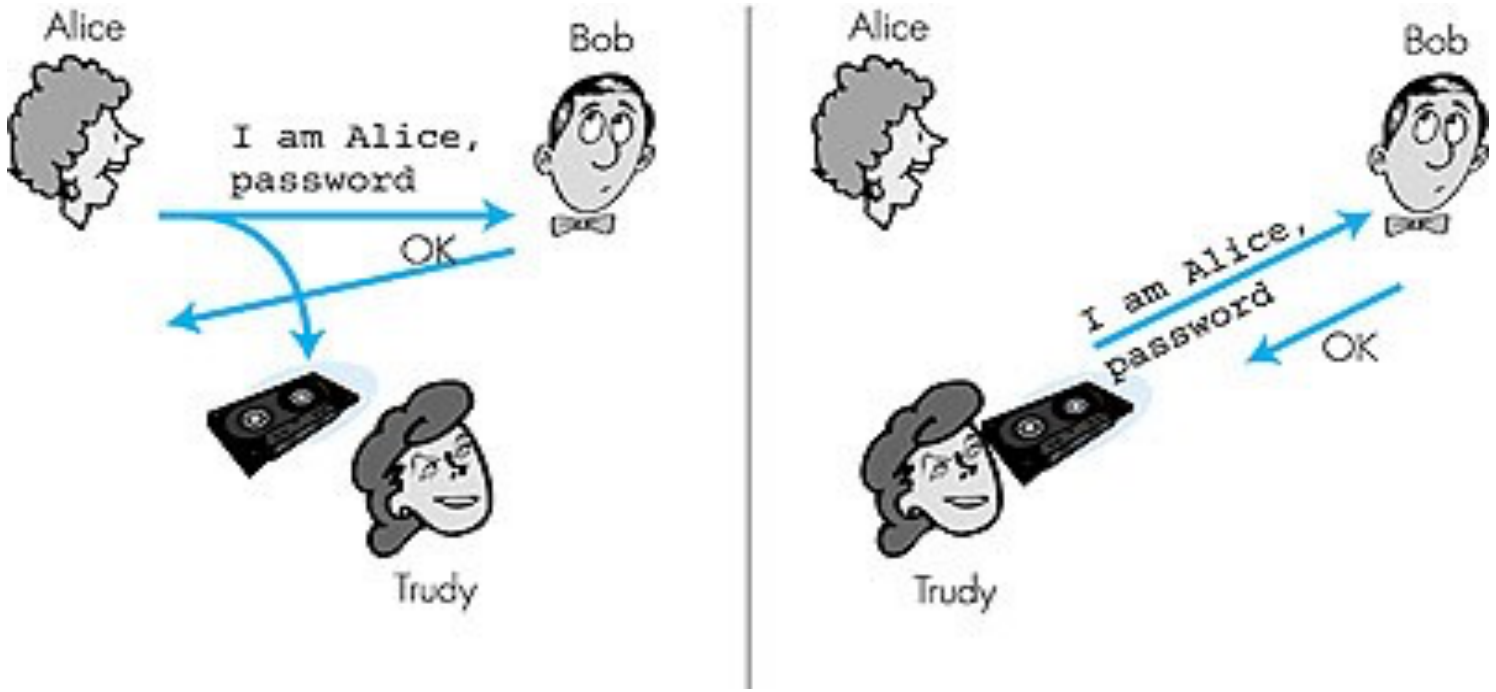
Segunda tentativa

- **Protocolo 2:** Alice diz "I am Alice" e envia também a sua "password"



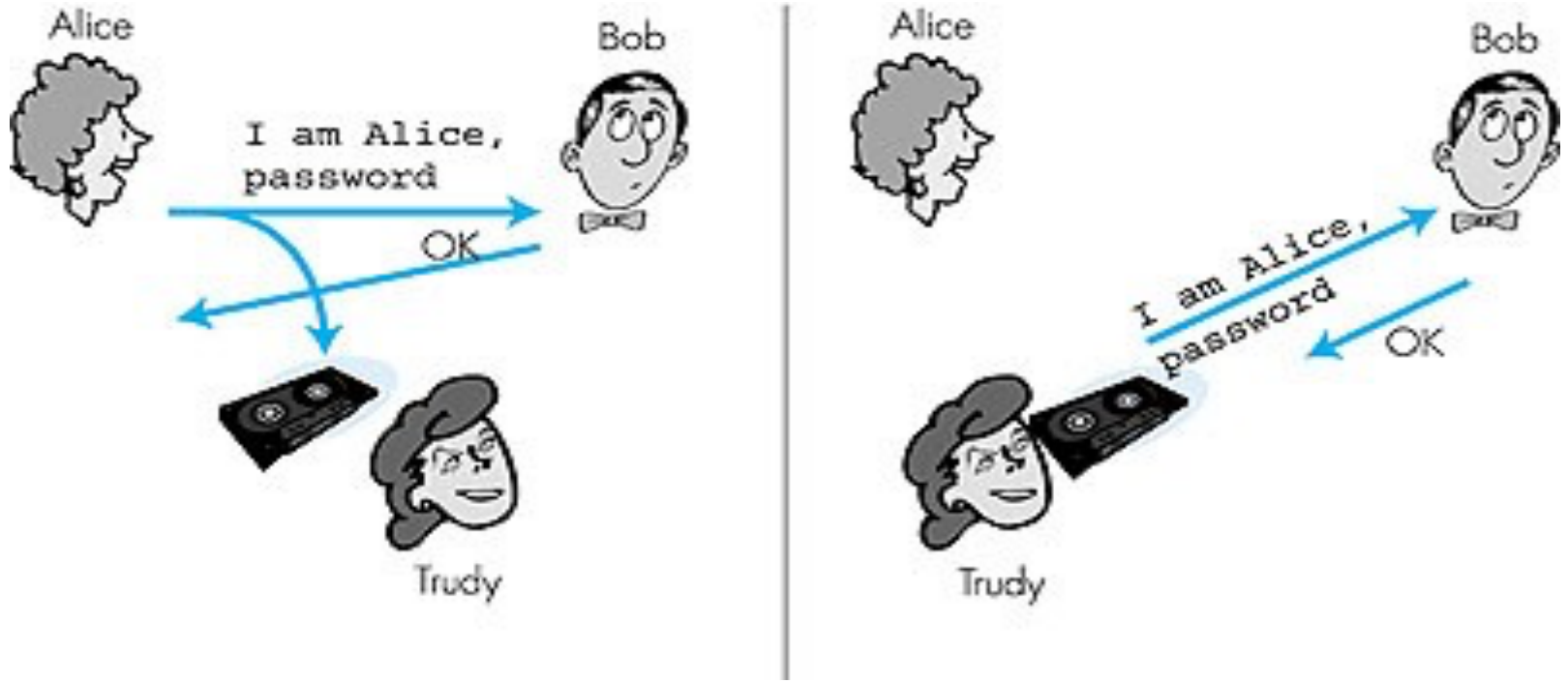
Segunda tentativa

- **Protocolo 2:** Alice diz "I am Alice" e envia também a sua "password"



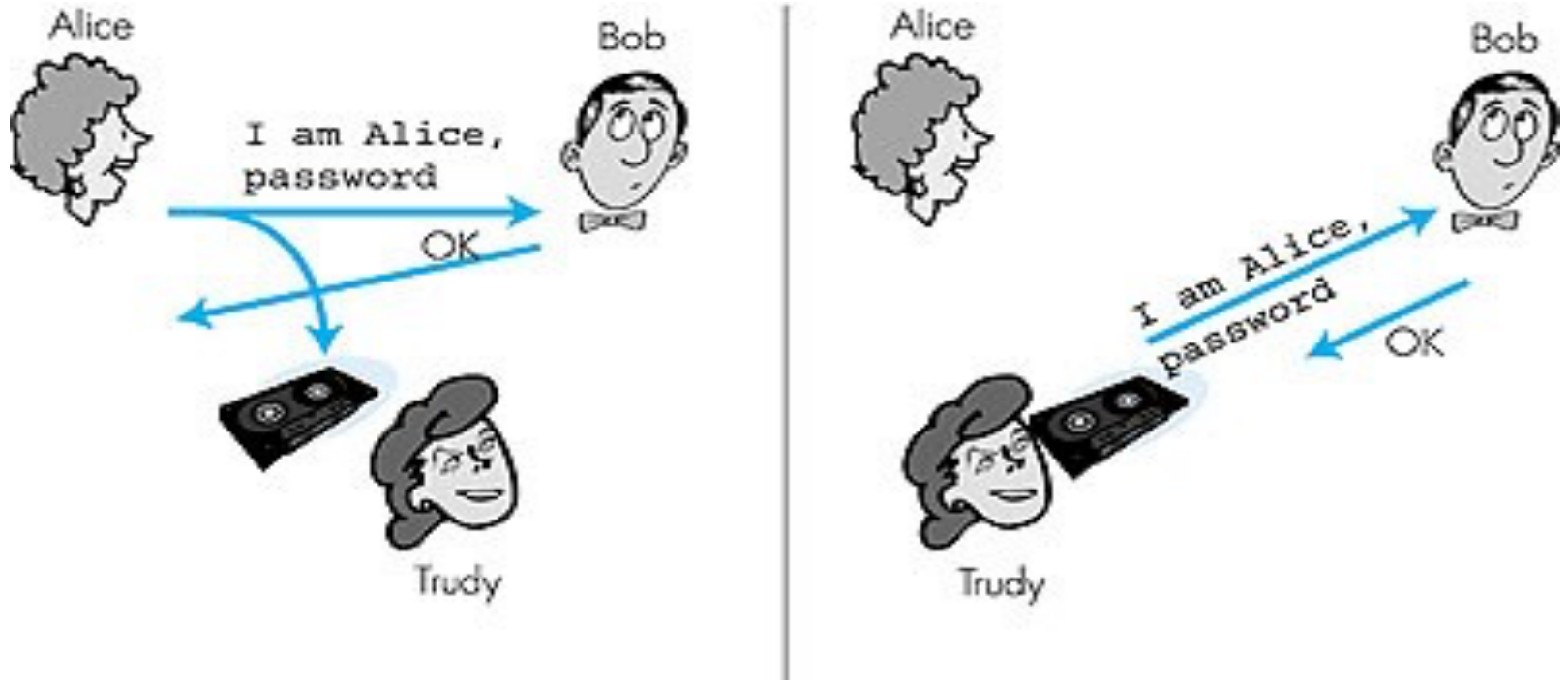
Variante da segunda tentativa

- **Protocolo 3':** Alice diz "I am Alice" e envia também a sua "password" cifrada com uma chave comum a ambos



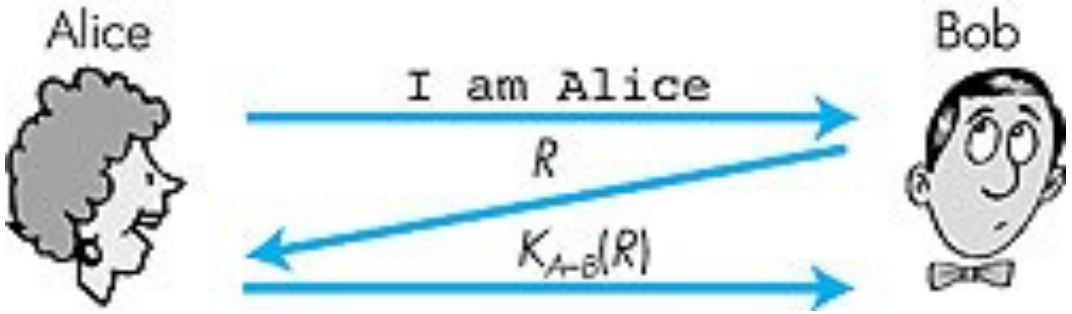
Variante da segunda tentativa

- **Protocolo 3':** Alice diz "I am Alice" e envia também a sua "password" cifrada com uma chave comum a ambos



Método desafio / resposta

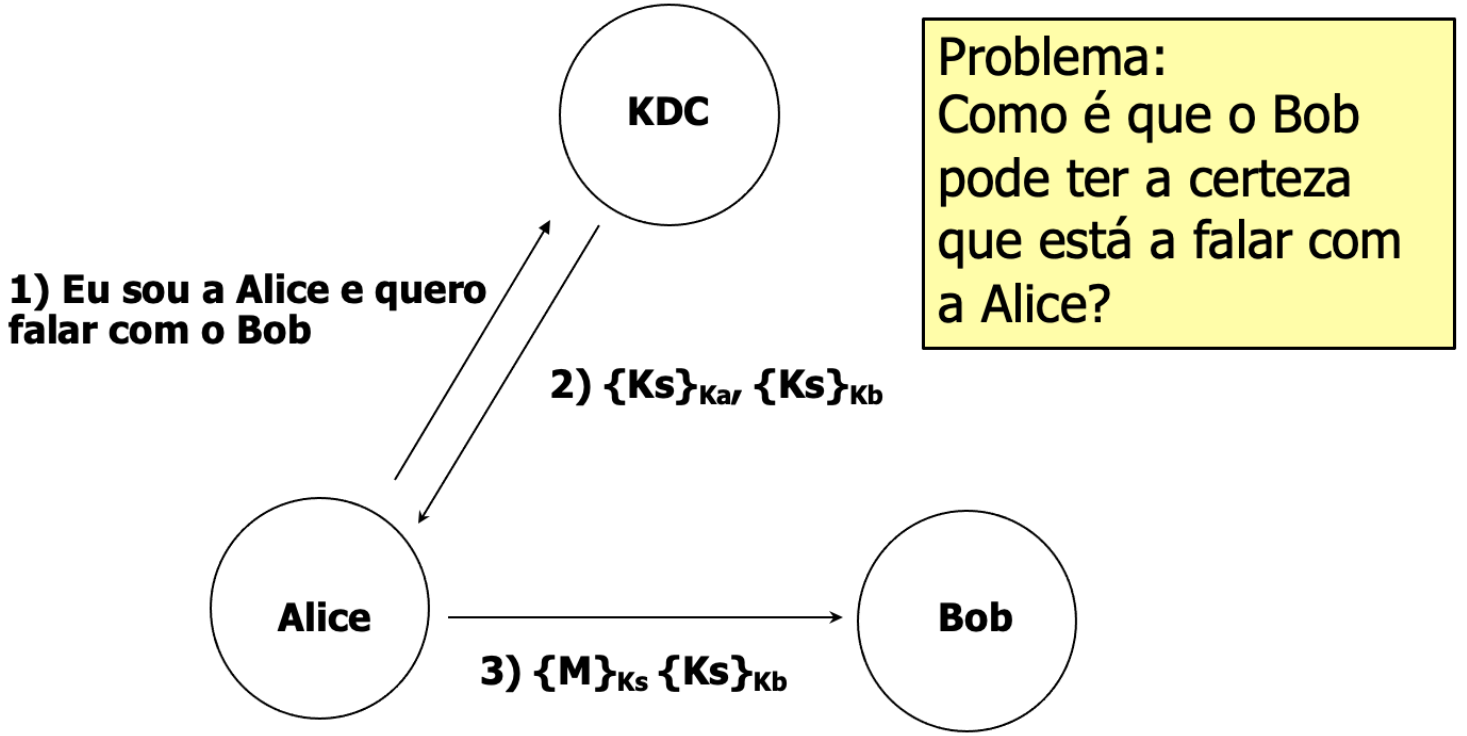
- **Objetivo**: evitar o ataque por repetição (replaying)
- **Nonce**: número usado uma única vez
- **Protocolo 3**: para garantir a "frescura" da transacção, Bob envia à Alice o nonce, R. Alice deve devolver R cifrado com a chave secreta que ambos partilham.



Problemas e possíveis soluções

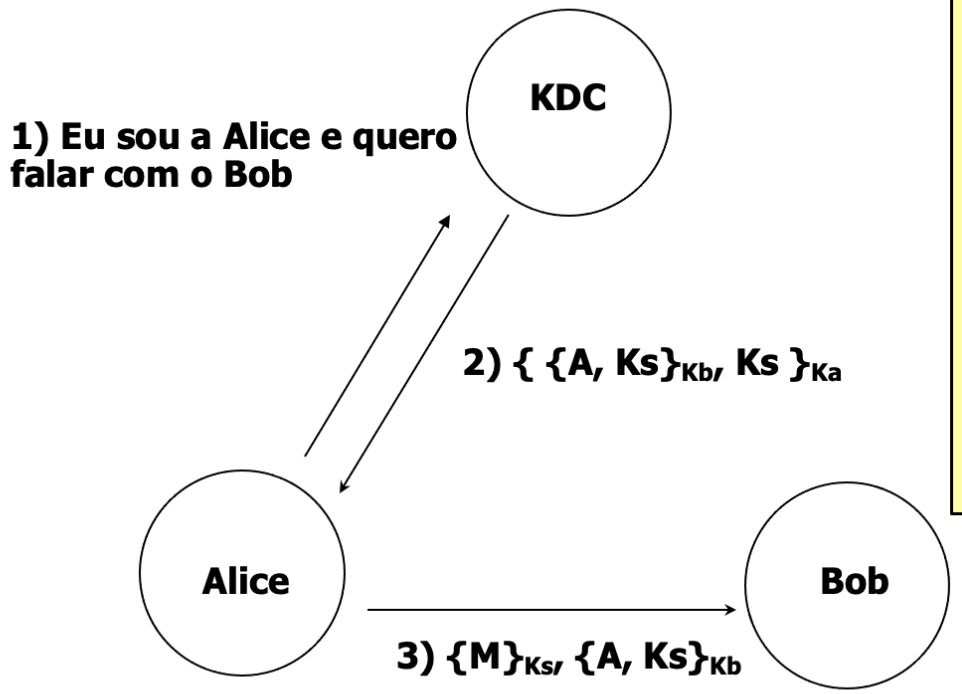
- Apesar de a chave secreta não necessitar de passar na rede durante a autenticação, Alice e Bob têm de arranjar alguma forma de se porem previamente de acordo sobre a chave secreta que vão usar.
- Dado que tal chave não pode ser passada na rede, terão de se encontrar ou usar uma terceira pessoa para trocarem a chave.
- Se isso se revelar um método complexo, vão ter tendência a manter a mesma chave durante muito tempo, o que é perigoso.
- **Solução**: usar uma chave diferente em cada sessão e usar um **centro de distribuição de chaves** (KDC) no qual Alice e Bob **confiam**

Distribuição de chaves de sessão através de um kdc – primeira tentativa



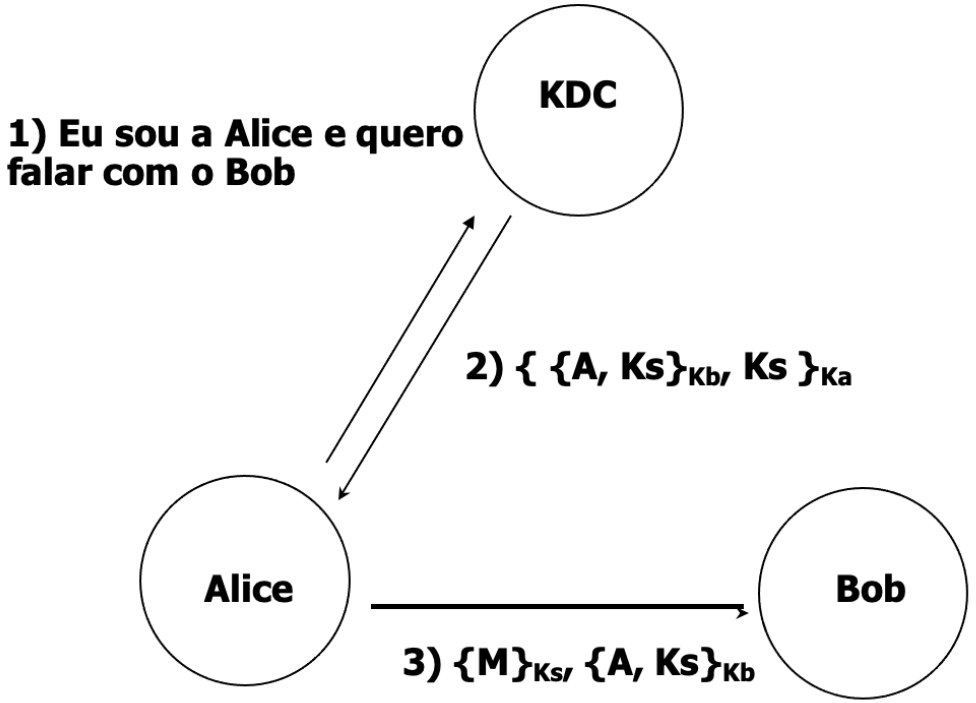
Problema:
Como é que o Bob pode ter a certeza que está a falar com a Alice?

Distribuição de chaves de sessão através de um kdc – segunda tentativa



Problema:
Como é que Bob pode ter a certeza que está a falar com Alice?
Por Alice ter sido capaz de obter $\{A,Ks\}K_b$. Só Alice conhece K_a

Distribuição de chaves de sessão através de um kdc – segunda tentativa



Problema:
Pode Trudy obter o conteúdo da mensagem?
Não, porque apenas Bob conhece K_b. Mas Trudy pode gravar a mensagem e mais tarde tentar incomodar Bob.

Protocolo de Needham-Schroeder

- Este protocolo permite a dois principais A e B (Alice e Bob) estabelecerem um canal seguro entre si e autenticarem-se mutuamente.

- Baseia-se na presença de um KDC (Key Distribution Center) que conhece as chaves secretas de A e B (Ka e Kb) e que é capaz de gerar chaves de sessão (Ks) que vão ser usadas por A e B para comunicarem de forma segura.

- O protocolo resiste aos ataques eavesdropping, masquerading, message tampering e replaying.

O protocolo NS com chaves simétricas

1) A -> KDC: A, B, Na

   A solicita uma chave para comunicar com B, sendo Na um número aleatório gerado por A para garantir a unicidade da transacção (Na deve ser único).

2) KDC -> A: { Na, B, Ks, { Ks, A }Kb }Ka

   O KDC devolve Na, a chave e um ticket ({ Ks, A }Kb), tudo cifrado com a chave de A.

3) A -> B: {Ks, A}Kb, {Na'}Ks

   A solicita comunicar com B, enviando-lhe o ticket

4) B -> A: {Na'-1}Ks

O protocolo NS com chaves simétricas

1) A -> KDC: A, B, Na

   A solicita uma chave para comunicar com B, sendo Na um número aleatório gerado por A para garantir a unicidade da transacção (Na deve ser único).

2) KDC -> A: { Na, B, Ks, { Ks, A }Kb }Ka

   O KDC devolve Na, a chave e um ticket ({ Ks, A }Kb), tudo cifrado com a chave de A.

   O que garante a A ter falado com o KDC ?

3) A -> B: {Ks, A}Kb, {Na'}Ks

   A solicita comunicar com B, enviando-lhe o ticket

4) B -> A: {Na'-1}Ks

O protocolo NS com chaves simétricas

1) A -> KDC: A, B, Na

   A solicita uma chave para comunicar com B, sendo Na um número aleatório gerado por A para garantir a unicidade da transacção (Na deve ser único).

2) KDC -> A: { Na, B, Ks, { Ks, A }Kb }Ka

   O KDC devolve Na, a chave e um ticket ({ Ks, A }Kb), tudo cifrado com a chave de A.

   A receção do Na único garante que A comunicou com o KDC – só o KDC conhece Ka, logo só KDC pode gerar a mensagem indicada.

3) A -> B: {Ks, A}Kb, {Na'}Ks

   A solicita comunicar com B, enviando-lhe o ticket

   O que garante a B estar a falar com A ?

4) B -> A: {Na'-1}Ks

O protocolo NS com chaves simétricas

1) A -> KDC: A, B, Na

   A solicita uma chave para comunicar com B, sendo Na um número aleatório gerado por A para garantir a unicidade da transacção (Na deve ser único).

2) KDC -> A: { Na, B, Ks, { Ks, A }Kb }Ka

   O KDC devolve Na, a chave e um ticket ({ Ks, A }Kb), tudo cifrado com a chave de A.

   A receção do Na único garante que A comunicou com o KDC – só o KDC conhece Ka, logo só KDC pode gerar a mensagem indicada.

3) A -> B: {Ks, A}Kb, {Na'}Ks

   A solicita comunicar com B, enviando-lhe o ticket

   B sabe que está a falar com A, porque apenas A pode obter Ks associada a um ticket indicando A (porque Ks passa cifrado com Ka em 2)). Além disso, é impossível forjar um ticket, porque só B e KDC conhecem Kb B -> A: {Na'-1}Ks

3) B -> A: {Na'-1}Ks

   O que garante a A estar a falar com B ?

O protocolo NS com chaves simétricas

1) A -> KDC: A, B, Na

   A solicita uma chave para comunicar com B, sendo Na um número aleatório gerado por A para garantir a unicidade da transacção (Na deve ser único).

2) KDC -> A: { Na, B, Ks, { Ks, A }Kb }Ka

   O KDC devolve Na, a chave e um ticket ({ Ks, A }Kb), tudo cifrado com a chave de A.

   A receção do Na único garante que A comunicou com o KDC – só o KDC conhece Ka, logo só KDC pode gerar a mensagem indicada.

3) A -> B: {Ks, A}Kb, {Na'}Ks

   A solicita comunicar com B, enviando-lhe o ticket

   B sabe que está a falar com A, porque apenas A pode obter Ks associada a um ticket indicando A (porque Ks passa cifrado com Ka em 2)). Além disso, é impossível forjar um ticket, porque só B e KDC conhecem Kb B -> A: {Na'-1}Ks

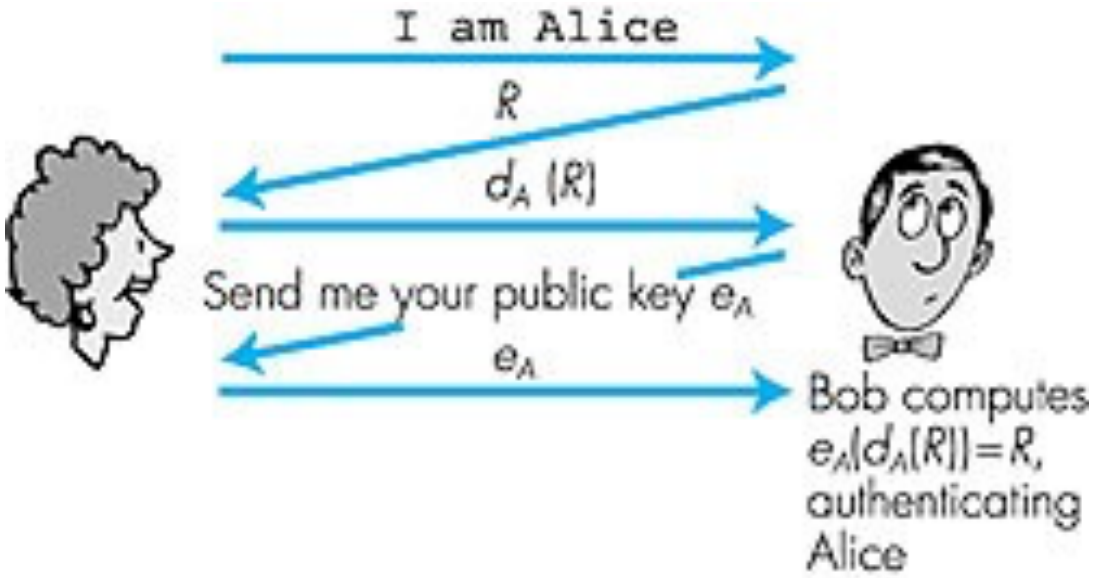
3) B -> A: {Na'-1}Ks

   B prova ser B por ser capaz de decifrar {Na'}Ks, o que pressupõe ter sido capaz de decifrar {Ks, A}Kb

   Porque não pode B enviar {Na'}Ks ?

Autenticação via chave pública

- **Problema**: como é que Bob e Alice se autenticam utilizando chaves assimétricas?
- **Resposta**: usa um nonce, e criptografia assimétrica



Protocolo de NS com chaves públicas

- Pressupondo que A e B **conhecem** as **chaves públicas** um do outro, de forma **certificada**, podem estabelecer um canal seguro e autenticarem-se mutuamente através de:

1. **A -> B:** **{ A, Na }K_{Bpub}**
2. **B -> A:** **{ Na, Nb, Ks } K_{Apub}**
3. **A -> B:** **{ Nb } K_s**

O que garante a A estar a falar com B?

Receber Na em 2 – apenas B consegue obter Na, porque apenas B tem K_{bpriv}

O que garante a B estar a falar com A?

Receber Nb em 3 – apenas A consegue obter Nb, porque apenas A tem $_{KApriv}$

O que garante que Ks é seguro?

Ks apenas passa na rede em 2 – apenas A consegue obter Ks, porque apenas A tem K_{apriv} (B conhece K_s porque gerou K_s)

Criptografia assimétrica é lenta. Como solucionar o problema?

Negoceia chave simétrica **Ks** para usar durante a comunicação após a autenticação

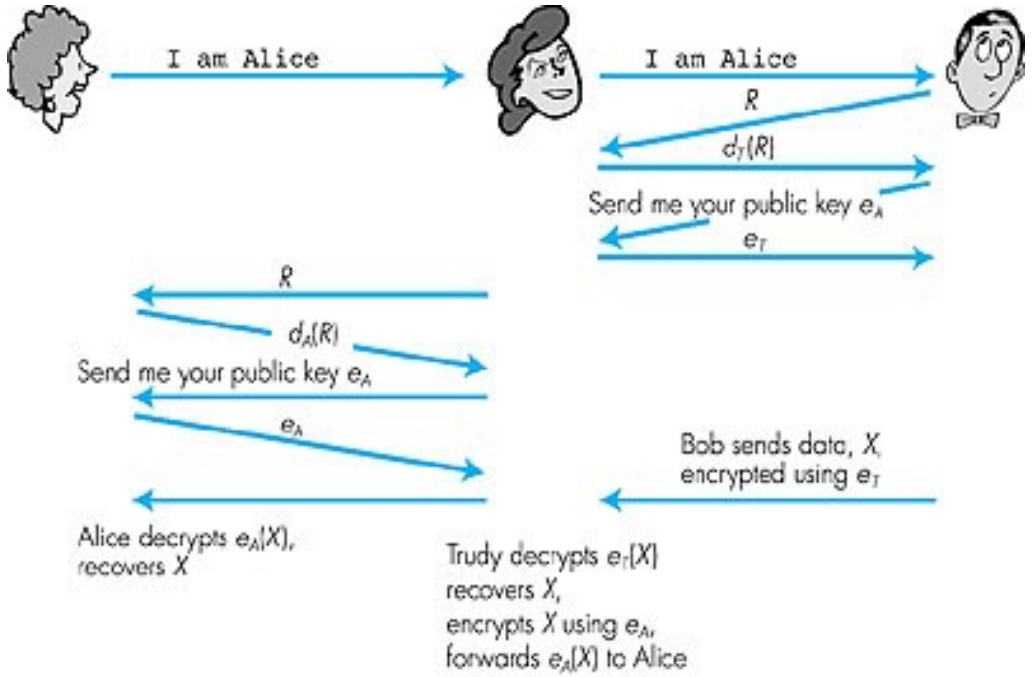
Protocolo de NS com chaves públicas

- Se A apenas quiser criar um canal cifrado e ter a certeza de que está a falar com B basta:

A -> B:	**{ A, Na, Ks }KBpub**
B -> A:	**{ Na } Ks**

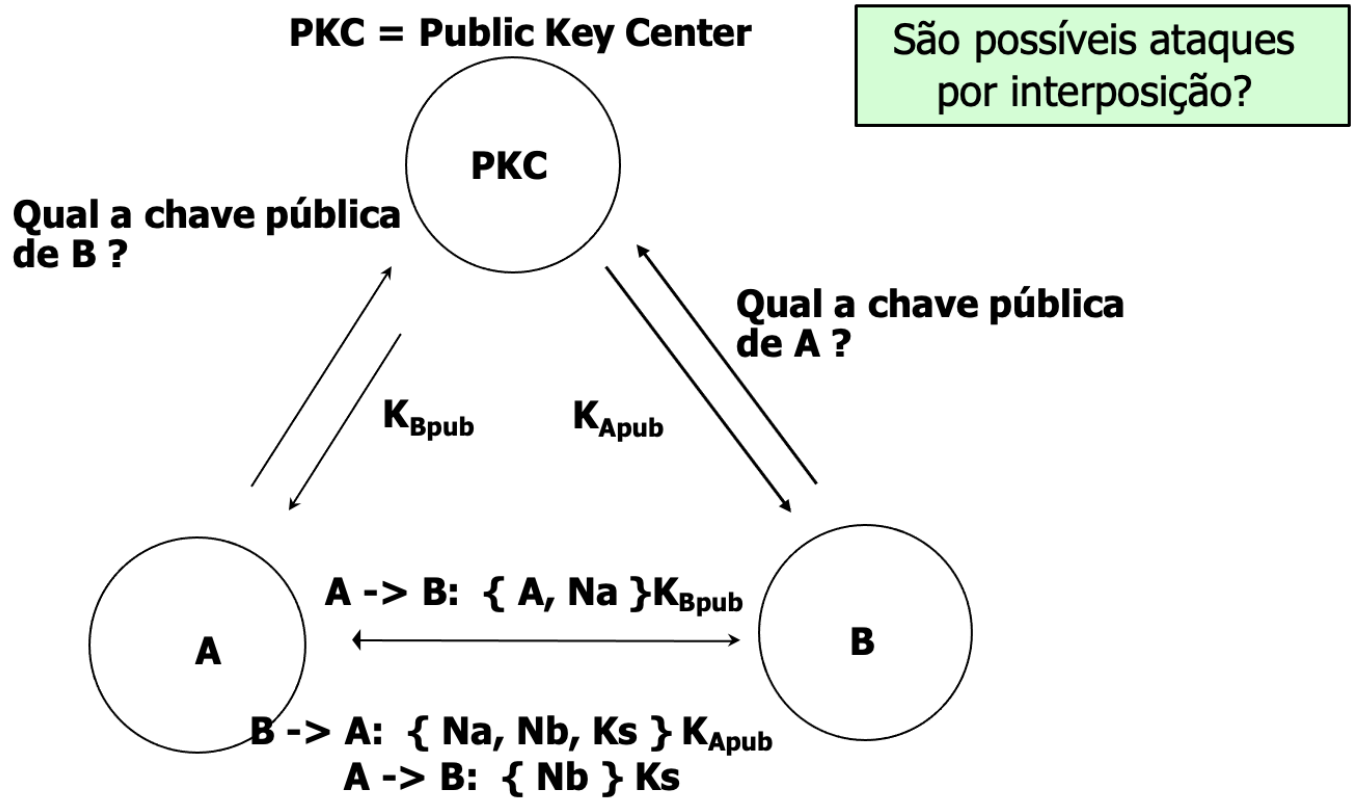
O que garante a A estar a falar com B?

Ataques por interposição



- São necessárias chaves públicas "certificadas"

Centro de distribuição de chaves públicas (1)



Centro de distribuição de chaves públicas (2)

- O Public Key Center (PKC) apenas conhece aquilo que é público, isto é, as chaves públicas dos principais. Tal é um progresso notável pois o papel da trusted computing base foi drasticamente reduzido.
- Para que tudo funcione bem é apenas necessário assegurar que o PKC tem as chaves públicas verdadeiras e que os principais têm a certeza que estão a falar com o verdadeiro PKC e não com um impostor.
- Nos protocolos anteriores, um intruso que se consiga fazer passar pelo PKC, consegue levar A e B a usarem chaves conhecidas
  - São necessárias chaves públicas "certificadas"

Distribuição das chaves públicas

- É necessário ter absoluta segurança de que se está a dialogar com uma fonte fidedigna que nos está a entregar a verdadeira chave pública que pretendemos
  - Se se conhece a chave pública dessa fonte fidedigna, uma forma de obter esta confiança é essa fonte cifrar a sua resposta com a sua chave secreta, assinando-a desse modo
- Métodos de distribuição de chaves:
  - **Certificate Granting Authority** – entidades cujas chaves públicas são bem conhecidas e que "assinam" as chaves / certificados que entregam. Este método está hoje em dia normalizado de forma oficial.
  - **Web of trust** - método informal por transitividade da relação de confiança, também suportado na "assinatura" das informações trocadas entre principais. Este método foi vulgarizado pelo programa PGP para troca de correio electrónico.

Assinaturas digitais

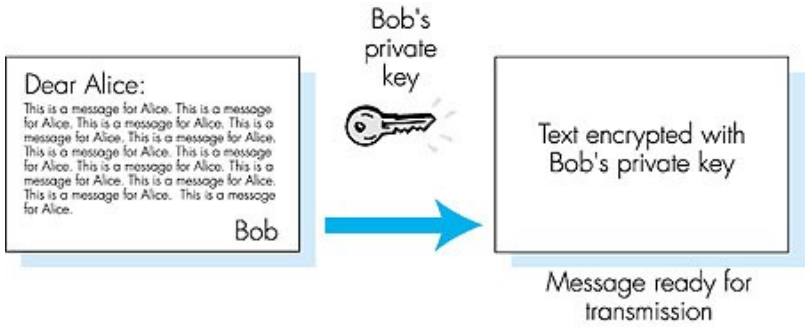
- **Objetivo**: desenvolver um mecanismo digital que substitua as assinaturas efetuadas num documento em papel
  - Quais as propriedades?

- Um sistema de assinaturas digitais deve ter as seguintes propriedades:
  - **Autenticação**: o recetor deve poder verificar que a assinatura é autentica
  - **Integridade**: a assinatura deve garantir que a mensagem assinada não foi alterada, nem durante o trajeto, nem pelo recetor, mesmo que tenha passado em claro
  - **Não repudio**: o emissor não poderá negar que de facto enviou a mensagem assinada

  NOTA: O objetivo das assinaturas não é esconder o conteúdo

Assinaturas digitais : primeira tentativa

- Assinatura digital da mensagem M efetuada por Bob: {M}Kbpriv



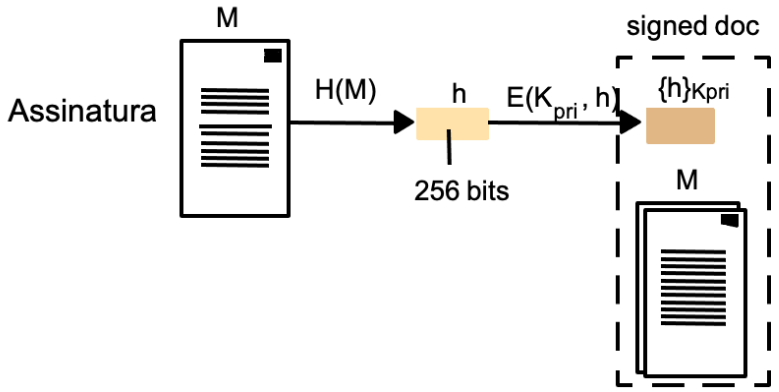
- Propriedades:
  - **Autenticação** verificada decifrando M = $\{\{M\}K_{Bpriv}\}K_{Bpub}$
  - **Integridade** e **não repudio** garantidos porque ninguém consegue alterar/criar $\{M\}K_{Bpriv}$ sem conhecer K_{bpriv}
  - Dimensão da assinatura idêntica à do documento
  - Necessidade de cifrar todo o documento com chaves assimétricas

Solução?

Assinatura digital: solução com hash

- Assinatura digital da mensagem M efectuada por Bob: $\{H(M)\}K_{Bpriv}$
  - Mensagem a enviar: $M,\{H(M)\}K_{Bpriv}$
  - H(M): função de síntese segura

Utilização de assinaturas digitais com chaves públicas

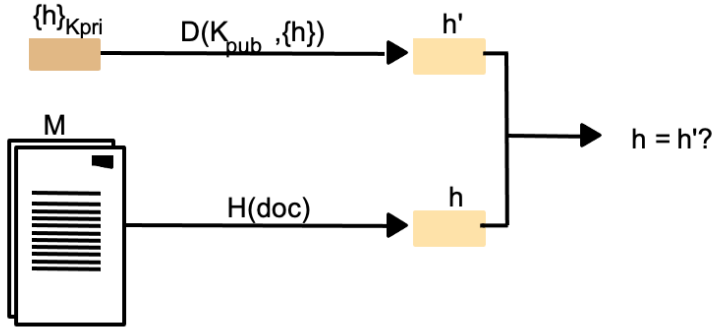


Bob envia uma mensagem com uma assinatura digital:

$M,\{H(M)\}K_{Bpriv}$

Alice verifica a assinatura e a integridade da mensagem Verificação

$H(M) = \{H(M)\}K_{bpriv}\}K_{Bpub}$?



Assinatura digital: solução com chaves públicas

- Assinatura digital da mensagem M efectuada por Bob: {H(M)} KBpriv

  Mensagem a enviar: M,{H(M)}KBpriv H(M):

  função de síntese segura

- Propriedades:

  Verificação ao receber M',{H(M)}KBpriv : H(M') = {{H(M)}KBpriv} KBpub

  Autenticação, integridade e não repudio garantidos porque apenas B consegue criar {H(M)}KBpriv e ninguém consegue alterar M para M' de forma que H(M')=H(M)

  Dimensão da assinatura pequena e constante

  Apenas assinatura é cifrada – mensagem pode passar em claro

Certificados de chaves públicas

- A segurança baseada em chaves públicas depende da validade das chaves públicas.
- **Objetivo**: um certificado deve permitir estabelecer a autenticidade de uma chave pública (por declaração de uma entidade fidedigna)
- Certificado contém assinatura relativa, pelo menos, à informação do nome e chave pública da entidade. Exemplo:

1. Certificate type	**Public key**
2. Name	**Bob's Bank**
3. Public key	**K_{Bpub}**
4. Certifying authority	**Fred – The Bankers Federation**
5. Signature	**{H(field 2 + field 3)} K_{Fpriv}**

Autoridades de certificação

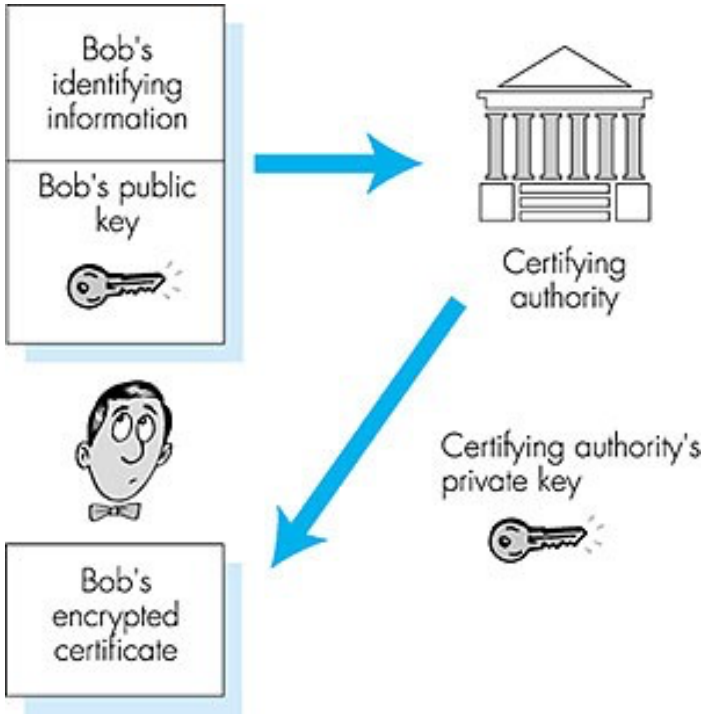
- Uma certification authority (CA) associa chaves públicas a principais

  Em geral, as chaves públicas (certificados) das CAs são bem-conhecidos

- Quando a Alice quer verificar qual a chave do Bob:

1) obtém um certificado dessa chave (Bob pode enviá-lo!).

2)verifica autenticidade do certificado verificando a assinatura efetuada pela CA (usando chave pública no certificado da CA)



Certificados x.509

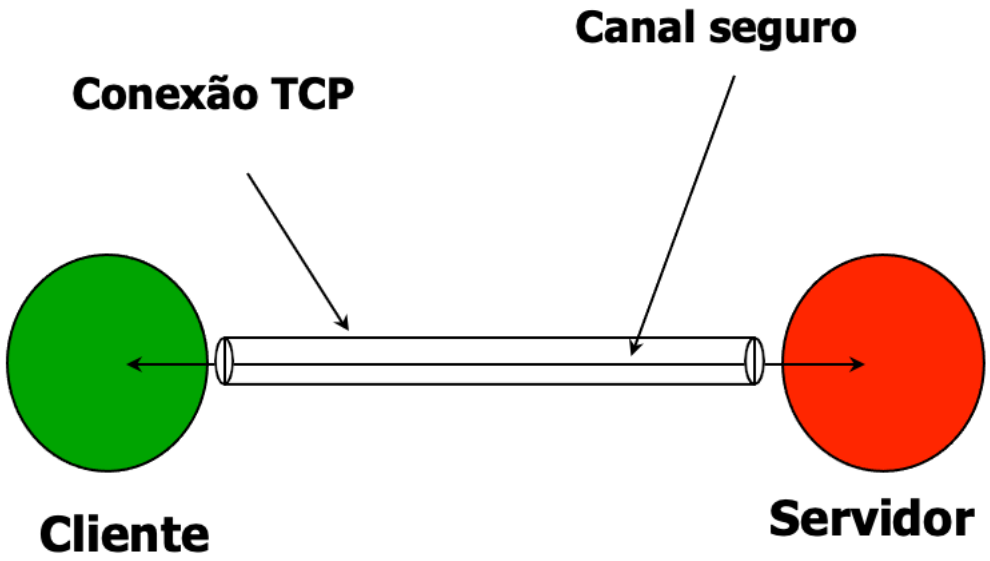
Entidade	Nome identificativo, chave pública
Emissor	Nome identificativo, assinatura
Período de validade	Data de início, data de fim
Informação administrativa	Versão, número de série
Informação adicional	

Repudio e validade de uma chave

- Quando uma chave privada é comprometida é impossível garantir a autenticidade de qualquer mensagem
- Por isso, quanto antes, é necessário revogar a chave pública que lhe estava associada.
- No caso dos certificados de chave pública, é necessário que as entidades de certificação memorizem os certificados válidos e revogados.
- Como os certificados expiram ou podem ser revogados, é fundamental verificar o seu estado antes de confiar na informação.
- A utilização de certificados, por si só, não é uma panaceia universal.

TLS (SSL) – Transport Layer Security

- Trata-se de um protocolo originalmente proposto e desenvolvido pela Netscape (SSL), do tipo sessão, isto é sobre o nível transporte, que permite estabelecer canais seguros e autenticar clientes e servidores. Hoje em dia trata-se de uma norma IETF.



O protocolo SSL

- O protocolo SSL funciona por cima do nível de transporte e permite fornecer segurança a qualquer aplicação baseada em TCP
- É usado entre browsers e servidores WWW (https).
- Funcionalidades de segurança:
  - autenticação do servidor
  - cifra dos dados
  - autenticação do cliente (opcional)

  - Autenticação do servidor:

    O cliente (browser) inclui as chaves públicas de várias CAs

    O cliente solicita ao servidor um certificado do servidor emitido por uma CA em que ele confie.

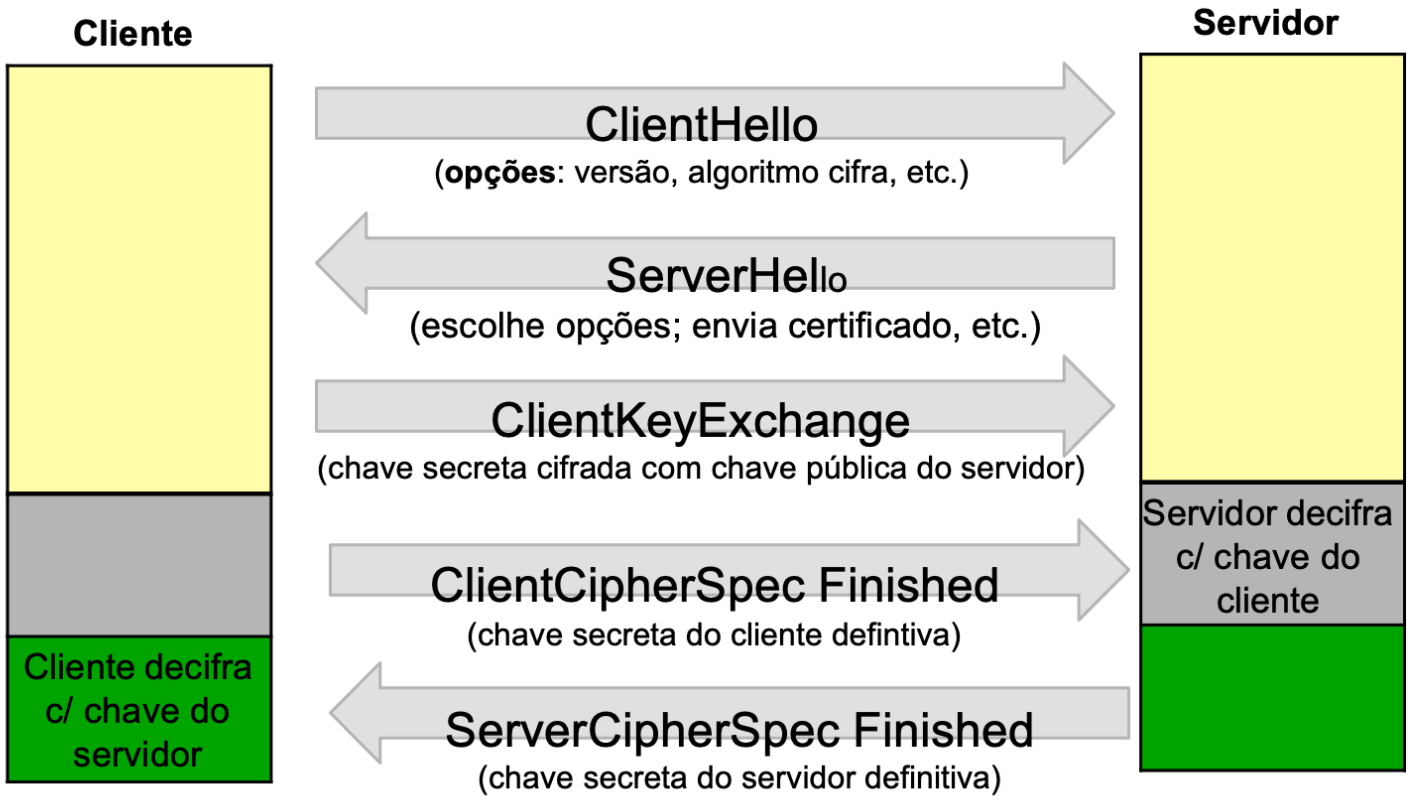
    O cliente verifica o certificado do servidor com a chave pública da CA

  - Autenticação do cliente:

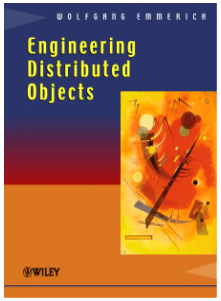
    Processa-se de forma semelhante

Veja no seu browser na secção de segurança.

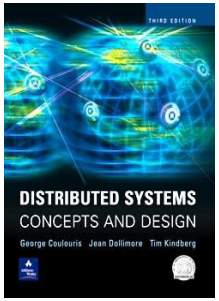
Negociação/Handshake TLS (estabelecimento de sessão)



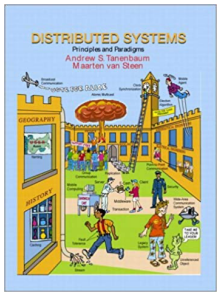
Bibliografia



From: Wolfgang Emmerich
Engineering Distributed Objects
John Wiley & Sons, Ltd 2000



From: Coulouris, Dollimore and Kindberg
Distributed Systems: Concepts and Design
Edition 4 © Addison-Wesley 2005



From: Andrew S., Tanembaum and Van Steen, Maarten
Distributed Systems: Principles and Paradigms
Edition 2 © Pearson 2013

Questões?