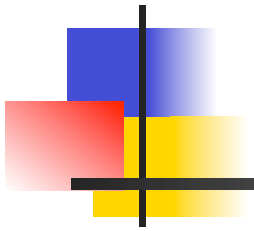


Cap. 6 – Introdução à segurança das comunicações

Este capítulo apresenta uma breve introdução às técnicas de segurança centrada na problemática da comunicação através de canais seguros

Nota prévia

A apresentação é semelhante e utiliza algumas das figuras do livro de base do curso

James F. Kurose and Keith W. Ross, "Computer Networking - A Top-Down Approach Featuring the Internet," Addison Wesley Longman, Inc., 3rd Edition, 2005

Organização do capítulo

- Apresentação do problema e terminologia
- Canais seguros e seus requisitos
- Criptografia simétrica
- Utilização da criptografia simétrica para autenticação e comunicação
- Criptografia assimétrica
- Utilização da criptografia assimétrica para autenticação e comunicação
- Assinaturas digitais
- Autoridades de certificação
- Estudo de caso: o protocolo SSL

Terminologia

- Num sistema existem entidades que do ponto de vista da segurança têm identidade própria, direitos e deveres - essas entidades podem ser utilizadores, máquinas, programas, etc.
- Utiliza-se o termo entidade para designar genericamente cada uma delas
- A segurança do sistema distribuído passa por:
  - Autenticar as entidades — autenticação
  - Verificar os seus direitos de acesso aos objectos — controlo de acessos
  - Utilizar canais seguros para impedir o acesso, alteração ou destruição indevida de informação, ou seja, proteger a privacidade

Definições

- *Entidade* - uma entidade (pessoa, processo, servidor, cliente, …) que é singular do ponto de vista dos direitos no sistema.

- *Controlo de acessos* - dada uma operação Op sobre um objecto O, é necessário decidir se o entidade E pode aplicar Op a O.

- *Autenticação* - uma entidade que pretende ter a identidade "E" tem de ser capaz de provar que é "E".

- Para a autenticação geralmente utiliza-se um método lógico do tipo segredo partilhado entre E e quem o autentica (de que uma palavra chave é o exemplo mais conhecido), mas também se pode basear na verificação de atributos físicos (identificação da voz, impressões digitais ou características da retina por exemplo) ou na posse de algo que só E pode possuir (um cartão magnético por exemplo).

Os canais normais não são seguros

Indiscrição (*Eaves-dropping*)

Mascarar-se ou pretender ser outro (*Masquerading*)

Reemissão de antigas mensagens (*Message replaying*)

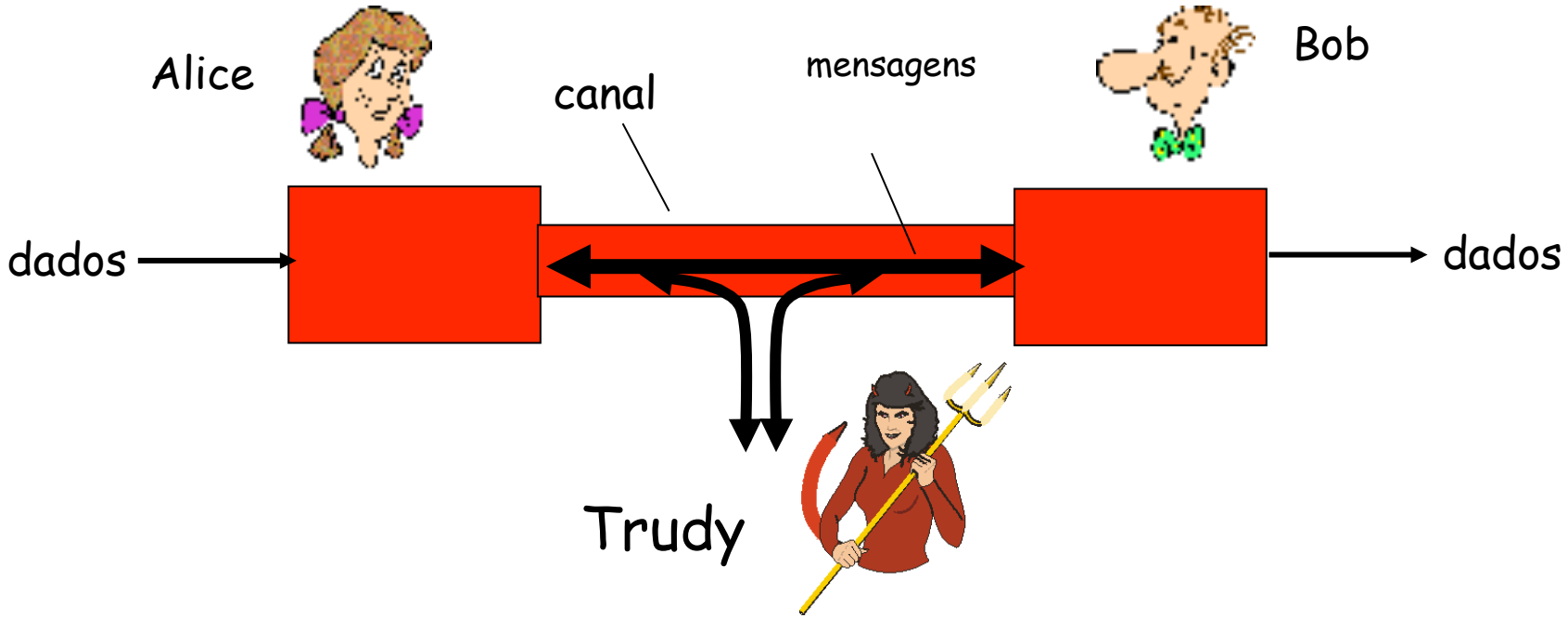
Alteração indevida do conteúdo das mensagens (*Message tampering*)

Supressão de mensagens (*Message supression*)

Repudiar as mensagens enviadas anteriormente pelo próprio

Alice, Bob e Trudy

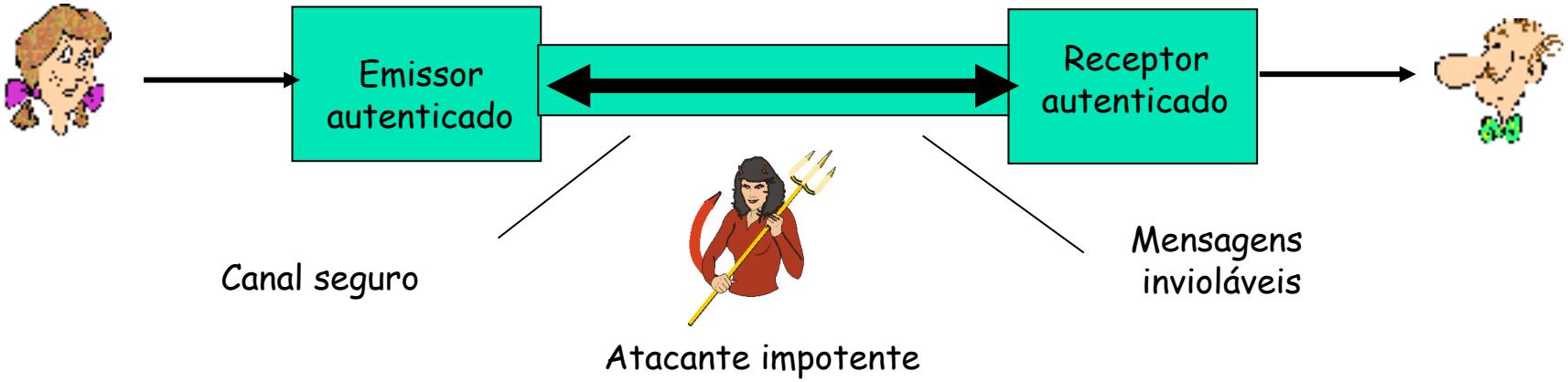
- Personagens (entidades) usadas com frequência em segurança
- Bob e Alice (namorados) pretendem comunicar de forma segura
- Trudy (*intruder* ou atacante) pode interceptar, suprimir, modificar ou re-injectar mensagens



Comunicação segura e canais seguros

- Num canal seguro as entidades estão autenticadas
- O inimigo não pode copiar, alterar ou introduzir mensagens
- O inimigo não pode reenviar mensagens antigas
- O inimigo não pode reordenar as mensagens

Comunicação segura — troca de dados com confidencialidade, integridade e autenticidade e, se necessário, não repúdio da emissão

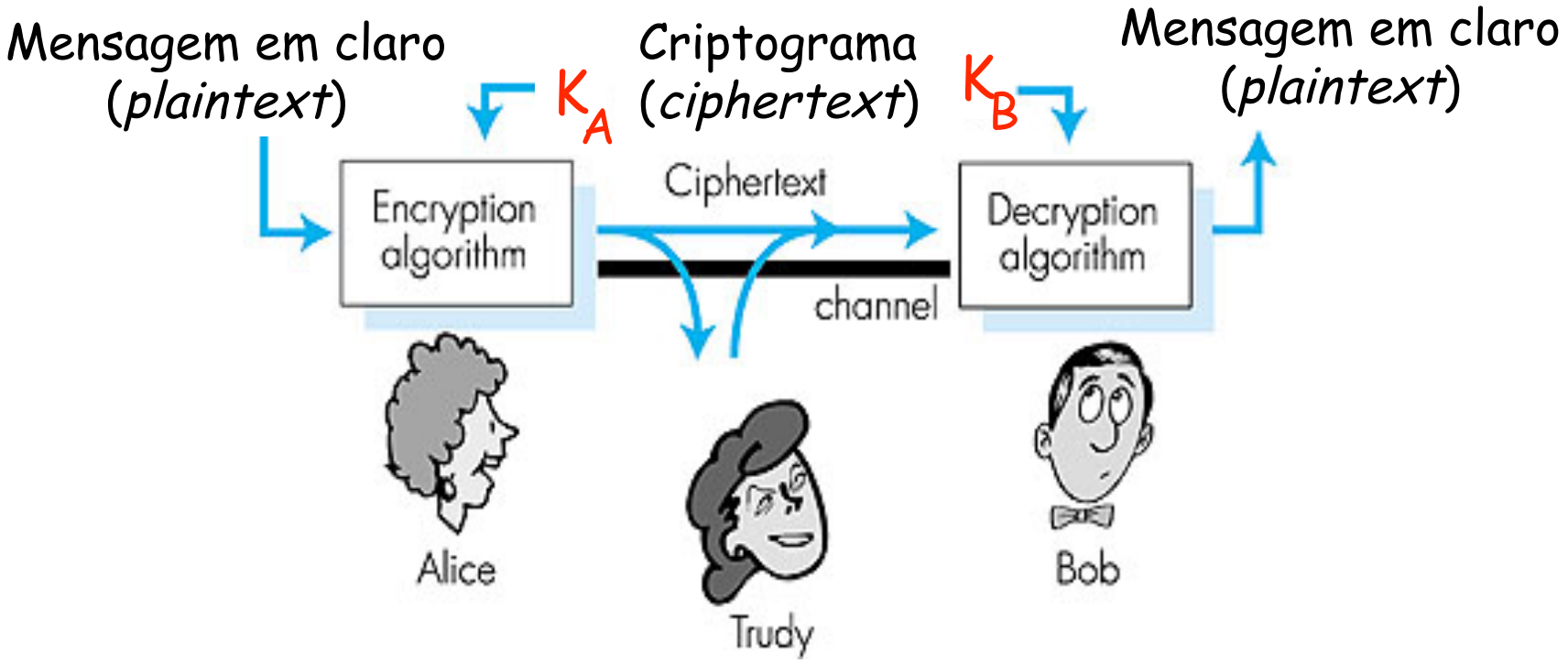


Criptografia

Criptografia é a disciplina que inclui os princípios, meios e métodos de transformação dos dados com a finalidade de esconder o seu conteúdo semântico, de estabelecer a sua autenticidade, de impedir que a sua alteração passe despercebida, de evitar o seu repúdio e/ou impedir a sua utilização não autorizada.

Chave criptográfica é um parâmetro utilizado com um algoritmo criptográfico para transformar, validar, autenticar, cifrar ou decifrar dados.

A linguagem da criptografia



Criptografia de chave simétrica: as chaves de cifra e de decifra são idênticas

Criptografia de chaves assimétrica ou de chave pública: cifra-se com a chave pública, decifra-se com a chave privada do receptor, ou vice-versa

Dicionário da Academia das Ciências (2001)

- Cifra – Códigos usados para esconder mensagens, chaves desses códigos, ...
- Cifrado – Mensagem codificada, ...
- Cifrar – Acto de escrever em código secreto, codificar, ...

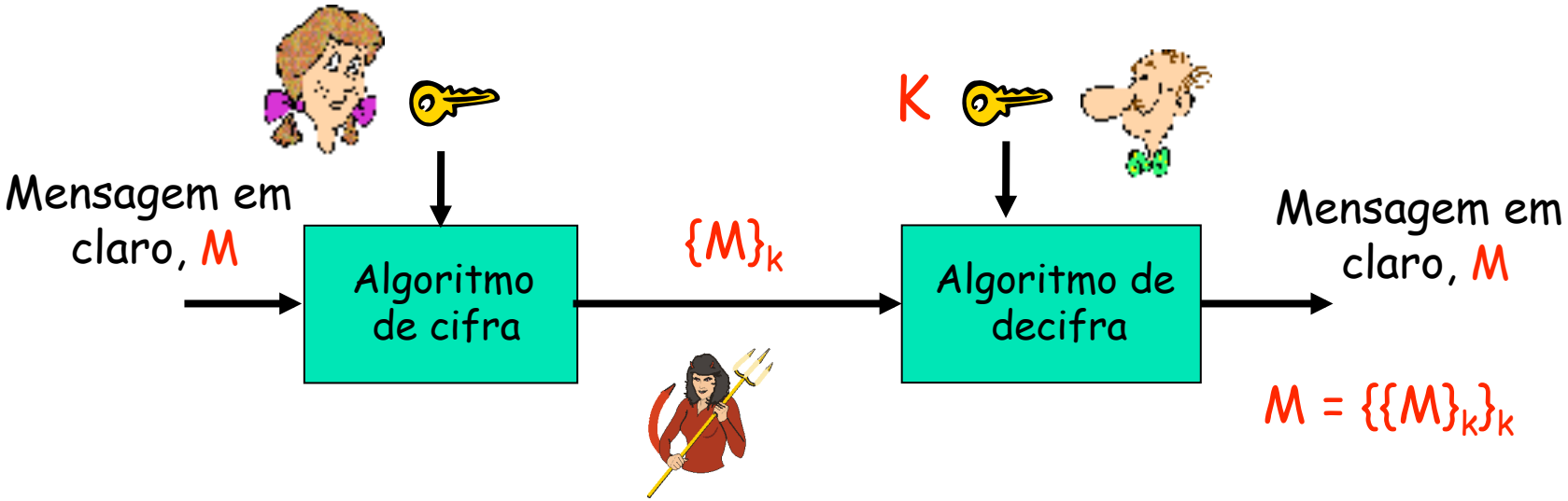
- Cripta – gruta, caverna, local secreto, ...
- Críptico – que diz respeito a uma cripta
- Cript(o) como prefixo – exprime noção de oculto
- Criptogénese – estado daquilo cuja origem é oculta

- Criptografia – Conjunto de técnicas que permitem esconder o conteúdo das mensagens
- Criptográfico – o que é relativo à criptografia
- Criptograma – Texto ou documento escrito em código

- Decifração – Acto ou efeito de decifrar, leitura de uma mensagem cifrada, ...
- Decifrar – Conseguir ler uma mensagem cifrada
- Decifrável – Aquilo que pode ser decifrado

- Criptar, encriptar, desencriptar, ... – NÃO EXISTEM !

Criptografia de chave simétrica



- A Alice e o Bob partilham uma chave criptográfica, K
- Os algoritmos de cifra e de decifra usam a mesma chave
- Notação: $\{M\}_k$=Cifra(M,K) M=Decifra($\{M\}_k$) ou $M = \{\{M\}_k\}_k$
- É frequente o algoritmo cifrar ou decifrar de cada vez uma sequência de bits da dimensão da chave, pelo que é aplicado repetidamente até cifrar todos os dados

Exemplo: criptografia de chave simétrica

Cifra por substituição mono alfabética: trocar uma letra por outra

plaintext: abcdefghijklmnopqrstuvwxyz

ciphertext: mnbvcxzasdfghjklpoiuytrewq

Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc

Como descobrir a chave ?

Eficácia da criptografia

O algoritmo criptográfico será tanto melhor quanto mais difícil for, sem se conhecer a chave, obter o texto original a partir do texto cifrado. A eficácia de um ataque depende de dois factores:

- Algoritmo de cifra
- Domínio da chave, isto é, número de bits da chave

Métodos de ataque:

Cripto analíticos - baseiam-se nos métodos matemáticos utilizados em criptografia

Força bruta - baseiam-se na exploração sistemática de todas as chaves possíveis

Nenhum algoritmo criptográfico é inteiramente seguro se o número de bits da chave tornar um ataque "força bruta" realista no quadro de dois factores:

- O "valor" da informação
- A capacidade computacional do atacante

Ataques força bruta

Se uma chave tiver 128 bits então há cerca de $3,4 \times 10^{38}$ combinações.

Se for possível arranjar um bilião (10^9) de computadores, capazes de testarem um bilião de chaves por segundo cada um, é possível testar 10^{18} chaves por segundo. Neste caso são necessários 10^{13} anos para completar o ataque a uma chave de 128 bits.

Estima-se que a idade do universo é de cerca de 10^{10} anos !

No entanto, se a chave tiver 54 bits, então há apenas cerca de 6×10^{16} combinações. Com o mesmo poder computacional (teste de 10^{18} chaves por segundo), é possível testar todas as hipóteses em menos de 1 segundo.

Conclusão: chaves geradas de forma aleatória perfeita, com um número de bits suficientemente grande, são relativamente seguras quando sujeitas a ataques do tipo "força bruta".

A teoria e a prática

A análise anterior pressupõe que há uma distribuição perfeita da probabilidade de se ter seleccionado uma dada chave em todo o universo possível. Na prática tais geradores perfeitos de chaves não existem e o atacante pode, conhecendo alguma das fraquezas do gerador de chaves, ir analisar um espaço de pesquisa muito mais reduzido.

O Método de ataque mais fácil consiste em tentar "advinhar" a semente usada.

Por outro lado, certas chaves são inadequadas e são facilmente quebradas conhecendo o algoritmo usado e por isso não podem ser usadas.

Ou seja, na prática, a geração de chaves adequadas é um problema muito delicado e os ataques de força bruta quase nunca são realizados de forma "cega" a todo o universo de chaves possíveis.

Conclusão: é necessário ter uma atitude "cautelosa" face a esta questão.

Confiança nos algoritmos criptográficos

Não é possível provar matematicamente que um algoritmo criptográfico não tem falhas capazes de o tornar frágil perante ataques cripto analíticos. Só se consegue provar exactamente o contrário, através de exemplos.

A segurança de um método é pois baseada em o mesmo ser público e estar sujeito à critica e análise por especialistas.

A ciência criptográfica é uma disciplina "altamente especializada".

Algoritmos de criptografia simétrica mais comuns

DES - *Data Encryption Standard* - chaves com 56 bits. Está a cair em desuso pois a chave é demasiado pequena para a potência de cálculo actual. É o método mais antigo. Está a cair em desuso.

Triple DES - DES reforçado - chave com 128 bits.

IDEA - International Data Encryption Algorithm - chave com 128 bits. Método desenvolvido na Europa. Após vários anos de utilização e divulgação pública não se conhecem ataques com êxito.

AES - Advanced Encryption Standard - Nova norma U.S.A. Definida por concurso público. Admite chaves de 128, 192 ou 256 bits. É o método que está em início de generalização.

Como evitar que o receptor fique "baralhado"

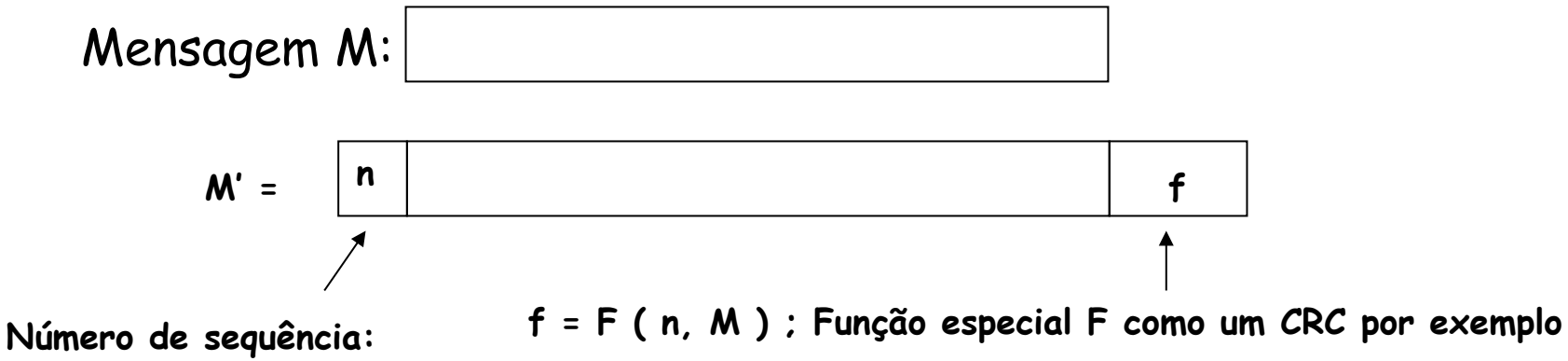
Se as mensagens a cifrar só contiverem bits significativos e válidos para as transacções em curso, a sua decifra com uma chave errada conduz apesar disso a um padrão de bits que pode ter significado (errado) para o receptor.

Este facto pode ser usado para levar o receptor a executar "disparates" mas também, em casos particulares, pode ser uma fonte de insegurança. Pelo menos trata-se de uma técnica que pode ser usada como base para ataques de impedimento de prestação de serviço tentando saturar o receptor ao processar dados aleatórios.

Por este motivo, é necessário introduzir bits redundantes antes de cifrar as mensagens que impeçam o receptor de ficar "baralhado". As soluções mais simples são:

- colocar um CRC
- colocar um número de ordem
- usar um campo MAC — "*Message Authentication Code*" (ver adiante)
- colocar um valor fixo pré-estabelecido, etc.

Exemplo



M' é a mensagem que é cifrada e transmitida. O receptor recebe M' cifrada e decifra o primeiro bloco. Começa logo por controlar se o mesmo contem um número de sequência válido. Se não contiver, ignora o resto da mensagem; senão considera-a mas volta a controlar o valor de f no fim.

Cifra por blocos encadeados

Um algoritmo de cifra por blocos pode revelar-se frágil na medida em que cada bloco é cifrado de forma independente, o que pode revelar padrões repetidos que facilitem a relação do texto cifrado com o texto em claro e facilita o ataque por métodos cripto analíticos.

Uma técnica possível para melhorar um método de cifra por blocos é usar cifra por blocos encadeados em cadeia (CBC - *cipher block chaining*).

Como a aplicação duas vezes seguidas na operação XOR é idempotente, faz-se o seguinte: durante o processo de cifra cada bloco, antes de o cifrar, o mesmo é *XORed* com o resultado da cifra do anterior; durante a decifra, cada bloco é decifrado, e *XORed* com o anterior cifrado.

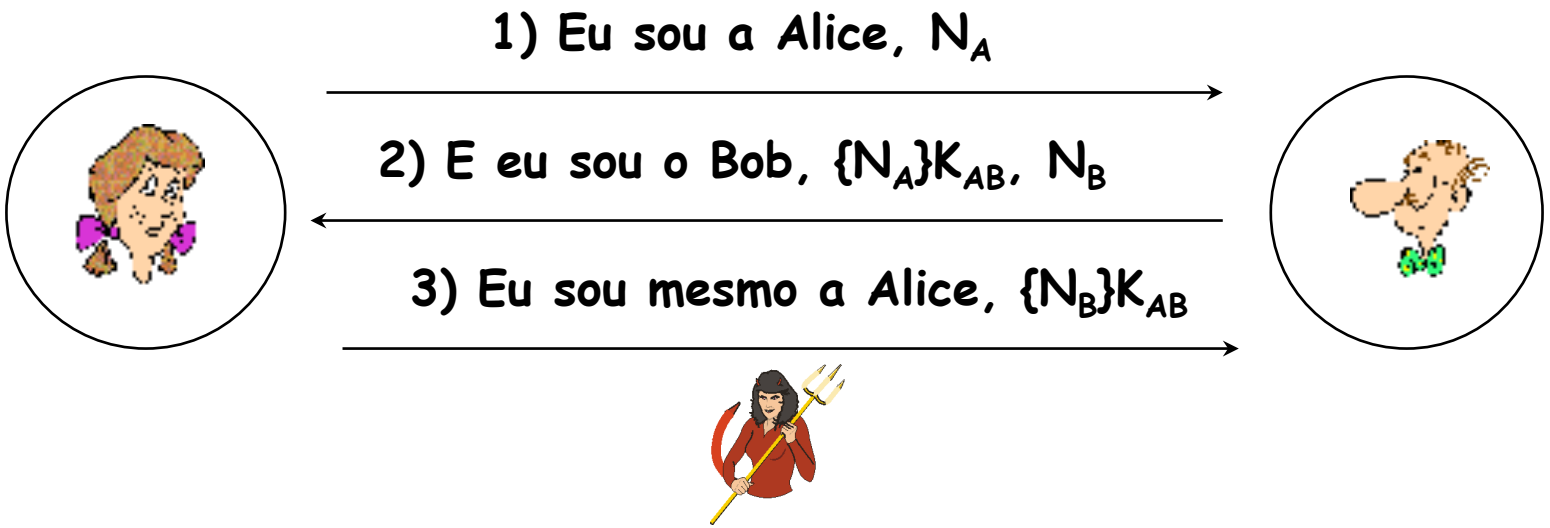
No início usa-se um vector de inicialização (por exemplo uma *timestamp* da dimensão de um bloco) para iniciar o processo. Desta forma, o mesmo texto, cifrado com a mesma chave, mas com vectores iniciais distintos, conduz a resultados distintos.

Esta técnica só se pode aplicar em canais fiáveis pois a perca de uma mensagem impede o processo de decifra.

Autenticação pelo método desafio / resposta

Bob e Alice têm de conhecer previamente uma chave secreta (K_{AB}) que partilham. Essa chave não pode nunca passar em claro na rede.

Este método permite autenticar e evitar o ataque por *replaying*. Garante-se a "*frescura*" da transacção, pois os números N_A e N_B são números aleatórios gerados no momento e que "nunca mais" devem ser reutilizados (costumam designar-se por "*nonces*" — *once-in-a-lifetime*).



O Problema da chave partilhada

- A Alice e o Bob têm de ter uma chave secreta partilhada
- Para aumentar o grau de segurança, idealmente essa chave devia ser diferente cada vez que falam entre si
- Como obter a nova chave de cada vez ?
- Este problema é ainda mais delicado se admitirmos que a Alice e o Bob vão ter um primeiro encontro e que não se conheceram antes (que é o caso de um comprador e um fornecedor normalmente).

- Um método possível é usarem uma terceira parte em quem confiam e com a qual partilharam previamente chaves

Distribuição de chaves de sessão através de um KDC

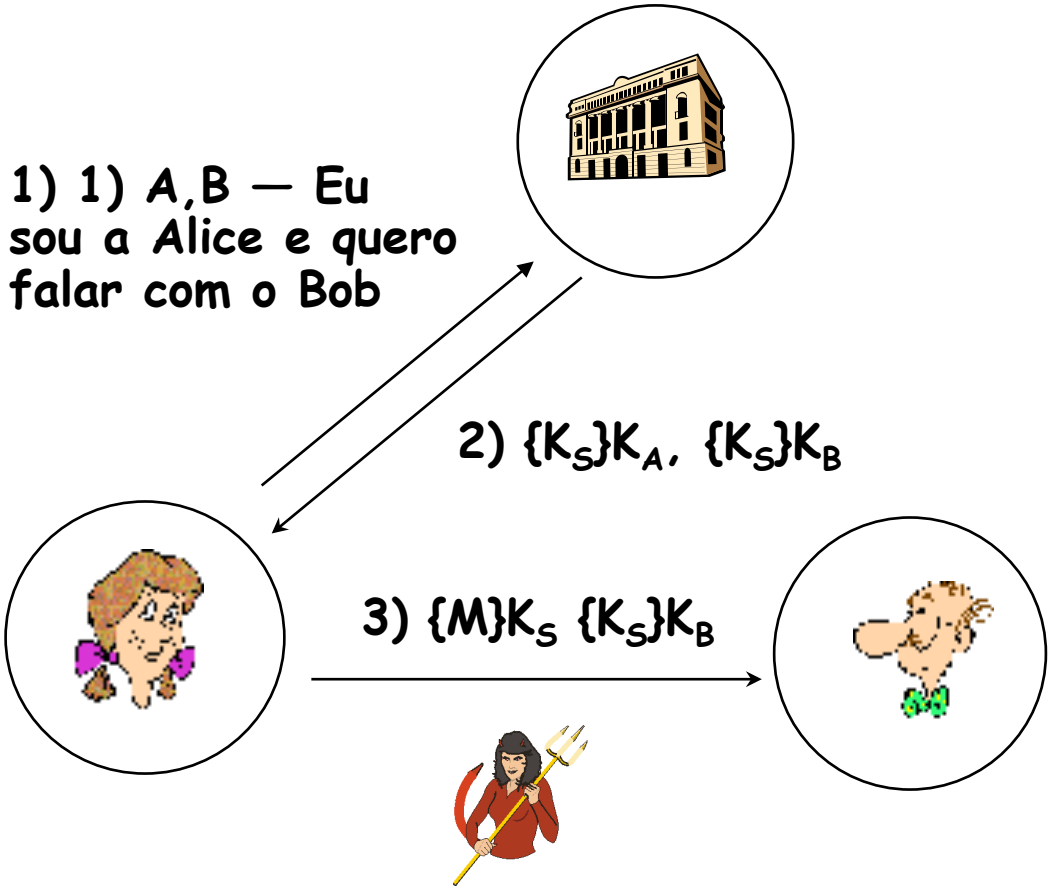
KDC - Key Distribution Center

K_S - chave de sessão, válida apenas para esta transacção

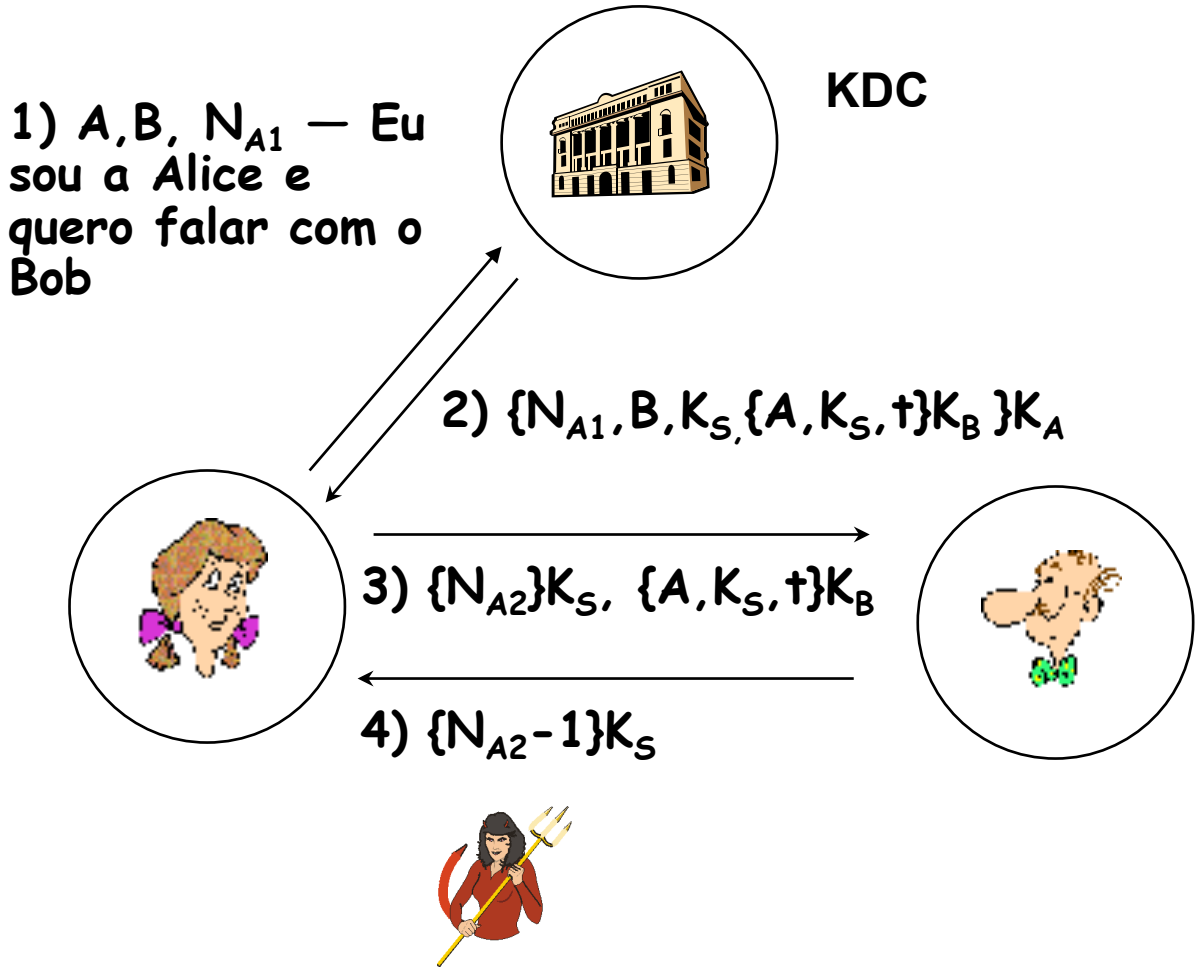
K_A, K_B - chaves da Alice e do Bob apenas partilhadas com o KDC

O protocolo permite que a Alice e o Bob adquiram uma chave comum para falarem, mesmo sem nunca se terem encontrado antes, que é a situação mais geral.

Tudo repousa sobre uma terceira parte da confiança de ambos e com a qual partilham chaves secretas.



Protocolo de Needham-Schroeder



Explicações

1) A,B, N_{A1}

Eu sou a Alice e quero falar com o Bob, dá-me um "*ticket*" para eu me autenticar perante ele. N_{A1} é um número aleatório gerado pela Alice que garante a "*unicidade*" da transacção, visto que o valor N_{A1} é único e não passou ainda em mensagens anteriores. t é uma marca temporal que permite marcar a época em que o *ticket* foi gerado. Ver a nota abiaxo.

2) $\{N_{A1},B,K_S,\{A,K_S,t\}K_B\}K_A$

Aqui estão N_{A1}, K_S e um *ticket* cifrados com a tua chave Alice. K_S não passa em claro e N_{A1} permite à Alice reconhecer o KDC pois só ele conhece K_A (para além dela). A Alice não pode modificar o *ticket* pois este está cifrado com a chave do Bob.

3) $\{N_{A2}\}K_S$, $\{A,K_S,t\}K_B$

A Alice diz ao Bob que quer falar com ele e manda-lhe o *ticket* e um desafio.

4) $\{N_{A2}-1\}K_S$

O Bob prova que é Bob pois foi capaz de decifrar $\{N_{A2}\}K_S$ o que pressupõe que decifrou o *ticket* $\{A,K_S,t\}K_B$. Bob ao reconhecer um bom *ticket* gerado para que a Alice e ele pudessem obter uma chave comum, sabe que está a falar com a Alice pois esta foi capaz de obter o *ticket* que lhe tinha sido enviado cifrado com a chave dela pelo KDC. Quando muito a Alice estaria a usar um *ticket* válido mas antigo o que seria completamente estúpido da parte dela e é detectado pela marca temporal t.

Nota: O *ticket* contém uma estampilha horária (t) e pressupõe que o KDC, a Alice e o Bob têm relógios sincronizados a menos de uma diferença considerada suficiente para que durante esse período de tempo a chave K_s não possa vir a ser comprometida. Com efeito, se um dia mais tarde K_s viesse a ser comprometida, o antigo *ticket* permitiria a Trudy autenticar-se junto de Bob como se fosse a Alice.

Criptografia assimétrica

A criptografia assimétrica foi inventada para permitir que os parceiros A e B em diálogo possam estabelecer entre si uma chave ou senha de sessão e autenticarem-se, sem necessidade de recurso a uma terceira parte de confiança (*Key Distribution Center*)

O Objectivo é reduzir ao mínimo, o papel da *Trusted third part* e agilizar o processo.

Baseia-se em que cada principal tem duas chaves (relacionadas entre si), uma designada chave secreta e outra chave pública. Por esta razão também se designa este método por criptografia baseada em chaves públicas.

Criptografia de chave pública

São dadas as funções públicas Cifrar e Decifrar (algoritmos para cifrar e decifrar) e uma função ChavePublica (algoritmo que permite gerar uma chave pública a partir de uma chave secreta) com certas propriedades.

Alice gera aleatoriamente a sua chave secreta (K_{ASec}). Depois calcula a sua chave pública K_{APub} = ChavePublica (K_{ASec}).

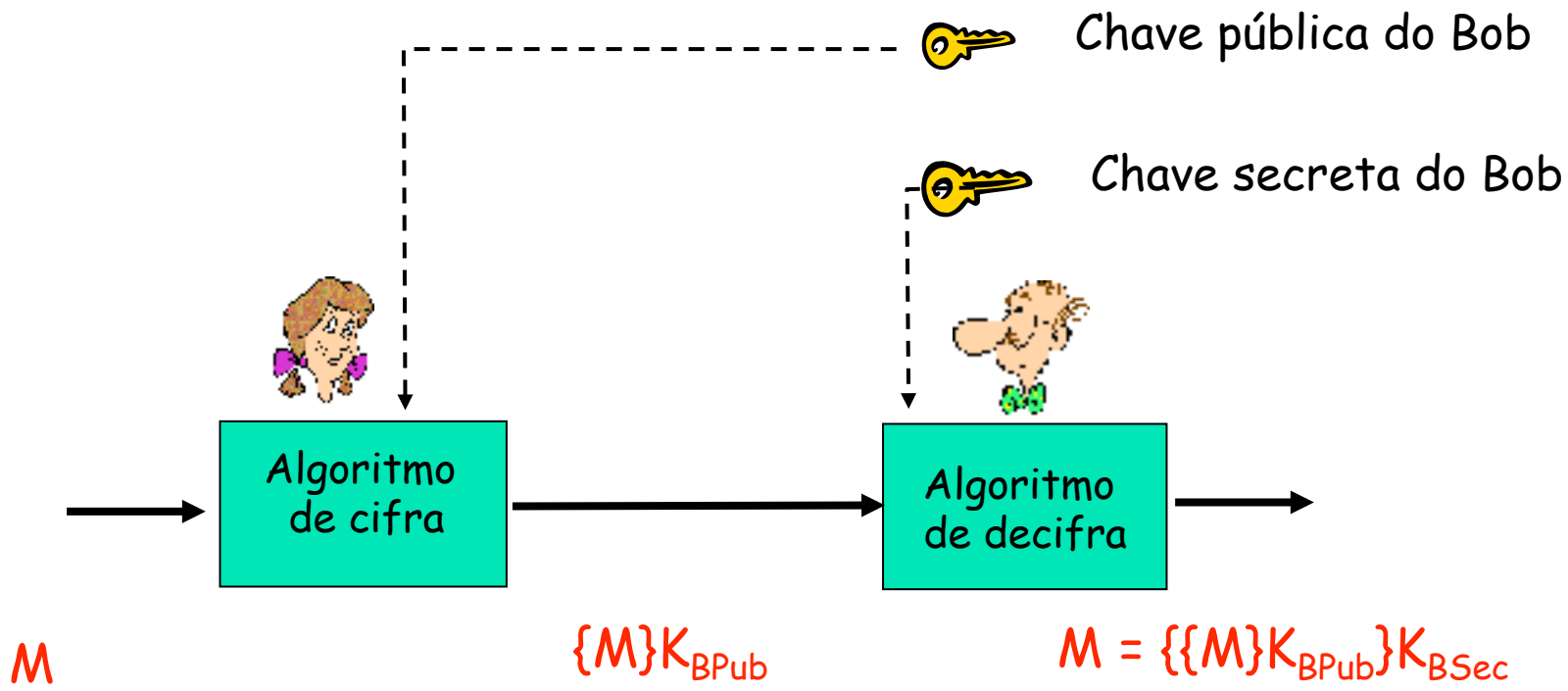
Com este método tem-se então:

Decifrar (Cifrar (M, K_{APub}), K_{ASec}) = M $M = \{\{M\}K_{BPub}\}K_{BSec}$

Cifrar (Decifrar (M, K_{ASec}), K_{APub}) = M $M = \{\{M\}K_{BSec}\}K_{BPub}$

Isto é, se a Alice conhecer a chave pública do Bob (K_{BPub}) e cifrar M com essa chave, então o Bob, e só Bob, será capaz de ler M decifrando com a sua chave secreta (K_{BSec}), e vice-versa. A utilização das duas chaves e das duas funções distintas (Cifrar e Decifrar) justifica também a designação destes métodos por métodos de criptografia assimétrica.

Como pode ser usada



Se a Alice conhecer algo que é público (a chave pública de Bob), pode enviar-lhe uma mensagem secreta que só ele pode decifrar, sem necessidade de nenhuma outra chave.

O que é necessário para que funcione

1) $M = \{ \{ M \} K_{Pub} \} K_{sec}$ e $M = \{ \{ M \} K_{sec} \} K_{Pub}$ isto é, funções com aquelas propriedades

2) Se for usado um número adequado de bits nas chaves, dada K_{Pub} é impossível obter K_{sec} com o poder *computacional actualmente disponível*, ou seja, é computacionalmente impossível inverter a função de geração da chave pública a partir da secreta.

3) Dados $\{ M \} K_{Pub}$ e K_{Pub} é computacionalmente impossível obter M, ou seja, é computacionalmente impossível inverter a função de cifra.

4) Dados M e $\{ M \} K_{sec}$ e verificando-se $M = \{ \{ M \} K_{sec} \} K_{Pub}$ é computacionalmente impossível obter $M_1 \neq M$ tal que $M = \{M_1\}K_{Pub}$ isto é, não é computacionalmente possível gerar um padrão aleatório, que decifrado com a chave pública, conduza a uma mensagem com um significado semântico específico.

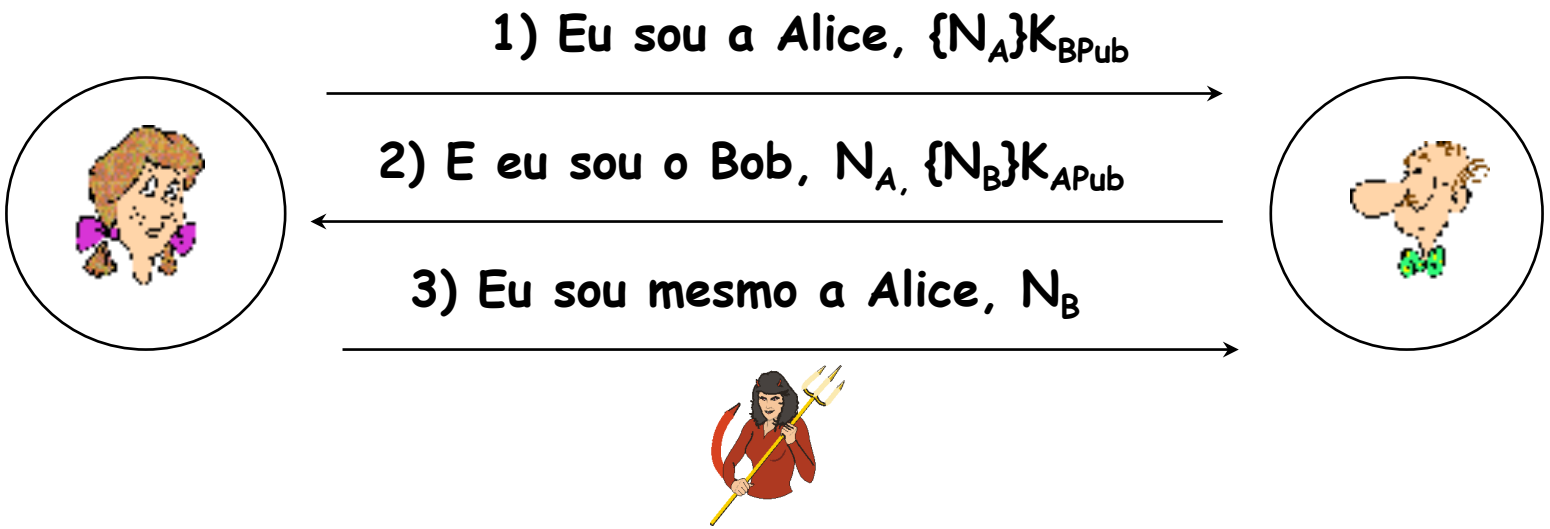
Que algoritmos satisfazem este requisitos ?

- Existem vários métodos e algoritmos que satisfazem os requisitos anteriores
- Um dos mais conhecidos é designado pela sigla RSA (do nome dos seus inventores: Rivest, Shamir e Adelman)
- Cifra usando as operações resto da divisão inteira e exponenciação com chaves com 512, 768, 1024, 2048, ... bits ou mais
- A dificuldade de inverter as funções advem de até ao momento ser computacionalmente impossível factorizar números de grande dimensão em tempo útil

Autenticação através do método RSA

Se o Bob e a Alice conhecerem previamente as chaves públicas um do outro, podem executar o método de desafio / resposta, exigindo ao outro que provem que conhecem a chave secreta associada à respectiva chave pública. Um exemplo poderia ser o seguinte.

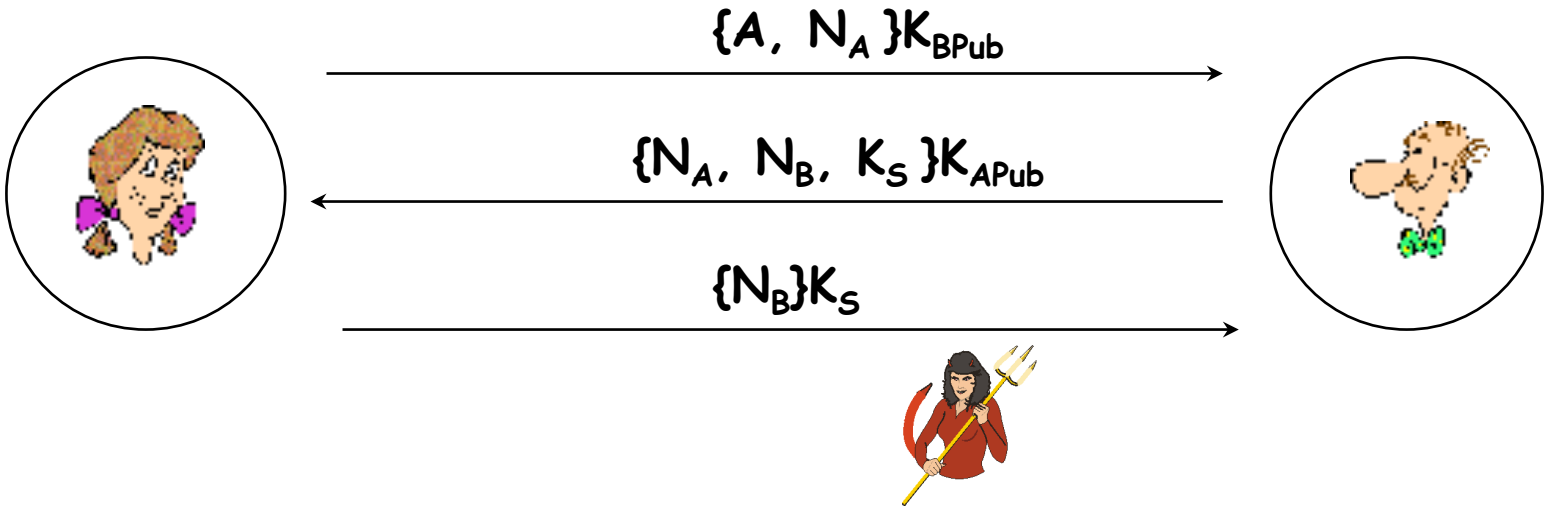
N_A e N_B são "*nonces*"



Comparação face à criptografia simétrica

Se se um usar um Public Key Center (KDC) para registar as chaves públicas, o KDC apenas conhece aquilo que é público, isto é, as chaves públicas das entidades. Tal é um progresso significativo pois o papel da *terceira parte de confiança* foi drasticamente reduzido.

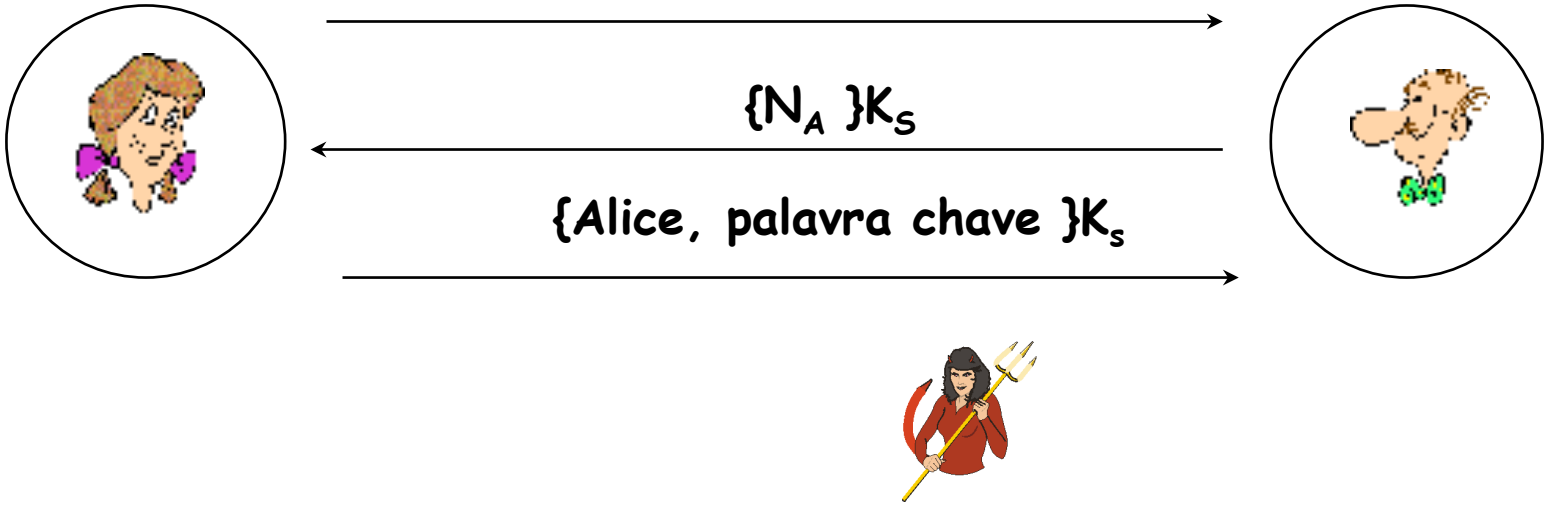
Em contrapartida os algoritmos são, em geral, duas a três ordens de grandeza mais lentos a cifrarem e a decifrarem. Por esta razão, os dois métodos são em geral usados em conjunto. A criptografia assimétrica é usada para autenticar e passar uma chave simétrica de sessão gerada por um dos parceiros. Por exemplo, o protocolo de Needham-Schroeder com autenticação via chaves públicas ficaria:



Alternativa utilizada frequentemente

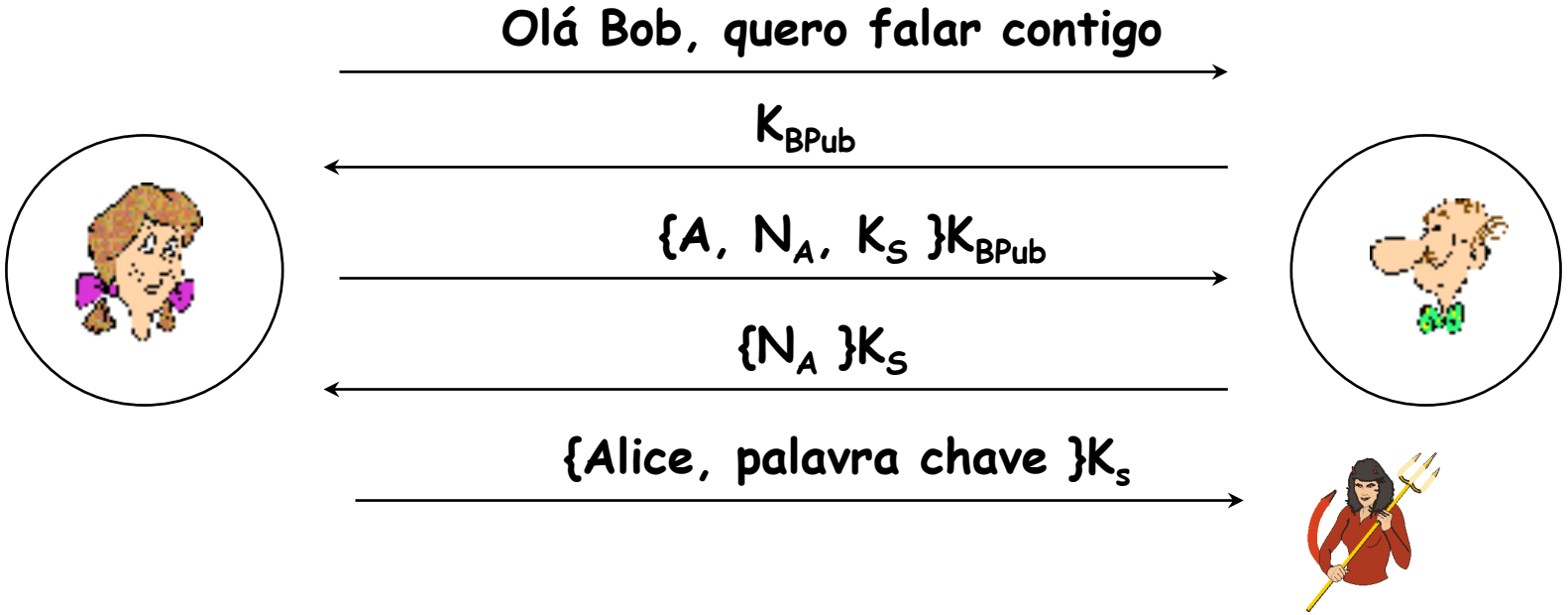
Muitas vezes o servidor (por exemplo o Bob) conhece uma palavra chave convencional do utilizador (por exemplo a Alice) e pode usá-la para o autenticar, por um canal cifrado em que o utilizador já tem a certeza que está a falar com o verdadeiro servidor (de qual conhece a chave pública).

Neste caso o protocolo pode ser mais simples ainda.



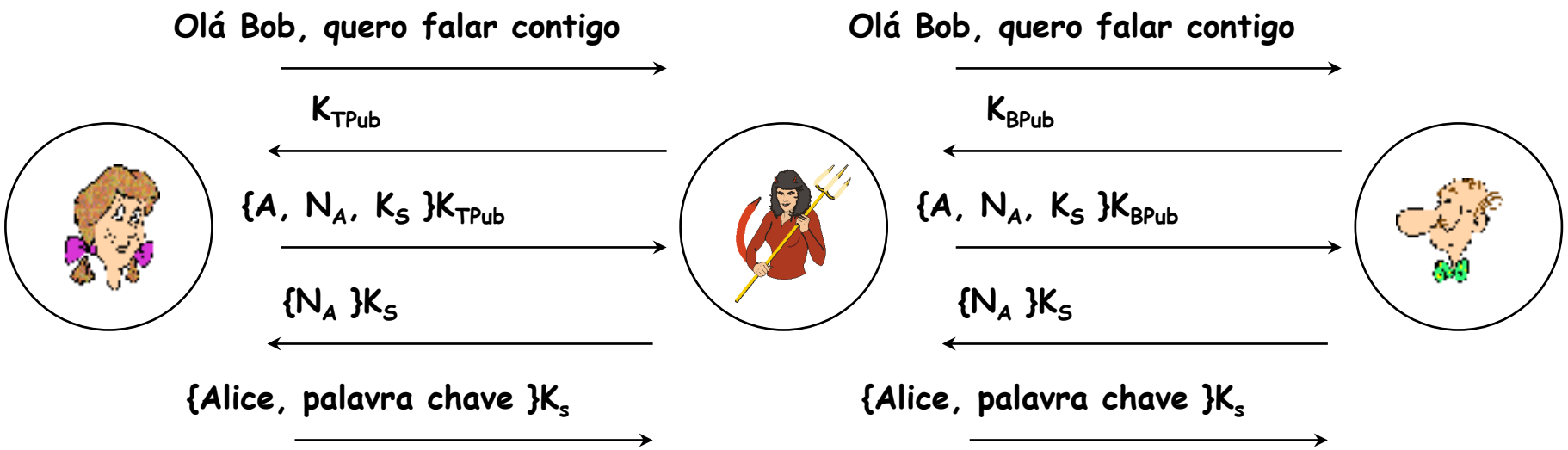
Podemos dispensar o PKC ?

Se a Alice não conhecer a chave pública do Bob, poderia simplesmente pedir ao próprio Bob que lha desse visto que a mesma é pública. Exigindo-lhe apenas que provasse que ele conhecia a chave secreta que lhe está associada.



O Problema é que se a Trudy conseguir colocar-se no endereço em que a Alice pensa que o Bob está, este método permite à Trudy fazer-se passar pelo Bob. Logo, as chaves públicas têm mesmo que ser as certas.

Ataque por interposição



Ese ataque designa-se por ataque por interposição ("man ou woman in the middle"), é muito difícil de detectar e permite que mais tarde a Trudy se apresente perante Bob como sendo a Alice pois apanhou-lhe a palavra chave. Repare-se que a Trudy executa um protocolo genérico aplicável em qualquer situação aplicacional.

Distribuição de 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 nós conhecermos a chave pública dessa fonte fidedigna, uma forma de obter esta confiança é essa fonte cifrar a sua resposta com a sua chave secreta por exemplo. Desta forma sabemos que estamos a obter a chave pública do "sítio certo", isto é, do verdadeiro *Public Key Center* ou de um seu substituto fidedigno.

Métodos de distribuição de chaves:

Certificate Granting Authority - cujas chaves públicas são bem conhecidas e que "assinam" os certificados de chaves públicas 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 parceiros confiáveis entre si. Este método tem sido vulgarizado pelo programa PGP para troca de correio electrónico.

Exposição das chaves secretas assimétricas

Se as chaves públicas e secretas apenas forem usadas para a autenticação e troca de uma chave simétrica de sessão com o outro parceiro, então:

1) As chaves públicas estão sempre expostas, mas

2) as chaves secretas só são usadas para decifrar as mensagens iniciais devendo ser apagadas da memória imediatamente a seguir

3) a chave de sessão tem a validade dessa sessão e nunca mais é reutilizada

As chaves secretas assimétricas são geralmente de grande dimensão pelo que não é prático memorizá-las; devem ser guardadas em suportes estáveis onde estão cifradas por sua vez através de uma palavra chave (sendo a palavra chave a chave de um processo simétrico de cifra da chave secreta).

Em alternativa a chave secreta pode estar gravada num cartão inteligente que cifra / decifra sem expor a chave secreta.

Assinaturas digitais baseadas em chaves públicas

Um sistema de assinaturas digitais deve ter as seguintes propriedades:

Autenticação: o receptor deve poder verificar que a assinatura é autentica

Integridade: a assinatura deve garantir que a mensagem assinada não foi alterada, nem durante o trajecto, nem pelo receptor, mesmo que tenha passado em claro

Não repudiamento: o emissor não poderá negar que de facto enviou a mensagem assinada

Como fazer ?

Para que Bob assine a mensagem M que emite, Bob deve juntar a M uma assinatura que pode consistir em cifrar M através da sua chave secreta.

Mensagem M assinada por Bob: $M,\{M\}K_{BSec}$

Quando a Alice receber $M,\{M\}K_{BSec}$ pode testar se se verifica ou não o seguinte:

$$M=\{M\ \{M\}K_{BSec}\}K_{BPub}$$

Isto é, decifrando $\{M\}K_{BSec}$ com a chave pública de Bob, se se obtiver M, isso prova que:

1) Bob enviou M por si assinada pois só Bob poderia ter gerado a assinatura

2) Bob enviou de facto M e não M' ≠ M

3) M foi enviada por Bob e não por outra pessoa que desconheça K_{BSec}

Assim, desde que se prove que a quando do envio da mensagem M, a chave de Bob era K_{BPub}, prova-se que Bob enviou exactamente M.

Funções de síntese

Como desta forma a assinatura de M tende a ter o mesmo número de bits que M, na prática utiliza-se uma função MD, dita *message digest* ou *função de síntese*. O emissor produz a assinatura de M, calculando: $\{MD(M)\}K_{BSec}$

A função MD, dita Função de Síntese, *Message Digest* ou *Secure Hash Function*, produz um resultado com 128, 160, 512, … bits e tem as seguintes propriedades:

1) É computacionalmente impossível criar M1 e M2 tais que MD(M1) = MD(M2) pois isso permitiria a falsificação pelo emissor.

2) Dada M' = MD(M1) é computacionalmente impossível criar M2 tal que M' = MD(M2) pois isso permitiria a falsificação pelo receptor.

3) Dado MD(M) é computacionalmente impossível calcular M.

4) Calcular MD(M) é computacionalmente fácil.

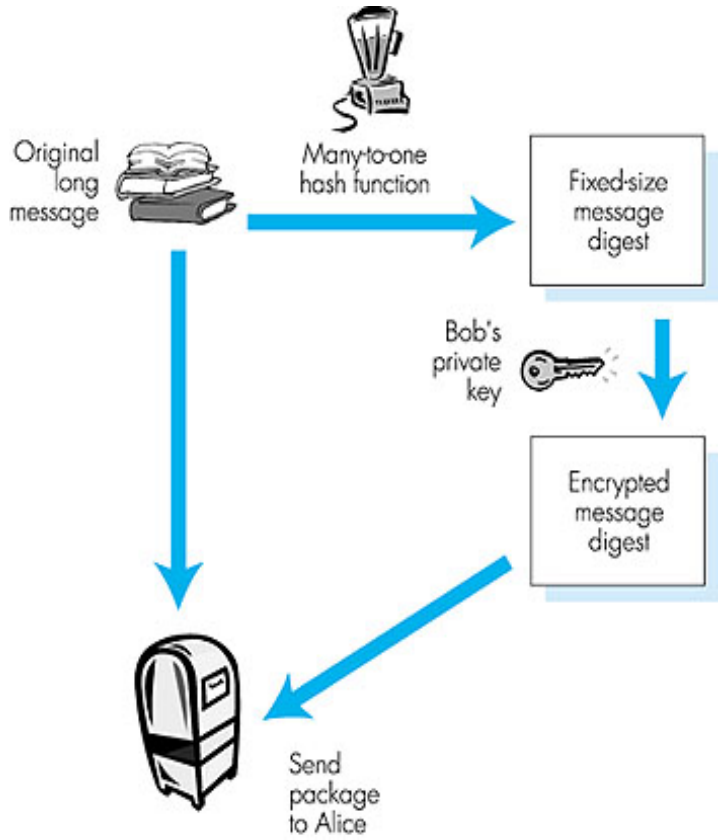
As funções de síntese com estas propriedades dizem-se "*Secure one-way hash functions*" ou "*funções de dispersão unidireccionais seguras*". Por se designarem por funções de *hash* seguras, utiliza-se às vezes a letra H para as denotar em vez de MD. O carácter seguro vem da propriedade 1), o carácter sem inverso vem das propriedades 2) e 3).

Funções seguras de síntese

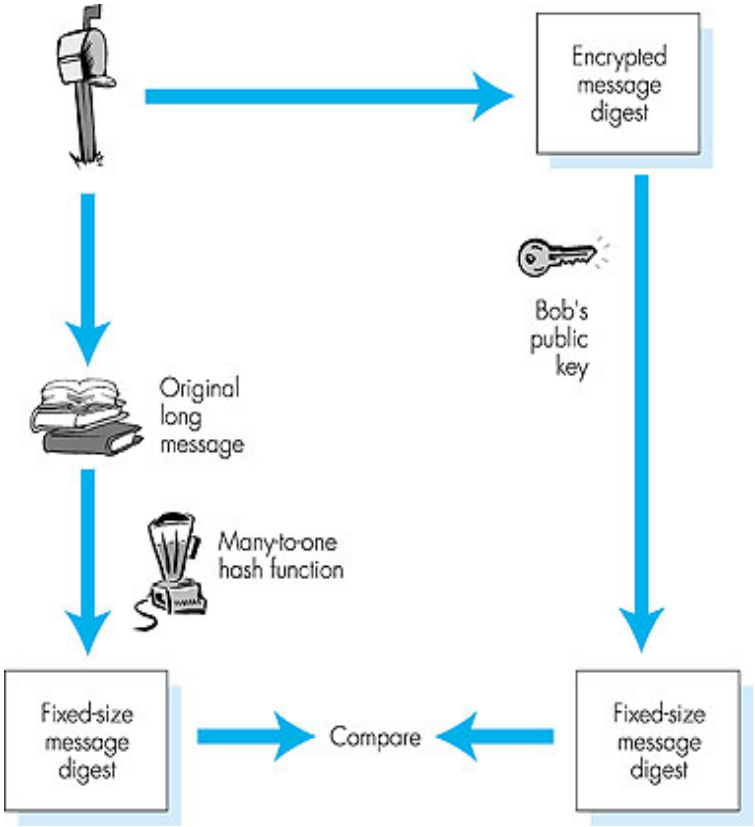
- **Função segura de síntese MD5**
  - **Calcula sínteses de 128 bits num processo em 4 fases.**
  - **Aplicada a uma mensagem M, devolve uma sequência arbitrária X de 128 bits, tal que é *computacionalmente impossível* construir uma outra mensagem M' cujo *hash* MD5 seja também igual a X.**
- **A função SHA-1 é também muito usada.**
  - **Norma USA. Conhecida publicamente.**
  - **Produz uma síntese de 160 bits**

Verificação da assinatura e da integridade

Alice envia uma mensagem com uma assinatura digital:



Bob verifica a assinatura e a integridade da mensagem:



Integridade das mensagens de um canal

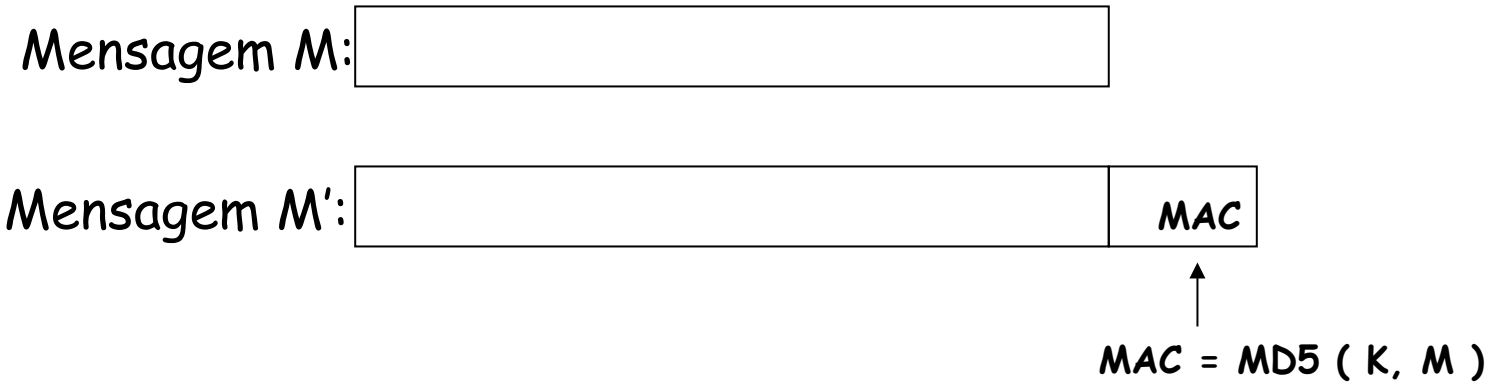
É possível usar as funções de síntese para acrescentar controlo de integridade num canal de dados. O método consiste em usar MACs ("*Message Authentication Codes*") que são assinaturas, computacionalmente fáceis de calcular, mas impossíveis de forjar, baseadas em chaves secretas. O método funciona assim:

1) A e B estabelecem uma chave secreta K só conhecida por ambos. K pode ser trocada a quando da autenticação por exemplo.

2) Cada mensagem M que A envia a B é concatenada com K (e eventualmente um número de sequência). Em seguida, calcula-se o MAC h = H (M, K) e junta-se a M, enviando M e h em claro pelo canal.

3) Ao receber M e h, B calcula o MAC h' = H (M, K) e verifica se h = h'. As propriedades das funções H permitem garantir que um MAC assim construído permite detectar a alteração de M.

Exemplo



M' é a mensagem que é transmitida. O receptor recalcula o MAC para detectar se a mensagem foi ou não alterada. O intruso não consegue modificar M sem que isso seja detectado pois não conhece K. Em geral M conterá também um número de sequência.

Assinaturas e certificados de chaves públicas

Como vimos, a segurança baseada em chaves públicas depende da validade das mesmas. A forma de evitar que estas sejam falsas, é obtê-las a partir de outras partes confiáveis.

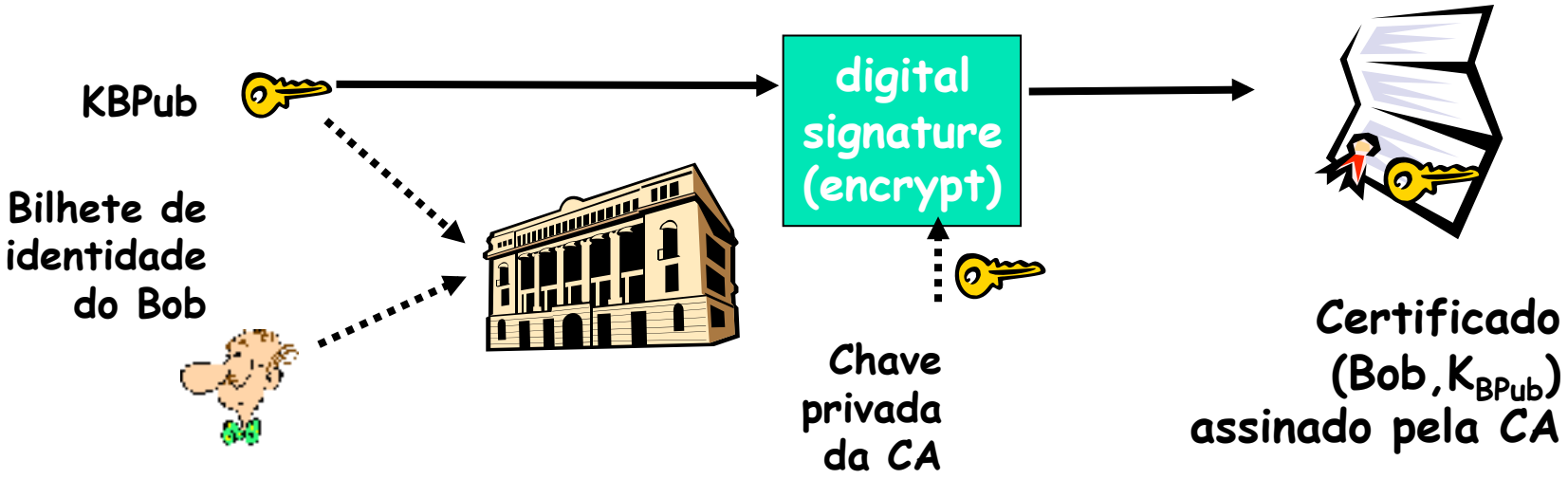
Em alternativa, podemos pedir a essas partes confiáveis que emitam certificados assinados declarando que a chave pública do Bob é K_{BPub} numa dada data.

Se podermos verificar a assinatura do certificado, e este ainda for válido (não caducou por passagem do tempo), então o Bob pode entregar-nos a chave e o certificado quando quisermos estabelecer uma sessão com ele.

Caso o certificado seja válido, isso impede os ataques por interposição ou por tentativa de falseamento de identidade.

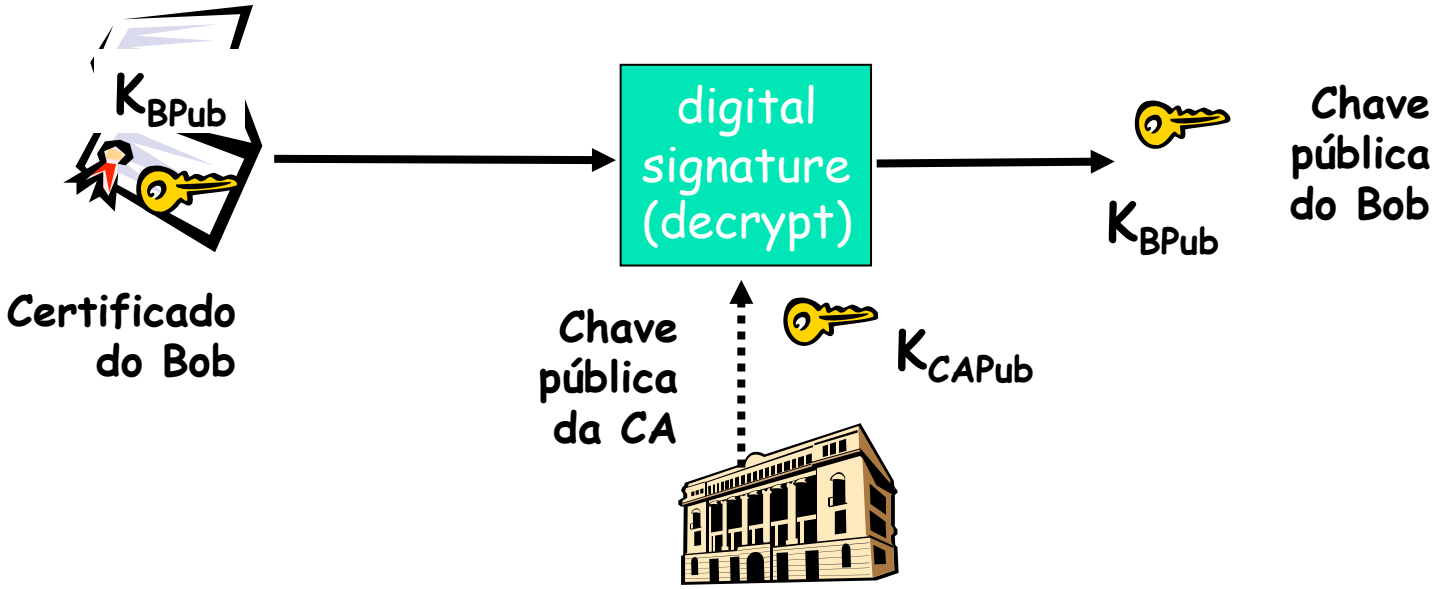
Autoridades de certificação

- **Certification authority (CA):** associa uma chave pública a uma entidade E.
- E (pessoa, router, ...) regista a chave na CA:
  - E mostra uma "prova de identidade" à CA.
  - A CA emite um certificado que associa E à sua chave.



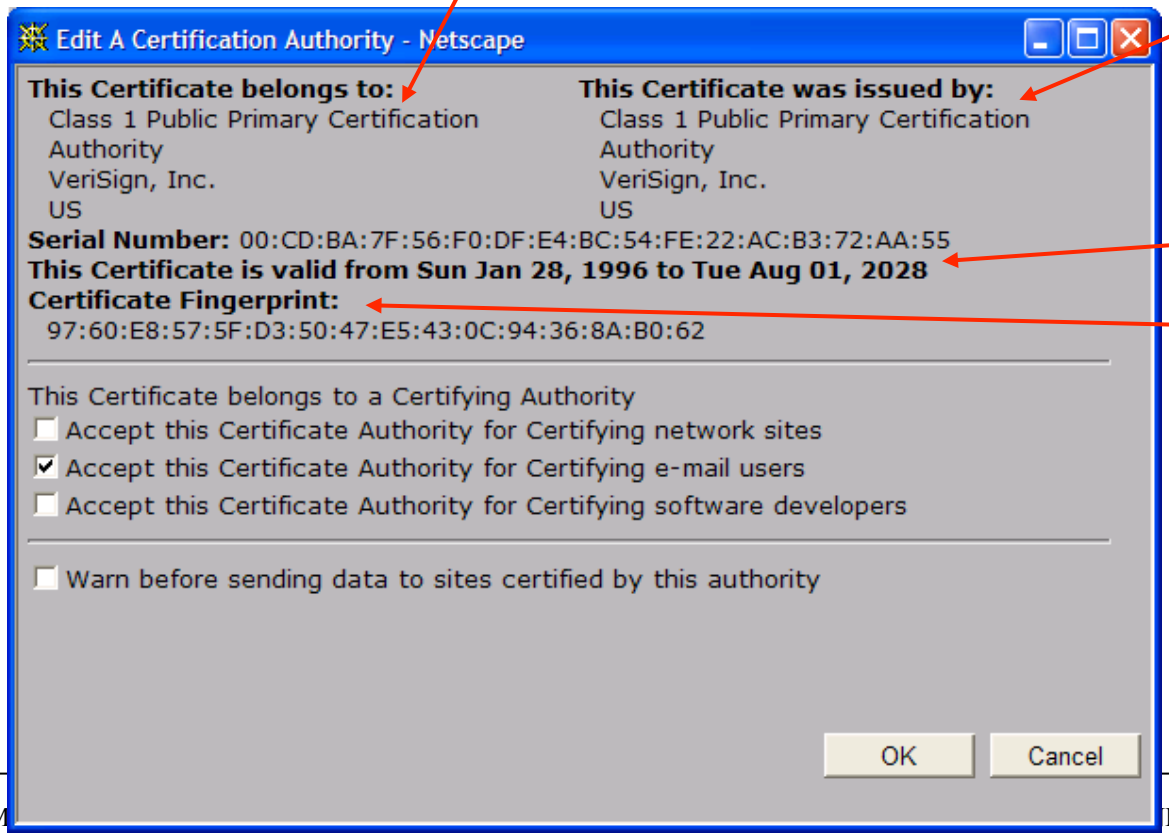
Verificação dos certificados

Quando a Alice recebe o certificado e a chave pública do Bob, pode verificar a validade da chave desde que tenha a chave pública da CA.



Formato normalizado de um certificado

- Número de série
- Informação sobre o dono do certificado
- Emissor (CA)
- Validade
- Assinatura digital



Repudiamento e validade de uma chave

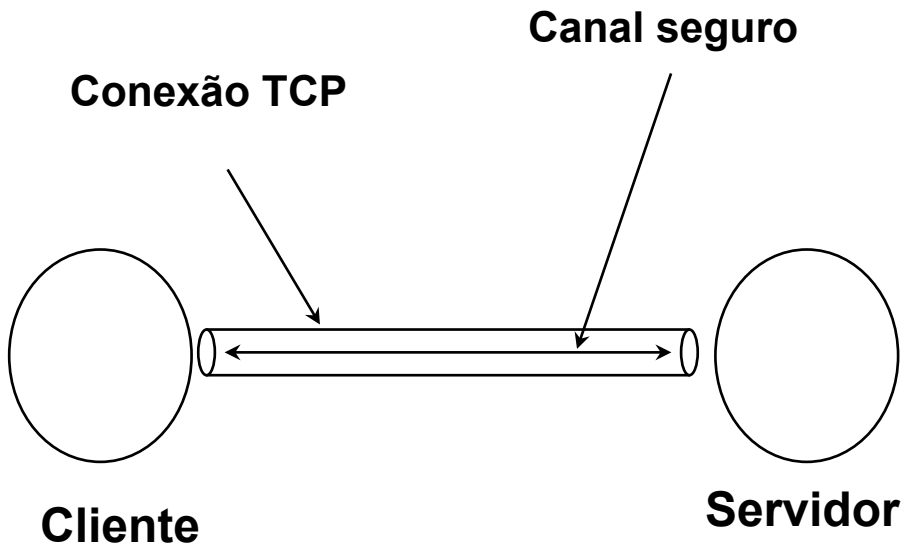
Se por qualquer motivo uma chave secreta for quebrada, é necessário revogar a chave pública que lhe estava associada. Dada a utilização das chaves públicas para assinaturas, é necessário que as terceiras partes de confiança (as autoridades de certificação por exemplo) memorizem as chaves válidas e listas de chaves revogadas.

Quando se pretende que a verificação da autenticidade da assinatura de uma mensagem seja viável num horizonte temporal alargado, a assinatura tem de ter uma estampilha temporal "t" incluída na assinatura. Para que tudo funcione correctamente é necessário que a autoridade de certificação ateste que no momento t (no passado) a chave pública da entidade era uma dada.

Assim, a utilização de certificados por si só não é uma panaceia universal.

SSL - Secure Socket Layer

Trata-se de um conjunto de protocolos proposto e desenvolvido pela Netscape, do tipo sessão, isto é sobre o nível transporte, que permitem estabelecer canais seguros e autenticar clientes e servidores. Constituí a base da norma IETF TLS - *Transport Layer Security*



Caracterização do SSL

- O protocolo SSL trabalha ao nível transporte. Requer um transporte orientado conexão como o TCP.
- É usado entre *browsers* e servidores WWW (https) por exemplo
- Funcionalidades de segurança:
  - autenticação do servidor
  - cifra e integridade dos dados
  - autenticação do cliente (opcional)
- Autenticação do servidor:
  - Um *browser* preparado para SSL inclui as chaves públicas de várias *CAs*
  - O *browser* solicita ao servidor um certificado emitido por uma *CA* em que ele confie.
  - O *browser* usa a chave pública da *CA* para extrair e verificar a chave pública do servidor.
- Vejam no vosso *browser* na secção de segurança.