Capítulo 8: Segurança em rede

REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

Objetivos do capítulo:

- entender os princípios de segurança em rede:
  - criptografia
  - confidencialidade
  - autenticação
  - integridade de mensagem
- segurança na prática:
  - firewalls e sistemas de detecção de invasão
  - segurança na camada de transporte

Capítulo 8: Esboço

REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

8.1 O que é segurança na rede?
8.2 Princípios de criptografia
8.3 Integridade de mensagem
8.5 Protegendo conexões TCP: SSL
8.8 Segurança operacional: firewalls e IDS

O que é segurança na rede?

Confidencialidade: apenas remetente e destinatário pretendido devem "entender" conteúdo da mensagem

- remetente criptografa mensagem
- destinatário decripta mensagem

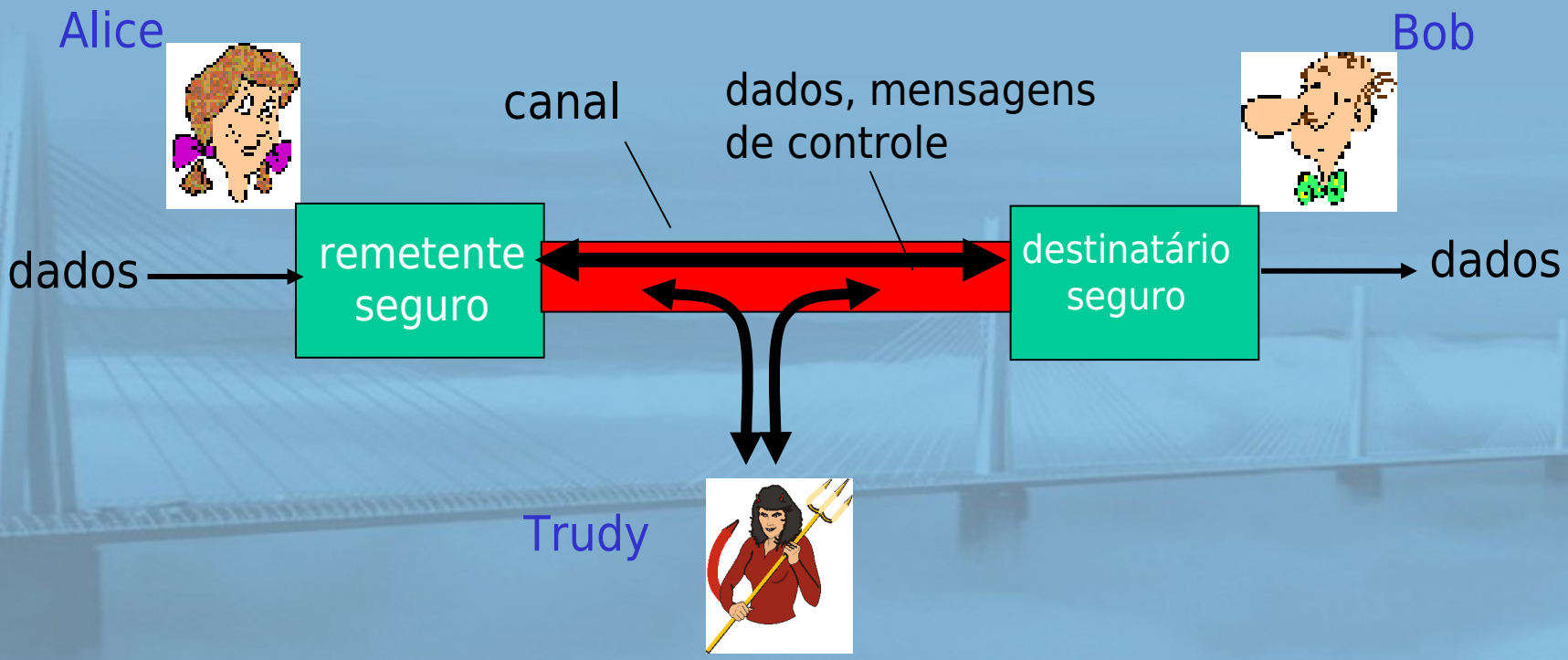
Autenticação: remetente e destinatário querem confirmar a identidade um do outro

Integridade da mensagem: remetente e destinatário querem garantir mensagem não alterada (em trânsito ou depois) sem detecção

Acesso e disponibilidade: serviços precisam ser acessíveis e disponíveis aos usuários

Amigos e inimigos: Alice, Bob, Trudy

- Bob, Alice (amigos!) querem se comunicar “com segurança”
- Trudy (intrusa) pode interceptar, excluir, acrescentar mensagens



Quem poderiam ser Bob e Alice?

REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

- ... bem, Bobs e Alices da vida real!
- navegador Web/servidor de transações eletrônicas (p. e., compras on-line)
- cliente/servidor de “Internet banking”
- servidores DNS
- roteadores trocando atualizações da tabela de roteamento

Existem perversos (e perversas) lá fora!

P: O que um “perverso” pode fazer?

R: Muita coisa!

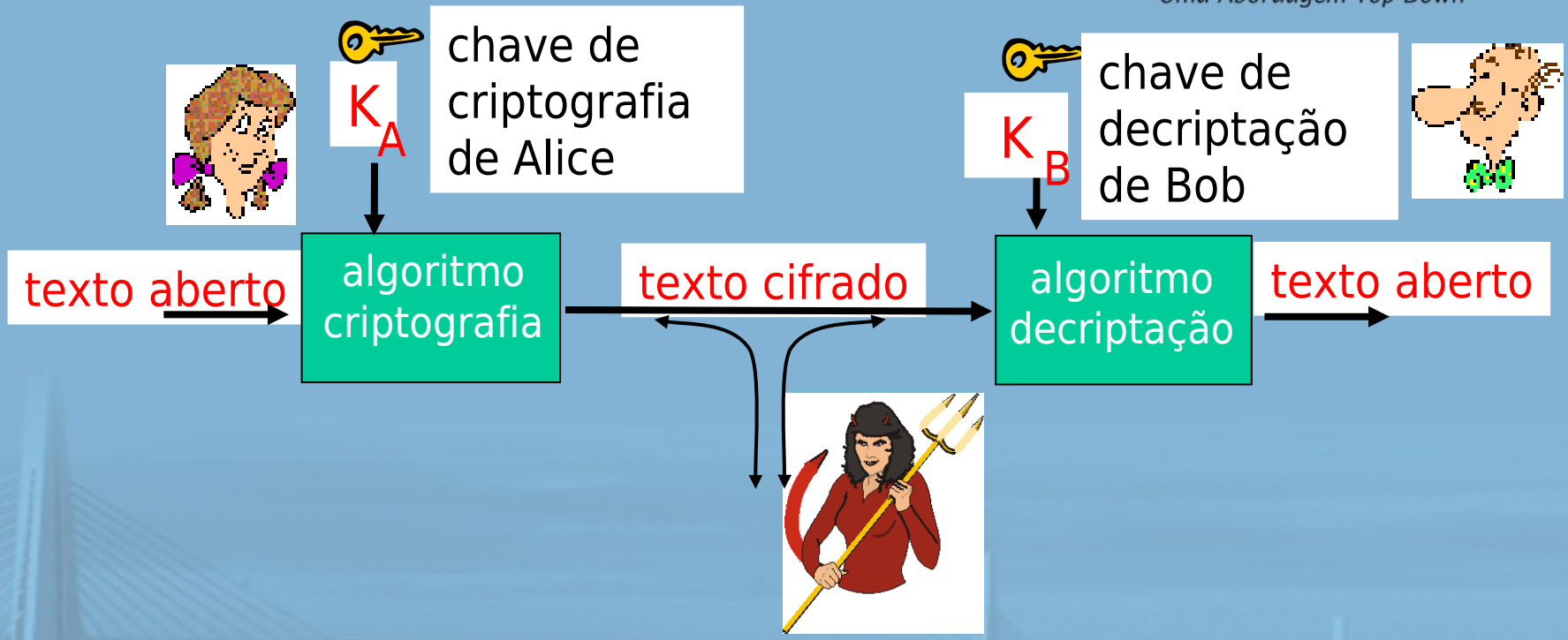
- *bisbilhotar:* interceptar mensagens
- *inserir* ativamente mensagens na conexão
- *personificação:* pode forjar (falsificar) endereço IP no pacote (ou qualquer campo no pacote)
- *sequestrar:* “apoderar-se” da conexão em andamento removendo remetente ou destinatário, inserindo-se no local
- *negação de serviço*: impedir que serviço seja usado por outros (p. e., sobrecarregando recursos)

Capítulo 8: Esboço

REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

8.1 O que é segurança na rede?
8.2 Princípios de criptografia
8.3 Integridade de mensagem
8.5 Protegendo conexões TCP: SSL
8.8 Segurança operacional: firewalls e IDS

A linguagem da criptografia



m mensagem em texto aberto

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

$m = K_B(K_A(m))$

Esquema de criptografia simples

cifra de substituição: substituir uma coisa por outra

- cifra monoalfabética: substituir uma letra por outra

texto aberto: abcdefghijklmnopqrstuvwxyz

texto cifrado: mnbvcxzasdfghjklpoiuytrewq

p. e.: **texto aberto: bob. i love you. alice**

texto cifrado: nkn. s gktc wky. mgsbc

Segredo: o mapeamento do conjunto de 26 a outro conjunto de 26 letras

Criptografia polialfabética

- n cifras monoalfabéticas, $M_1,M_2,\ldots,M_n$
- Padrão cíclico:
  - p. e., n = 4, M_1,M_3,M_4,M_3,M_2; M_1,M_3,M_4,M_3,M_2;
- Para cada novo símbolo de texto aberto, use padrão monoalfabético subsequente no padrão cíclico
  - dog: d de M_1, o de M_3, g de M_4
- Segredo: as n cifras e o padrão cíclico

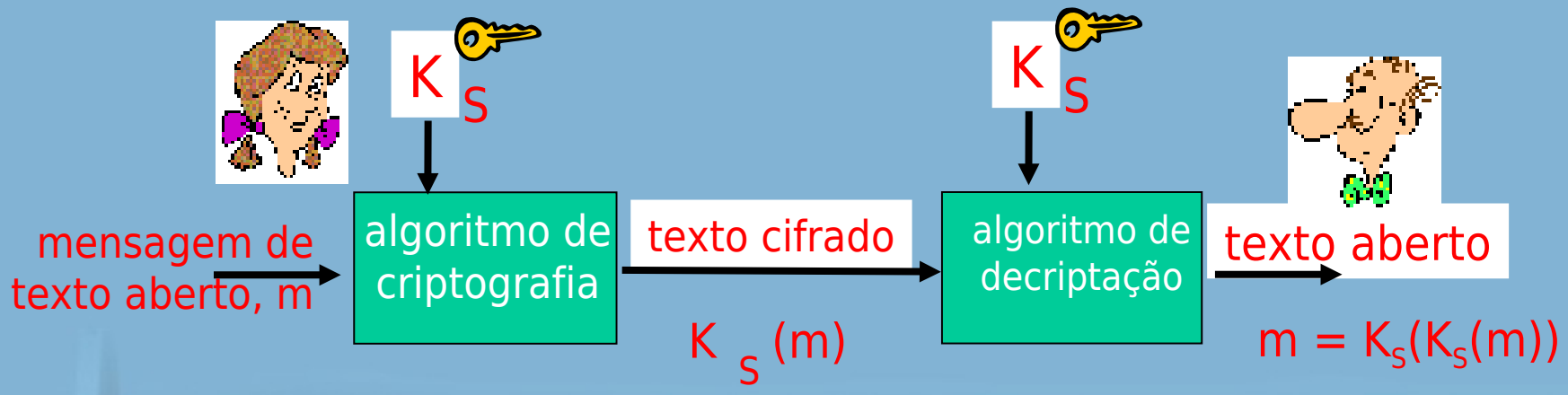
Quebrando um esquema de criptografia

- Ataque apenas a texto cifrado: Trudy tem o texto cifrado que ela pode analisar
- Duas técnicas:
  - Procura por todas as chaves: deve ser capaz de diferenciar texto aberto resultante do texto sem sentido
  - Análise estatística
- Ataque de texto aberto conhecido: Trudy tem algum texto aberto correspondente a algum texto cifrado
  - p. e., na cifra monoalfabética, Trudy determina pares para a,l,i,c,e,b,o,
- Ataque de texto aberto escolhido: Trudy pode conseguir o texto cifrado para algum texto aberto escolhido

Tipos de criptografia

- criptografia normalmente usa chaves:
  - algoritmo é conhecido de todos
  - somente “chaves” são secretas
- criptografia de chave pública
  - envolve o uso de duas chaves
- criptografia de chave simétrica
  - envolve o uso de uma chave
- funções de hash
  - não envolve o uso de chaves
  - nada secreto: Como isso pode ser útil?

Criptografia de chave simétrica



criptografia de chave simétrica: Bob e Alice compartilham alguma chave (simétrica) : K_S

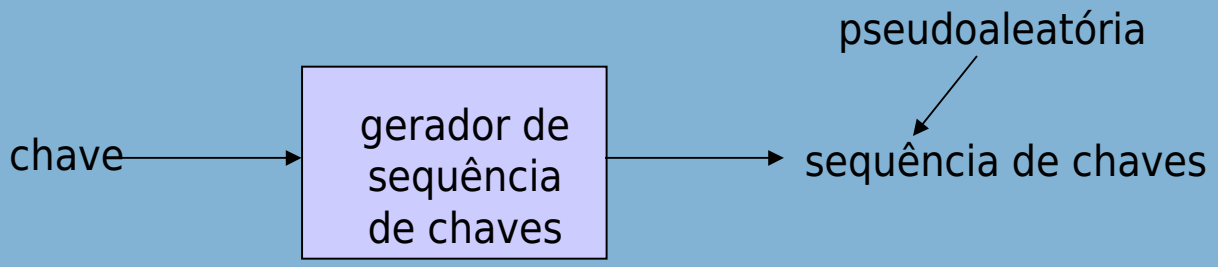
- p. e., segredo é saber padrão de substituição na cifra de substituição monoalfabética

P: Como Bob e Alice combinam um valor de segredo?

Dois tipos de cifras simétricas

- Cifras de fluxo
  - criptografam um bit por vez
- Cifras de bloco
  - Quebram a mensagem de texto aberto em blocos de mesmo tamanho
  - Criptografam cada bloco como uma unidade

Cifras de fluxo



- Combinam cada bit da sequência de chaves com bit de texto aberto para obter bit de texto cifrado
- m(i) = iº bit da mensagem
- ks(i) = iº bit da sequência de chaves
- c(i) = iº bit do texto cifrado
- c(i) = ks(i) ⊕ m(i) (⊕ = OR exclusivo, ou XOR)
- m(i) = ks(i) ⊕ c(i)

Cifra de fluxo RC4

- RC4 é uma cifra de fluxo popular
  - bastante analisada
  - chave pode ter de 1 a 256 bytes
  - usada por WEP para 802.11
  - pode ser usada em SSL

Cifras de bloco

- Mensagem a ser criptografada é processada em blocos de k bits (p. e., blocos de 64 bits).
- Mapeamento 1-para-1 é usado para mapear bloco de k bits de texto aberto para bloco de k bits de texto cifrado

Exemplo com k = 3:

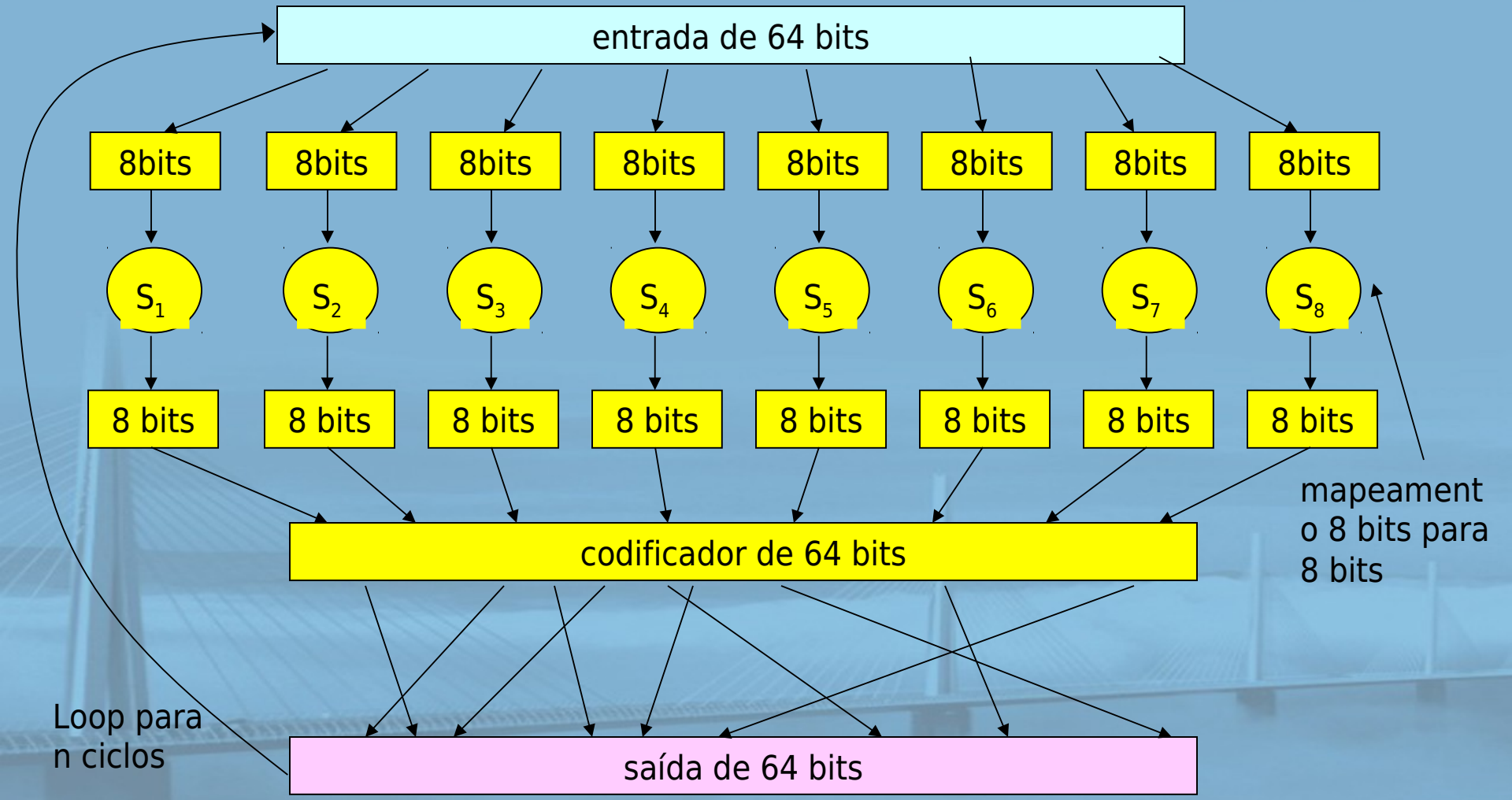
entrada	saída
000	110
001	111
010	101
011	100

entrada	saída
100	011
101	010
110	000
111	001

Qual é o texto cifrado para 010110001111 ?

- Quantos mapeamentos existem para k = 3?
  - Quantas entradas de 3 bits?
  - Quantas permutações das entradas de 3 bits?
  - Resposta: 40.320 ; não muitas!
- Em geral, $2^k!$ mapeamentos; imenso para k = 64
- Problema:
  - Técnica de tabela requer tabela com 2^{64} entradas, cada entrada com 64 bits
- Tabela muito grande: em vez disso, use função que simula tabela permutada aleatoriamente

Função de protótipo



Fonte: Kaufman, 1995

O que acontece no protótipo?

- se um único ciclo, então um bit de entrada afeta no máximo 8 bits de saída.
- no 2º ciclo, os 8 bits afetados são espalhados e inseridos em múltiplas caixas de substituição.
- quantos ciclos?
  - quantas vezes você precisa misturar cartas?
  - torna-se menos eficiente quando *n* aumenta

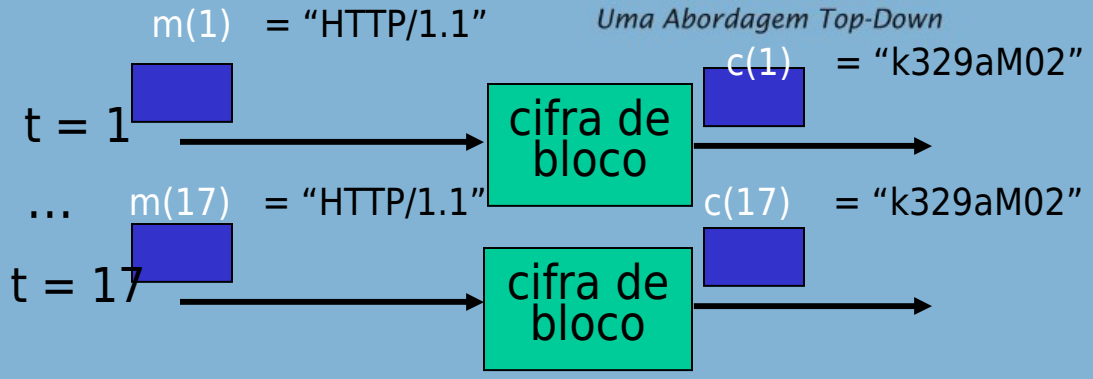
Criptografando uma mensagem grande

- Por que não apenas quebra a mensagem em blocos de 64 bits e criptografar cada bloco separadamente?
  - se mesmo bloco de texto aberto aparecer duas vezes, gerará o mesmo texto cifrado.
- Que tal:
  - gerar número aleatório de 64 bits r(i) para cada bloco de texto aberto m(i)
  - calcular $c(i) = K_s(m(i) \oplus r(i))$
  - transmitir c(i), r(i), i = 1,2,...
  - no destinatário: $m(i) = K_s(c(i)) \oplus r(i)$
  - problema: ineficaz, precisa enviar c(i) e r(i)

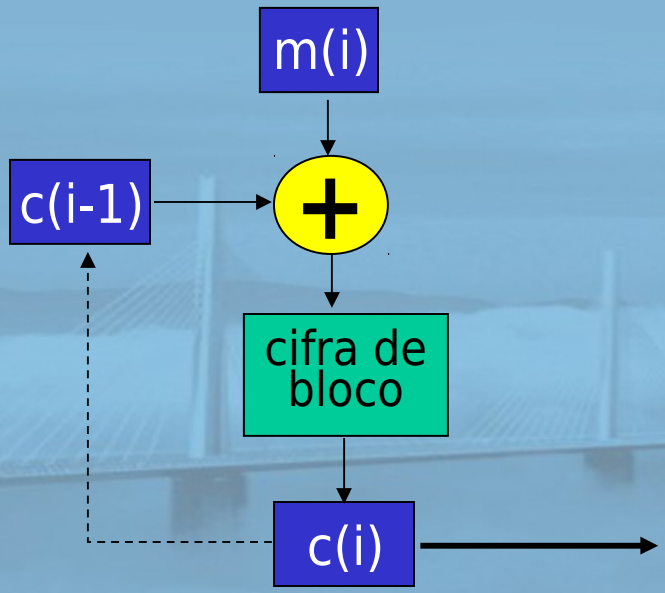
Cipher Block Chaining (CBC)

- CBC gera seus próprios números aleatórios
  - faça a criptografia do bloco atual depender do resultado do bloco anterior
  - $c(i) = K_S(m(i) \oplus c(i-1))$
  - $m(i) = K_S(c(i)) \oplus c(i-1)$
- Como criptografamos o primeiro bloco?
  - vetor de inicialização (IV): bloco aleatório = c(0)
  - IV não precisa ser secreto
- mude IV para cada mensagem (ou sessão)
  - garante que, ainda que a mesma mensagem seja enviada repetidamente, o texto cifrado será completamente diferente a cada vez

- bloco de cifra: se bloco de entrada repetido, produzirá o mesmo texto cifrado:



- *Cipher Block Chaining:* XOR do iº bloco de entrada, m(i), com bloco anterior do texto cifrado, c(i-1)
  - c(0) transmitido ao destinatário abertamente
  - o que acontece no cenário "HTTP/1.1" anterior?



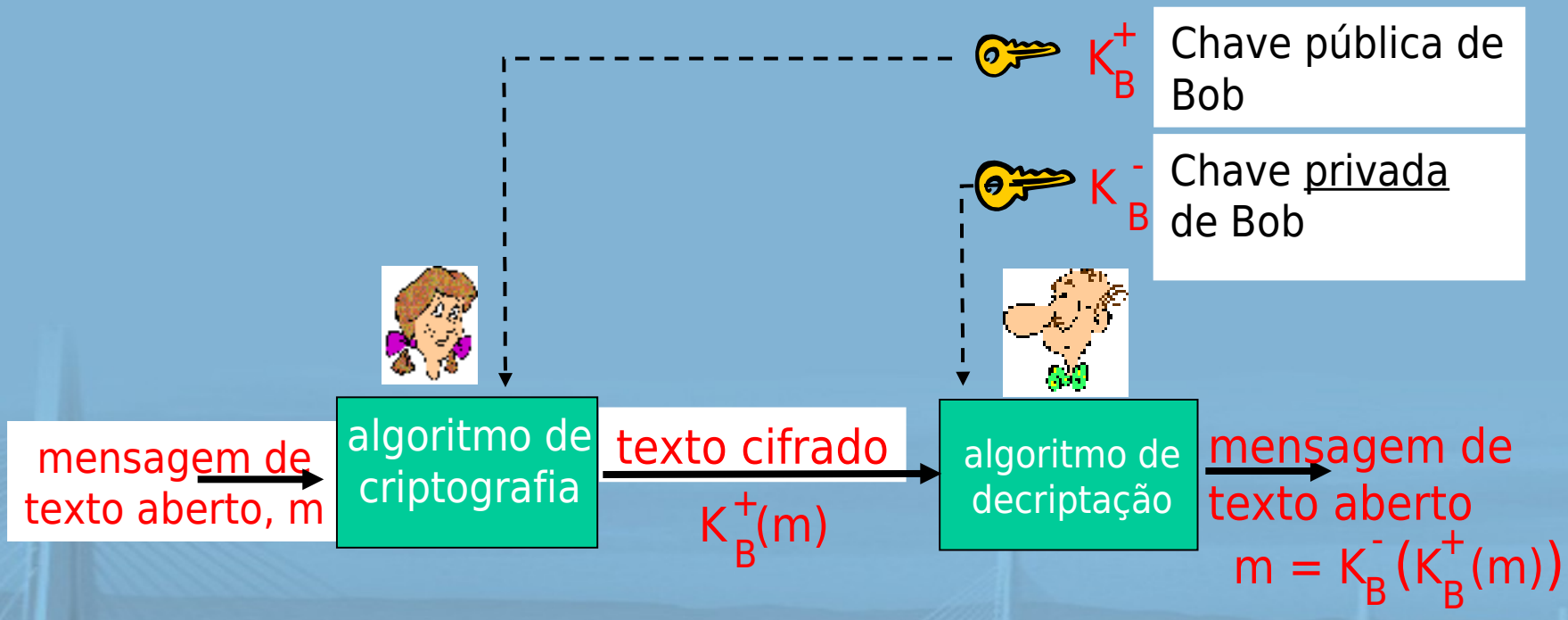
Criptografia de chave pública

chave *simétrica*

- requer que remetente e destinatário conheçam chave secreta
- P: Como combinar sobre a chave em primeiro lugar (principalmente se nunca se "encontraram")?

chave *pública*

- técnica radicalmente diferente [Diffie-Hellman76, RSA78]
- remetente e destinatário *não* compartilham chave secreta
- chave criptográfica *pública* conhecida por *todos*
- chave de decriptação *privada* conhecida apenas pelo receptor



Algoritmo de criptografia de chave pública

Requisitos:

1. precisa de $K_B^+(\bullet)$ e $K_B^-(\bullet)$ tais que

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

2. dada a chave pública K_B^+ , deverá ser impossível calcular chave privada K_B^-

RSA: Algoritmo de Rivest, Shamir, Adelson

Pré-requisito: aritmética modular

- x mod n = resto de x quando divide por n
- Fatos:

  [(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: x = 14, n = 10, d = 2:

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

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

RSA

- Uma mensagem é um padrão de bits.
- Um padrão de bits pode ser representado exclusivamente por um número inteiro.
- Assim, criptografar uma mensagem é equivalente a criptografar um número.

Exemplo

- m = 10010001. Essa mensagem é representada exclusivamente pelo número decimal 145.
- Para criptografar m, criptografamos o número correspondente, que gera um novo número (o texto cifrado).

RSA: Criando par de chave pública/privada

1. Escolha dois números primos grandes *p*, *q*. (p. e., 1024 bits cada)

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

3. Escolha *e* (com $e<n$) que não tenha fatores comuns com z. (*e*, *z* são "relativamente primos").

4. Escolha *d* tal que *ed-1* seja divisível exatamente por *z*. (em outras palavras: *ed* mod *z* = *1*).

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

RSA: criptografia, decriptação

0. Dados (n,e) e (n,d) conforme calculamos

1. Para criptografar a mensagem m $(<n)$, calcule

$$c = m^e \bmod n$$

2. Para decriptar padrão de bits recebido, c, calcule

$$m = c^d \bmod n$$

A mágica acontece!

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

Exemplo de RSA:

Bob escolhe $p = 5$, $q = 7$. Depois, $n = 35$, $z = 24$.
$e = 5$ (assim, e, z relativamente primos).
$d = 29$ (assim, $ed-1$ divisível exatamente por z).

Criptografando mensagens de 8 bits.

criptografia:

padrão de bits	m	m^e	$c = m^e \bmod n$
00001100	12	24832	17

decriptação:

c	c^d	$m = c^d \bmod n$
17	481968572106750915091411825223071697	12

Por que RSA funciona?

- Deve mostrar que $c^d \bmod n = m$ onde $c = m^e \bmod n$
- Fato: para qualquer x e y: $x^y \bmod n = x^{(y \bmod z)} \bmod n$
  - onde n = pq and z = (p-1)(q-1)
- Assim,

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

RSA: outra propriedade importante

A propriedade a seguir será *muito* útil adiante:

$$\underbrace{K_B^-(K_B^+(m))}_{\text{use chave pública primeiro, seguida por chave privada}} = m = \underbrace{K_B^+(K_B^-(m))}_{\text{use chave privada primeiro, seguida por chave pública}}$$

O resultado é o mesmo!

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

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 RSA é seguro?

- Suponha que você conheça a chave pública de Bob (n,e). Qual é a dificuldade de determinar d?
- Basicamente, é preciso encontrar fatores de n sem conhecer os dois fatores p e q.
- Fato: fatorar um número muito grande é difícil.

Gerando chaves RSA

- É preciso achar números primos p e q grandes!

Chaves de sessão

- Exponenciação é computacionalmente intensa
- CBC é pelo menos 100 vezes mais rápido que RSA

Chave de sessão, K_S

- Bob e Alice usam RSA para trocar uma chave simétrica K_S
- Quando ambos tiverem K_S, eles usam a criptografia de chave simétrica

Capítulo 8: Esboço

REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

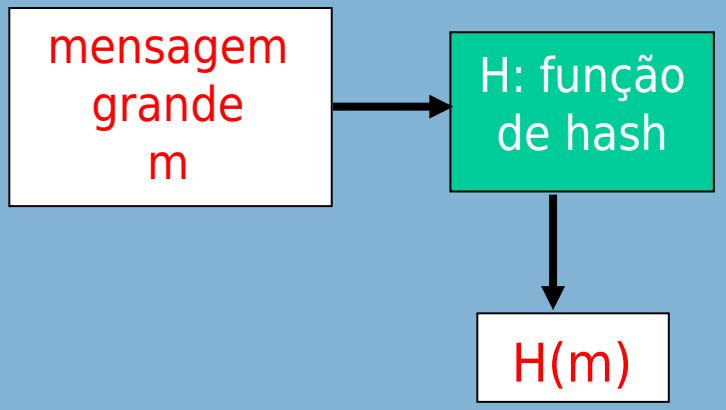
8.1 O que é segurança na rede?
8.2 Princípios de criptografia
8.3 Integridade de mensagem
8.5 Protegendo conexões TCP: SSL
8.8 Segurança operacional: firewalls e IDS

Integridade de mensagem

- permite a comunicação das partes para verificar que as mensagens recebidas são autênticas.
  - conteúdo da mensagem não foi alterado
  - origem da mensagem é quem/o que você pensa ser
  - mensagem não foi reproduzida (replay)
  - sequência de mensagens é mantida
- primeiro, vamos falar sobre resumos de mensagem

Resumos de mensagem

- função H( ) que toma como entrada uma mensagem de tamanho qualquer e gera uma sequência de tamanho fixo: "assinatura da mensagem"
- note que H( ) é uma função muitos-para-um
- H( ) normalmente é chamada "função de hash"



- propriedades desejáveis:
  - fácil de calcular
  - irreversibilidade: não é possível saber m por H(m)
  - resistência a colisão: computacionalmente difícil de produzir m e m' tal que H(m) = H(m')
  - saída aparentemente aleatória

Soma de verificação da Internet: resumo de mensagem fraco

soma de verificação da internet tem propriedades da função de hash:

- produz resumo de tamanho fixo (soma de 16 bits) de entrada
- é muitos-para-um

  - mas, dada mensagem com dado valor de hash, é fácil achar outra mensagem com o mesmo valor de hash.
  - exemplo: soma de verificação simplificada: soma porções de 4 bytes de cada vez:

mensagem	formato ASCII
I O U 1	49 4F 55 31
0 0 . 9	30 30 2E 39
9 B O B	39 42 D2 42
	B2 C1 D2 AC

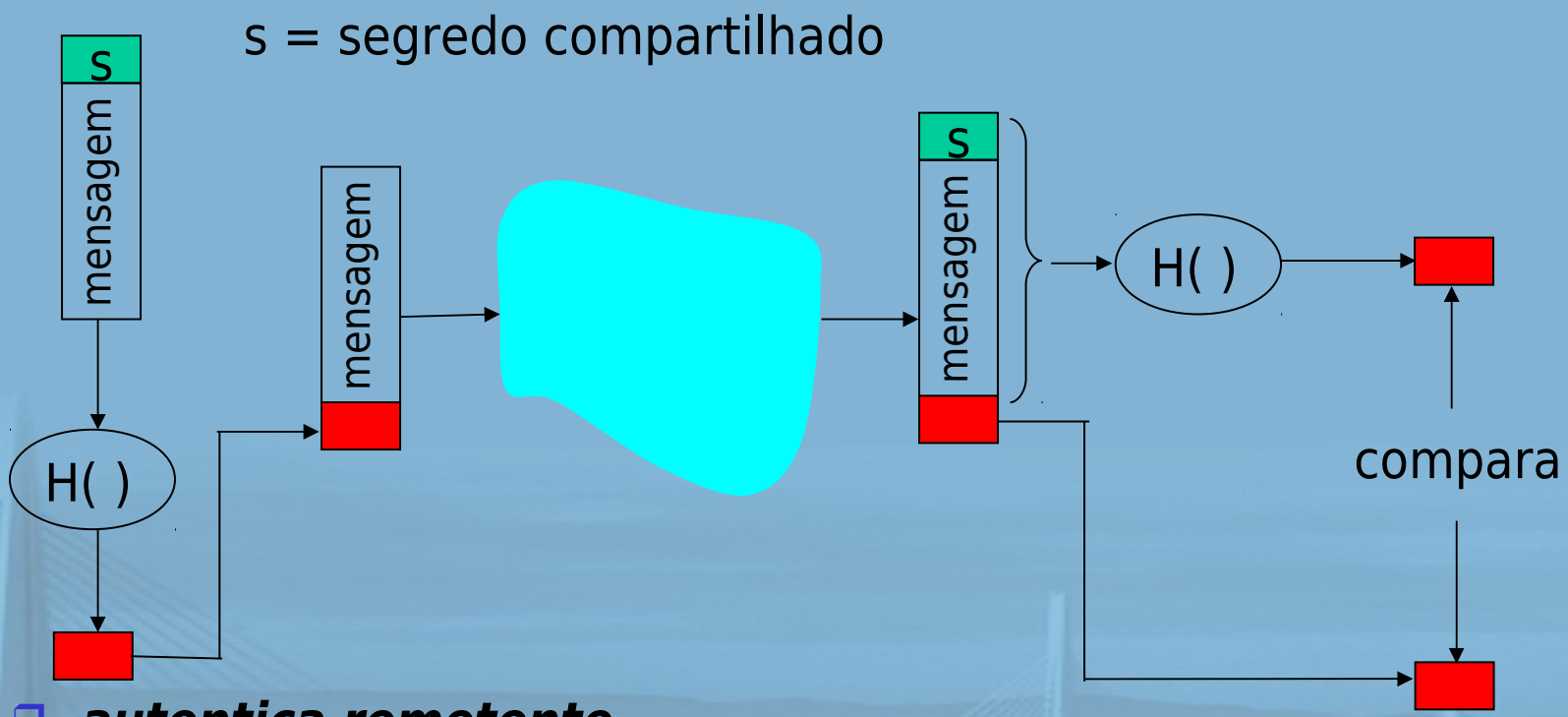
mensagem	formato ASCII
I O U 9	49 4F 55 39
0 0 . 1	30 30 2E 31
9 B O B	39 42 D2 42
	B2 C1 D2 AC

mensagens diferentes mas somas de verificação idênticas!

Algoritmos de função de hash

- função de hash MD5 bastante usada (RFC 1321)
  - calcula resumo de mensagem de 128 bits em processo de 4 etapas.
- SHA-1 também é usado.
  - resumo de mensagem de 160 bit

Message Authentication Code (MAC)



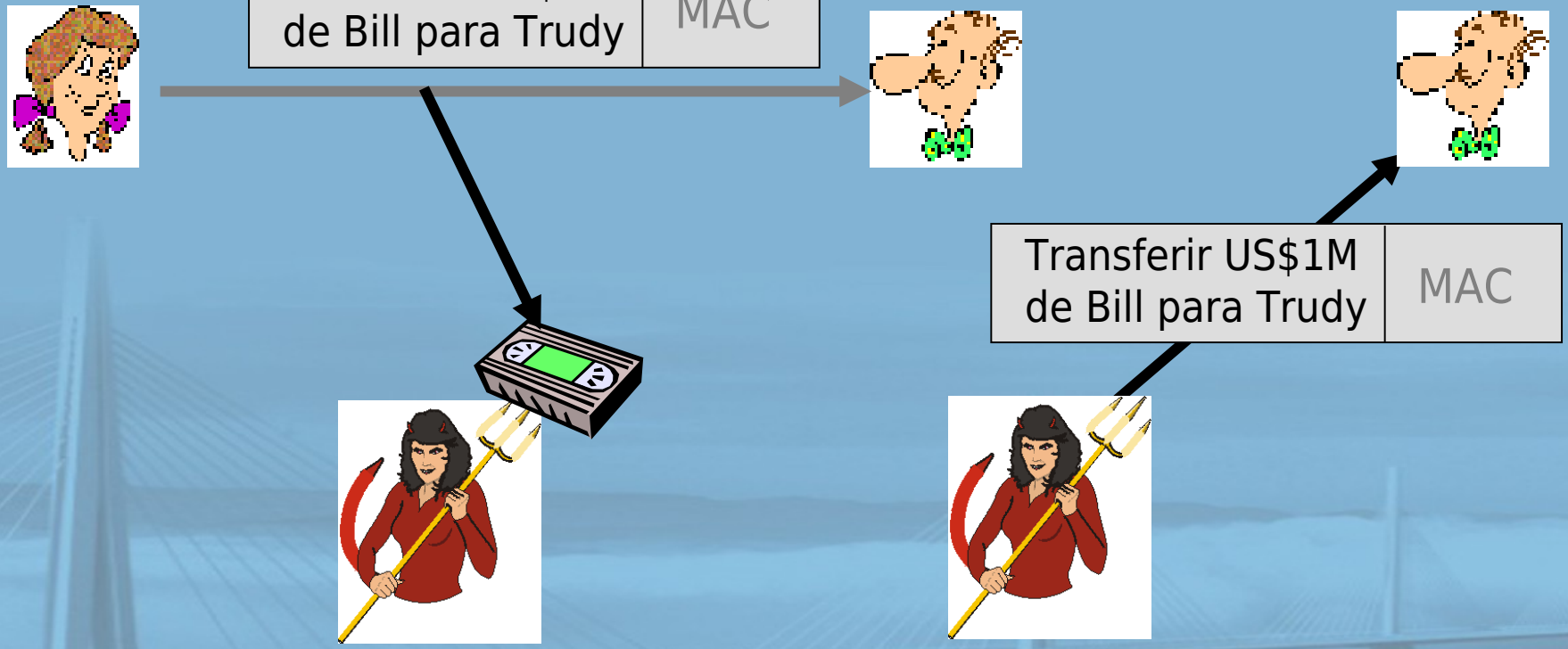
- ***autentica remetente***
- ***verifica integridade da mensagem***
- sem criptografia!
- também chamado “hash chaveado”
- notação: $MD_m = H(s||m)$; envia $m||MD_m$

Autenticação do ponto final

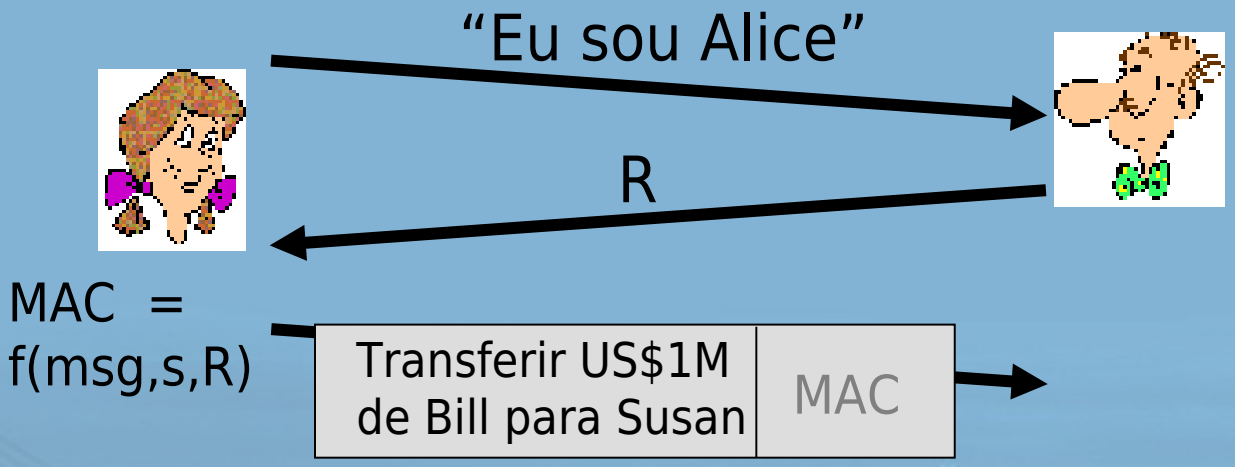
- deseja ter certeza do remetente da mensagem – *autenticação do ponto final*
- supondo que Alice e Bob tenham um segredo compartilhado, MAC oferecerá autenticação do ponto final
  - sabemos que Alice criou a mensagem
  - mas ela a enviou?

Ataque de reprodução

MAC =f(msg,s)



Defendendo contra ataque de reprodução: nonce



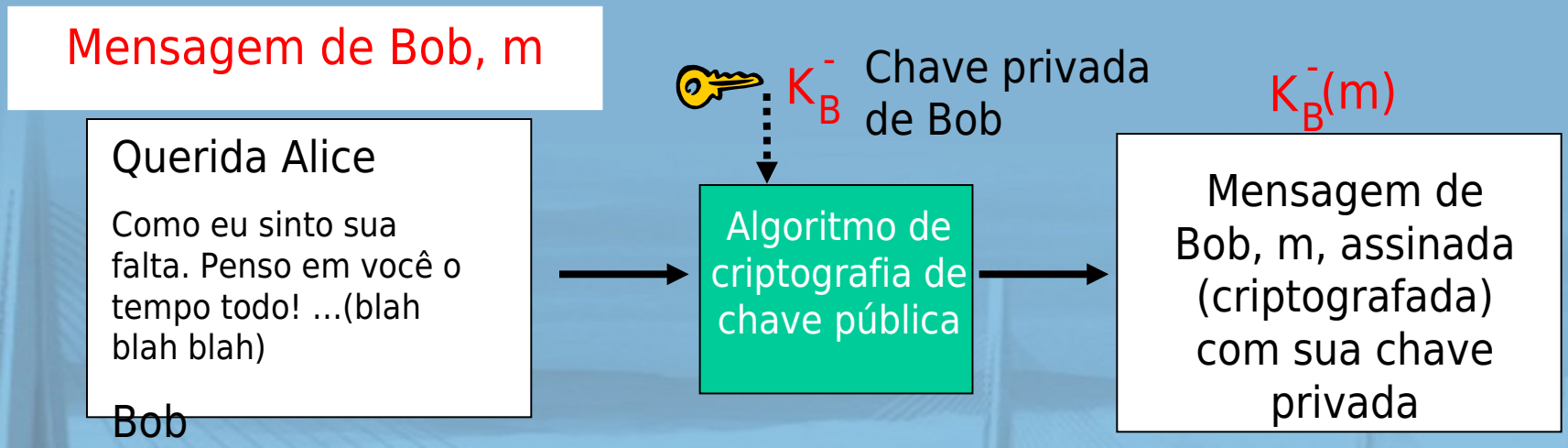
Assinaturas digitais

Técnica criptográfica semelhante a assinaturas escritas a mão.

- remetente (Bob) assina documento digitalmente, estabelecendo que é o dono/criador do documento.
- objetivo semelhante a um MAC, exceto que agora usamos criptografia de chave pública.
- verificável, não falsificável: destinatário (Alice) pode provar a alguém que Bob, e ninguém mais (incluindo Alice), deverá ter assinado o documento.

assinatura digital simples para mensagem m:

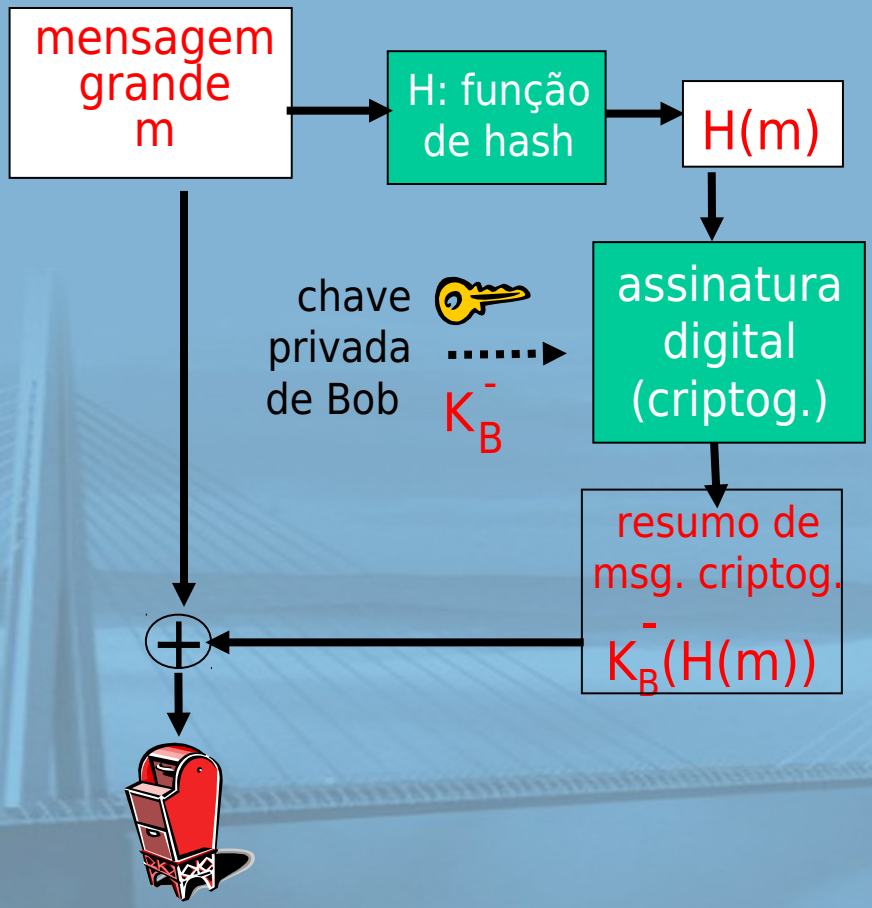
- Bob assina m criptografando com sua chave privada K_B^-, criando mensagem "assinada", $K_B^-(m)$



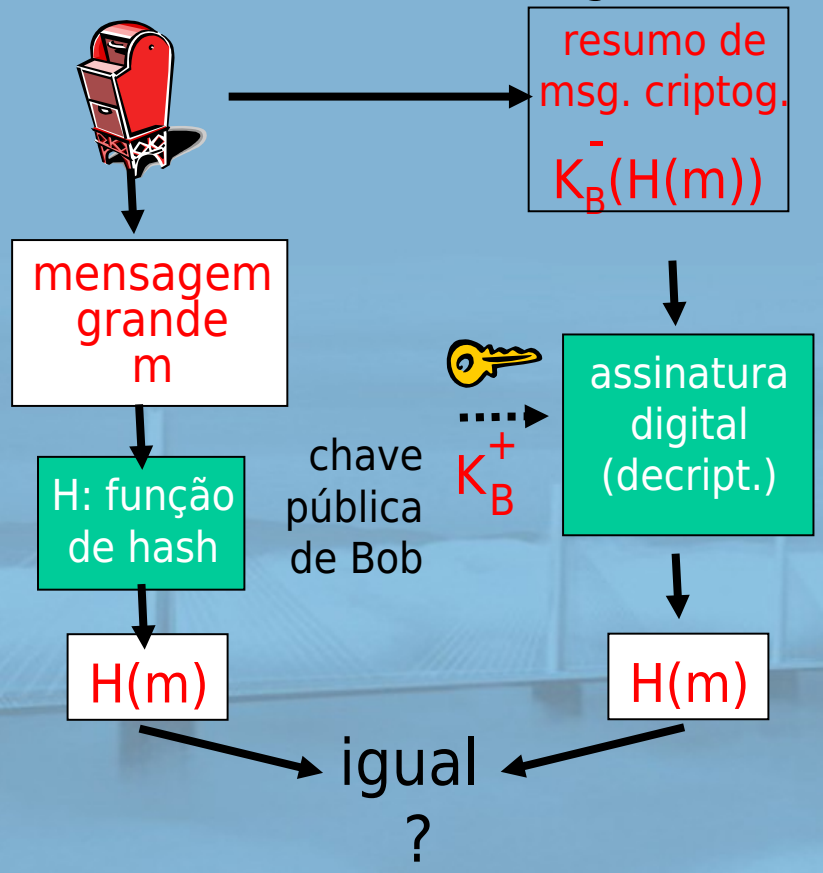
Assinatura digital = resumo de mensagem assinada

REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

Bob envia mensagem assinada em forma digital:



Alice verifica assinatura e integridade da mensagem assinada em forma digital:



Assinaturas digitais (mais)

- Suponha que Alice receba msg m, assinatura digital $K_B^-(m)$
- Alice verifica m assinada por Bob aplicando chave pública de Bob K_B^+ a $K_B^-(m)$, depois verifica $K_B^+(K_B^-(m)) = m$.
- se $K_B^+(K_B^-(m)) = m$, quem assinou m deve ter usado a chave privada de Bob.

Assim, Alice verifica se:

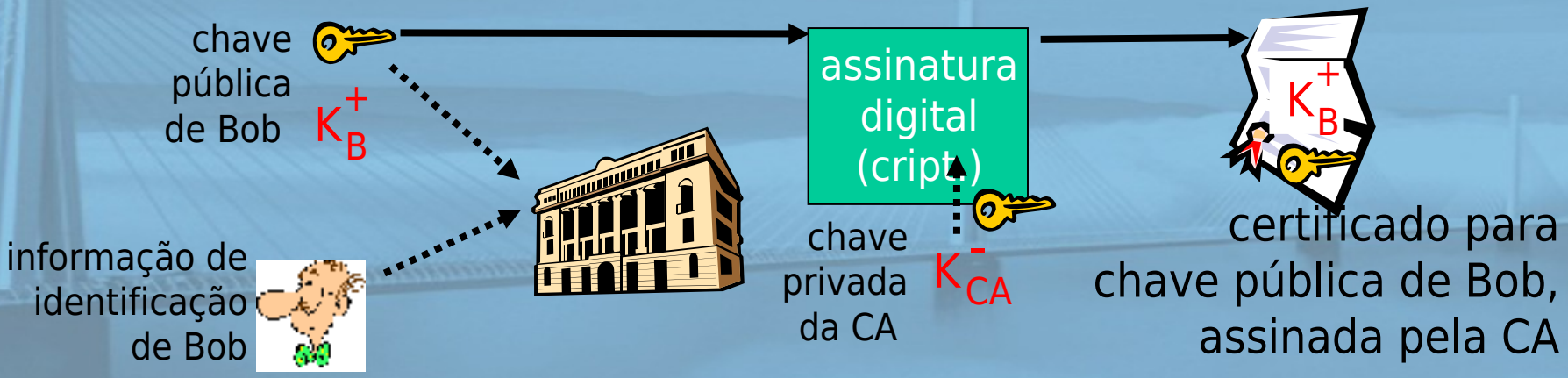
- Bob assinou m.
- Ninguém mais assinou m.
- Bob assinou m e não m'.

Não repudiação:

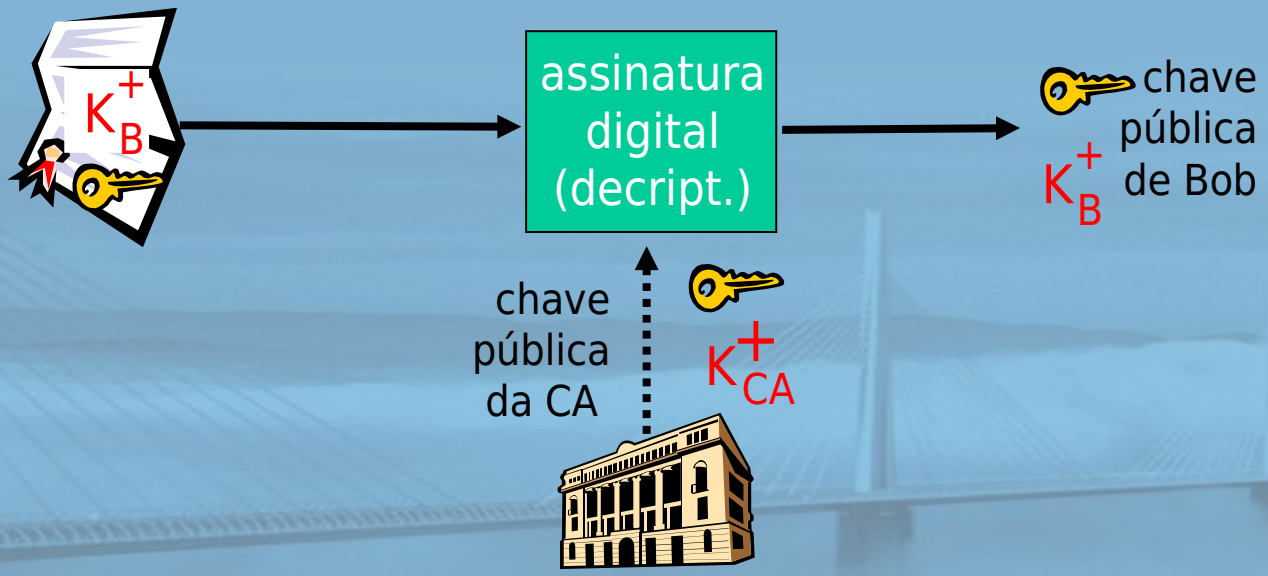
- ✓ Alice pode levar m e assinatura $K_B^-(m)$ ao tribunal e provar que Bob assinou m.

Autoridades de certificação

- autoridade de certificação (CA): vincula chave pública à entidade particular, E.
- E (pessoa, roteador) registra sua chave pública com CA.
  - E fornece "prova de identidade" à CA.
  - CA cria certificado vinculando E à sua chave pública.
  - certificado contendo chave pública de E assinada digitalmente pela CA – CA diz "esta é a chave pública de E"



- quando Alice quer a chave pública de Bob:
  - recebe certificado de Bob (Bob ou outro).
  - aplica chave pública da CA ao certificado de Bob, recebe chave pública de Bob



Capítulo 8: Esboço

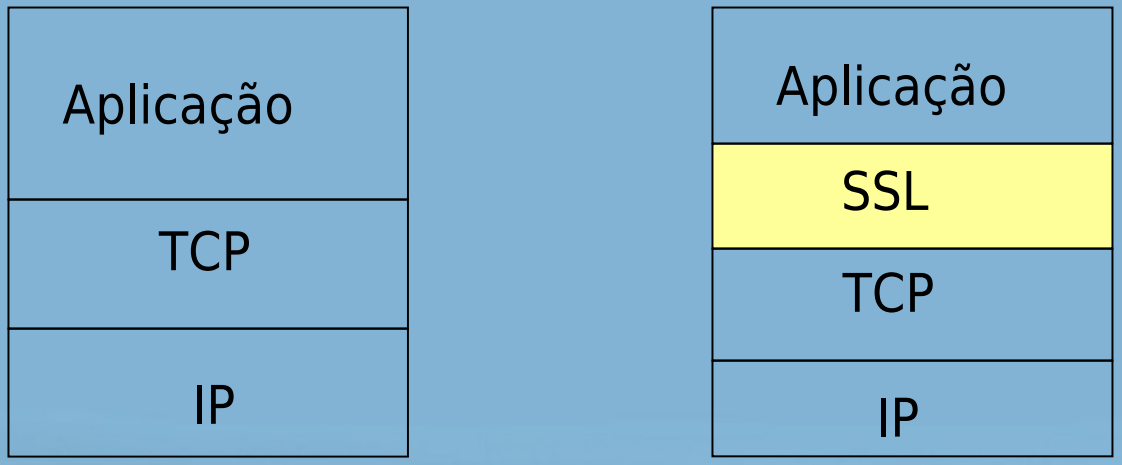
REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

8.1 O que é segurança na rede?
8.2 Princípios de criptografia
8.3 Integridade de mensagem
8.5 Protegendo conexões TCP: SSL
8.8 Segurança operacional: firewalls e IDS

SSL: Secure Sockets Layer

- protocolo de segurança bastante implantado
  - aceito por quase todos os navegadores e servidores Web
  - https
  - dezenas de bilhões de US$ gastos por ano sobre SSL
- originalmente projetado pela Netscape em 1993
- variações:
  - TLS: Transport Layer Security, RFC 2246
- oferece
  - Confidencialidade
  - Integridade
  - Autenticação
- objetivos originais:
  - teve em mente transações de comércio eletrônico na Web
  - criptografia (especialmente números de cartão de crédito)
  - autenticação de servidor Web
  - autenticação de cliente opcional
  - mínimo de incômodo ao fazer negócios com novos comerciantes
- disponível a todas as aplicações TCP
  - Interface Secure Socket

SSL e TCP/IP



aplicação normal

aplicação com SSL

- SSL oferece interface de programação de aplicação (API) às aplicações
- bibliotecas/classes SSL em C e Java prontamente disponíveis

SSL: um canal seguro simples

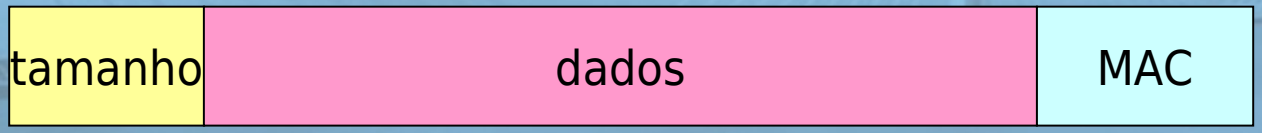
- apresentação: Alice e Bob usam seus certificados e chaves privadas para autenticar um ao outro e trocar segredo compartilhado
- derivação de chave: Alice e Bob usam segredo compartilhado para derivar conjunto de chaves
- transferência de dados: dados a serem transferidos são desmembrados em uma série de registros
- encerramento de conexão: mensagens especiais para encerrar conexão com segurança

Derivação de chave

- considerado ruim usar a mesma chave para mais de uma operação criptográfica
  - use chaves diferentes para código de autenticação de mensagem (MAC) e criptografia
- quatro chaves:
  - K_c = chave de criptografia para dados enviados do cliente ao servidor
  - M_c = chave MAC para dados enviados do cliente ao servidor
  - K_s = chave de criptografia para dados enviados do servidor ao cliente
  - M_s = chave MAC para dados enviados do servidor ao cliente

Registros de dados

- Por que não criptografar dados em fluxo constante enquanto o escrevemos no TCP?
  - Onde colocaríamos o MAC? Se no final, nenhuma integridade de mensagem até todos os dados processados.
  - Por exemplo, com mensagens instantâneas, como podemos fazer verificação de integridade por todos os bytes enviados antes da exibição?
- Em vez disso, quebre fluxo em série de registros
  - cada registro transporta um mac
  - receptor pode atuar em cada registro quando ele chega
- Problema: no registro, receptor precisa distinguir MAC dos dados
  - quer usar registros de tamanho variável



Números de sequência

- invasor pode capturar e reproduzir registro ou reordenar registros
- solução: colocar número de sequência em MAC:
  - MAC = MAC(M_x, sequência||dados)
- invasor ainda poderia reproduzir todos os registros
  - use nonce aleatório

Cifras simétricas mais comuns em SSL

- DES – Data Encryption Standard: bloco
- 3DES – Força tripla: bloco
- RC2 – Rivest Cipher 2: bloco
- RC4 – Rivest Cipher 4: fluxo

Criptografia de chave pública

- RSA

SSL: Apresentação

Propósito

1. Autenticação do servidor
2. Negociação: concordar sobre algoritmos de criptografia
3. Estabelecer chaves
4. Autenticação do cliente (opcional)

Capítulo 8: Esboço

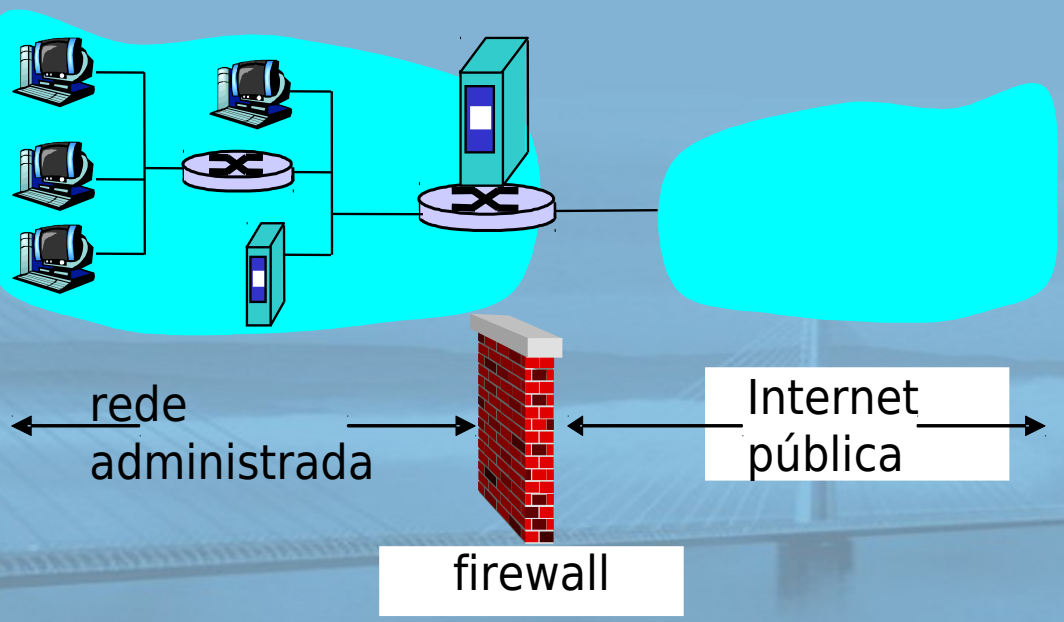
REDES DE COMPUTADORES E A INTERNET 5ª edição
Uma Abordagem Top-Down

8.1 O que é segurança na rede?
8.2 Princípios de criptografia
8.3 Integridade de mensagem
8.5 Protegendo conexões TCP: SSL
8.8 Segurança operacional: firewalls e IDS

Firewalls

firewall

isola rede interna da organização da Internet maior, permitindo que alguns pacotes passem e bloqueando outros.



Firewalls: Por que

impedir ataques de negação de serviço:

- inundação de SYN: atacante estabelece muitas conexões TCP falsas, sem recursos deixados para conexões “reais”

impedir modificação/acesso ilegal de dados internos

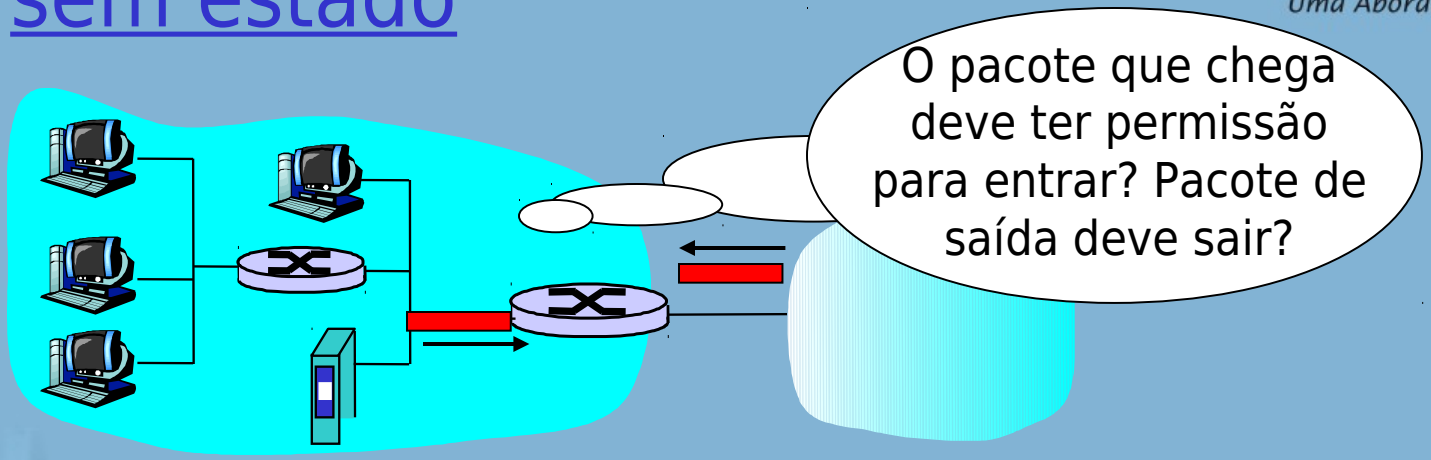
- p. e., atacante substitui página inicial da companhia por algo diferente

permite apenas acesso autorizado à rede interna (conjunto de usuários/hospedeiros autenticados)

três tipos de firewalls:

- filtros de pacotes sem estado
- filtros de pacotes com estado
- gateways de aplicação

Filtragem de pacotes sem estado



- rede interna conectada à Internet via firewall do roteador
- roteador filtra pacote-por-pacote, decisão de repassar/descartar pacote com base em:
  - endereço IP de origem, endereço IP de destino
  - números de porta de origem e destino do TCP/UDP
  - tipo de mensagem ICMP
  - bits SYN e ACK do TCP

Filtragem de pacotes sem estado: exemplo

- exemplo 1: bloco entrando e saindo datagramas com campo de protocolo IP = 17 e com porta de origem ou destino = 23
  - todo UDP entrando e saindo fluxos e conexões telnet são bloqueados
- exemplo 2: bloco entrando segmentos TCP com ACK = 0
  - impede que clientes externos façam conexões TCP com clientes internos, mas permite que clientes internos se conectem ao exterior

Filtragem de pacotes sem estado: mais exemplos

Política	configuração de firewall
sem acesso externo à Web	descarta todos os pacotes que saem para qualquer endereço IP, porta 80
sem conexões TCP entrando, exceto aquelas apenas para o servidor Web público da instituição	descarta todos pacotes TCP SYN que chegam a qualquer IP, exceto 130.207.244.203, porta 80
impedir que Web-radios devorem a largura de banda disponível	descarta todos os pacotes UDP que chegam - exceto DNS e broadcasts do roteador
impedir que sua rede seja usada para um ataque DoS smurf	descarta todos os pacotes ICMP indo para um endereço de “broadcast” (p. e., 130.207.255.255)
impedir que sua rede interaja com o programa Traceroute	descarta todo tráfego expirado ICMP TTL de saída

Listas de controle de acesso

- *ACL:* tabela de regras, aplicadas de cima para baixo aos pacotes que chegam: pares (ação, condição)

Ação	Endereço de origem	Endereço de destino	Protocolo	Porta de origem	Porta de destino	Flag bit
Permitir	222.22/16	Fora de 222.22/16	TCP	>1023	80	Qualquer um
Permitir	Fora de 222.22/16	222.22/16	TCP	80	>1023	ACK
Permitir	222.22/16	Fora de 222.22/16	UDP	>1023	53	_
Permitir	Fora de 222.22/16	222.22/16	UDP	53	>1023	_
Negar	Todos	Todos	Todos	Todos	Todos	Todos

Filtragem de pacotes com estado

- filtro de pacotes sem estado
  - admite pacotes que “não fazem sentido”, p. e., porta destino = 80, bit ACK marcado, mesmo sem conexão TCP estabelecida:

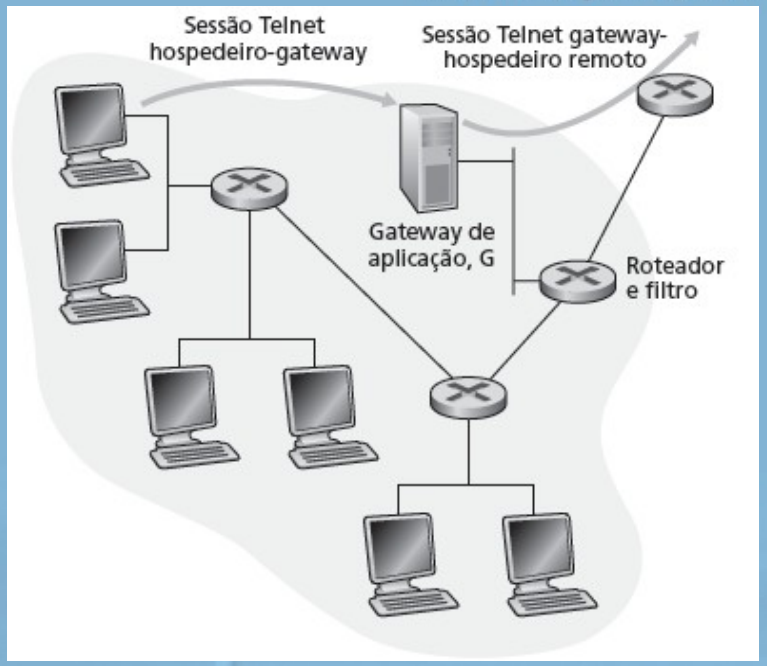
ação	endereço de origem	endereço de destino	protocolo	porta de origem	porta de destino	bit de flag
permitir	fora de 222.22/16	222.22/16	TCP	80	> 1023	ACK

- *filtro de pacotes com estado:* rastreia status de cada conexão TCP
  - rastrear configuração de conexão (SYN), encerramento (FIN): pode determinar se pacotes de entrada e saída “fazem sentido”
  - timeout de conexões inativas no firewall: não admite mais pacotes

- ACL aumentada para indicar necessidade de verificar tabela de estado da conexão antes de admitir pacote

Ação	Endereço de origem	Endereço de destino	Protocolo	Porta de origem	Porta de destino	Flag bit	Conexão de checagem
Permitir	222.22/16	Fora de 222.22/16	TCP	>1023	80	Qualquer um	
Permitir	Fora de 222.22/16	222.22/16	TCP	80	>1023	ACK	X
Permitir	222.22/16	Fora de 222.22/16	UDP	>1023	53	_	
Permitir	Fora de 222.22/16	222.22/16	UDP	53	>1023	_	X
Negar	Todos	Todos	Todos	Todos	Todos	Todos	

Gateways de aplicação



- filtra pacotes nos dados da aplicação, além de campos IP/TCP/UDP.
- exemplo: permitir seleção de usuários internos ao telnet externo.

1. requer que todos os usuários telnet passem pelo gateway.
2. para usuários autorizados, gateway estabelece conexão telnet ao hospedeiro de destino. Gateway repassa dados entre 2 conexões
3. filtro do roteador bloqueia todas as conexões telnet não originando do gateway.

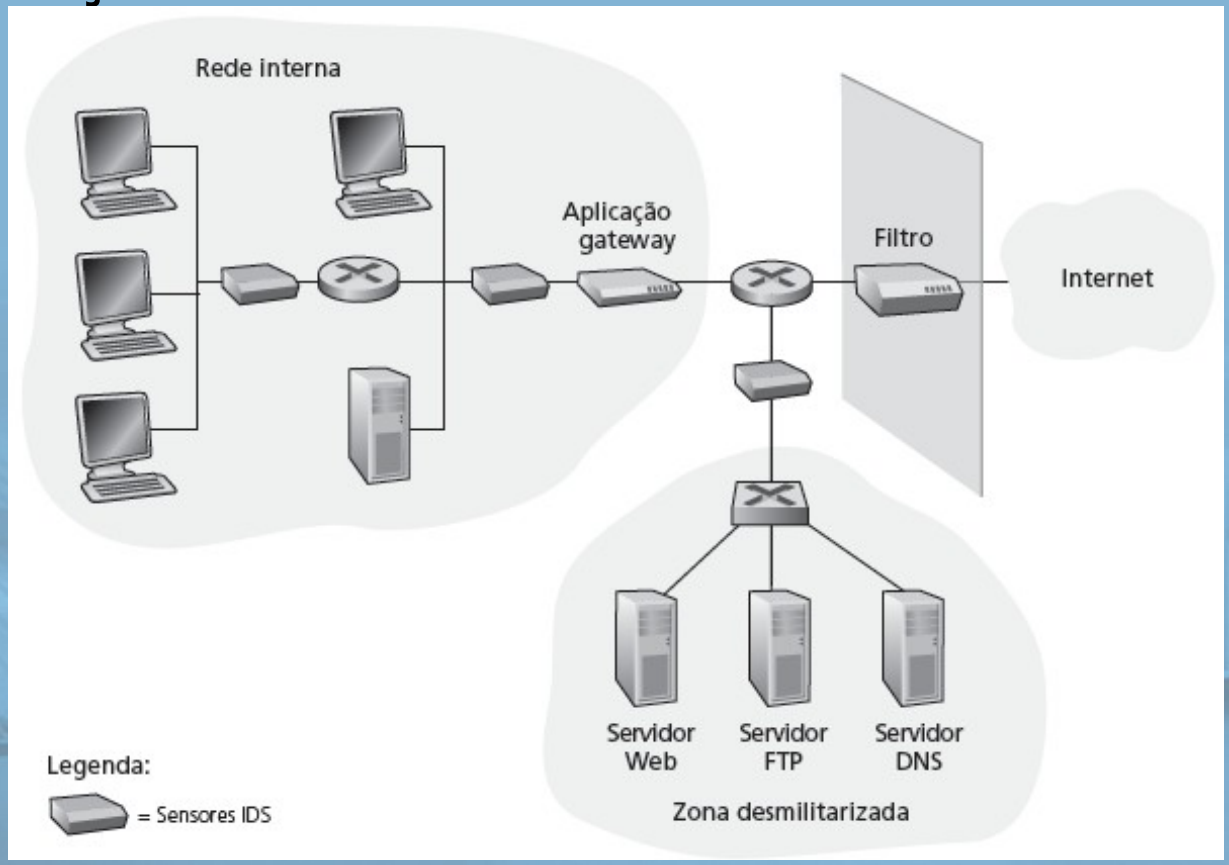
Limitações de firewalls e gateways

- falsificação de IP: roteador não sabe se os dados "realmente" vêm de fonte alegada
- se múltiplas aplicações precisam de tratamento especial, cada uma tem gateway próprio.
- software cliente deve saber como contatar gateway.
  - p. e., deve definir endereço IP do proxy no servidor Web
- filtros normalmente usam toda ou nenhuma política para UDP.
- dilema: grau de comunicação com mundo exterior, nível de segurança
- muitos sites altamente protegidos ainda sofrem de ataques.

Sistemas de detecção de invasão

- filtragem de pacotes:
  - opera apenas sobre cabeçalhos TCP/IP
  - sem verificação de correlação entre sessões
- *IDS: Intrusion Detection System*
  - *profunda inspeção de pacotes:* examina conteúdo do pacote (p. e., verifica strings de caracteres no pacote contra banco de dados de vírus conhecidos e sequências de ataque)
  - examine correlação entre múltiplos pacotes
    - escaneamento de portas
    - mapeamento de rede
    - ataque de DoS

- múltiplos IDSs: diferentes tipos de verificação em diferentes locais



Segurança de rede (resumo)

técnicas básicas.....
- criptografia (simétrica e pública)
- integridade da mensagem
- autenticação do ponto final

.... usado em muitos cenários de segurança diferentes
- e-mail seguro
- transporte seguro (SSL)
- IPsec
- 802.11

Segurança Operacional: firewalls e IDS