

Capítulo 2: Camada de aplicação

- ❑ 2.1 Princípios de aplicações de rede
- ❑ 2.2 A Web e o HTTP
- ❑ 2.3 FTP
- ❑ 2.4 Correio eletrônico
 - ❖ SMTP, POP3, IMAP
- ❑ 2.5 DNS
- ❑ 2.6 Aplicações P2P
- ❑ 2.7 Programação de sockets com UDP
- ❑ 2.8 Programação de sockets com TCP

Web e HTTP

primeiro, algum jargão

- ❑ **página Web** consiste em **objetos**
- ❑ objeto pode ser arquivo HTML, imagem JPEG, applet Java, arquivo de áudio,...
- ❑ página Web consiste em **arquivo HTML básico** que inclui vários objetos referenciados
- ❑ cada objeto é endereçável por um **URL**
- ❑ exemplo de URL:

`www.someschool.edu/someDept/pic.gif`

nome do hospedeiro

nome do caminho

Visão geral do HTTP

HTTP: HyperText Transfer Protocol

- protocolo da camada de aplicação da Web
- modelo cliente/servidor
 - ❖ *cliente*: navegador que requisita, recebe, "exibe" objetos Web
 - ❖ *servidor*: servidor Web envia objetos em resposta a requisições



usa TCP:

- ❑ cliente inicia conexão TCP (cria socket) com servidor, porta 80
- ❑ servidor aceita conexão TCP do cliente
- ❑ mensagens HTTP (do protocolo da camada de aplicação) trocadas entre navegador (cliente HTTP) e servidor Web (servidor HTTP)
- ❑ conexão TCP fechada

HTTP é "sem estado"

- ❑ servidor não guarda informações sobre requisições passadas do cliente

aparte

Protocolos que mantêm "estado" são complexos!

- ❑ história passada (estado) deve ser mantida
- ❑ se servidor/cliente falhar, suas visões do "estado" podem ser incoerentes, devem ser reconciliadas

Conexões HTTP

HTTP não persistente

- ❑ no máximo um objeto é enviado por uma conexão TCP.

HTTP persistente

- ❑ múltiplos objetos podem ser enviados por uma única conexão TCP entre cliente e servidor.

HTTP não persistente

Suponha que o usuário digite o URL

`www.someSchool.edu/someDepartment/home.index`

(contém texto,
referências a 10
imagens JPEG)

1a. Cliente HTTP inicia conexão TCP com servidor HTTP (processo) em `www.someSchool.edu` na porta 80.

1b. Servidor HTTP no hospedeiro `www.someSchool.edu` esperando conexão TCP na porta 80. "aceita" conexão, notificando cliente

2. Cliente HTTP envia *mensagem de requisição* HTTP (contendo URL) pelo socket de conexão TCP. Mensagem indica que cliente deseja o objeto `someDepartment/home.index`.

3. Servidor HTTP recebe mensagem de requisição, forma *mensagem de resposta* contendo objeto requisitado e envia mensagem para seu socket

tempo

4. Servidor HTTP fecha conexão TCP.

5. Cliente HTTP recebe mensagem de resposta contendo arquivo html, exibe html. Analisando arquivo html, acha 10 objetos JPEG referenciados.

6. Etapas 1-5 repetidas para cada um dos 10 objetos JPEG.

tempo



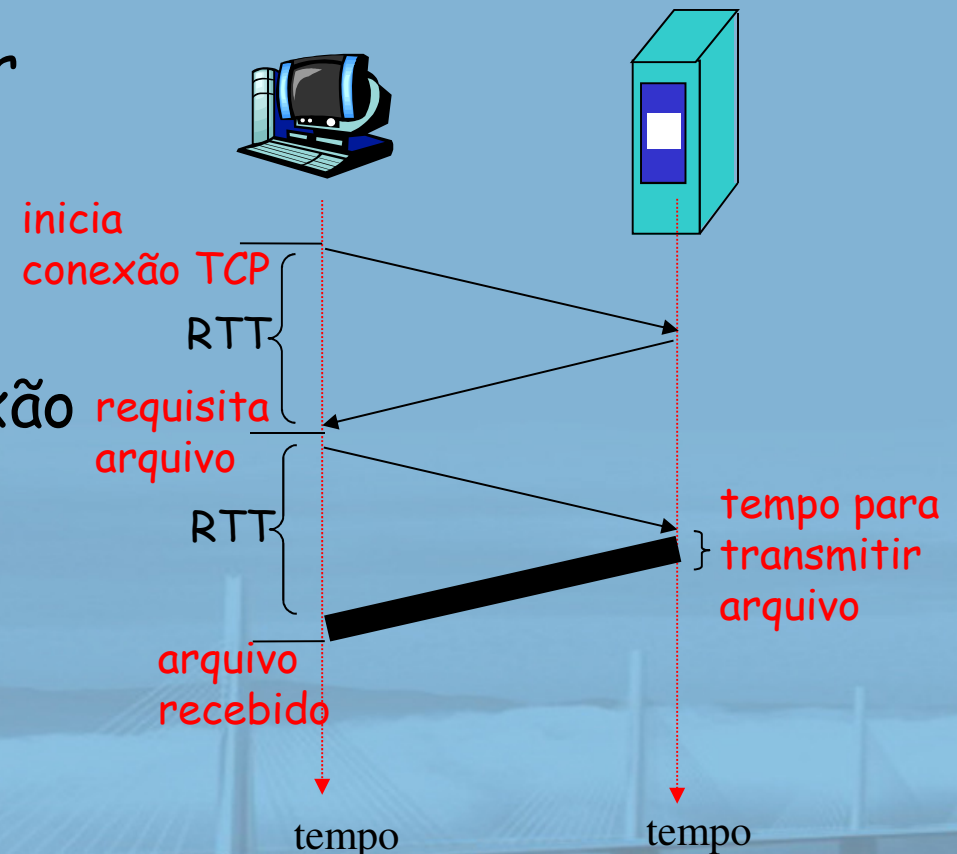
HTTP não persistente: tempo de resposta

definição de RTT: tempo para um pequeno pacote trafegar do cliente ao servidor e retornar.

tempo de resposta:

- ❑ um RTT para iniciar a conexão TCP
- ❑ um RTT para a requisição HTTP e primeiros bytes da resposta HTTP retornarem
- ❑ tempo de transmissão de arquivo

total = $2RTT + \text{tempo de transmissão}$



HTTP persistente

problemas do HTTP não persistente:

- ❑ requer 2 RTTs por objeto
- ❑ overhead do SO para *cada* conexão TCP
- ❑ navegadores geralmente abrem conexões TCP paralelas para buscar objetos referenciados

HTTP persistente:

- ❑ servidor deixa a conexão aberta depois de enviar a resposta
- ❑ mensagens HTTP seguintes entre cliente/servidor enviadas pela conexão aberta
- ❑ cliente envia requisições assim que encontra um objeto referenciado
- ❑ no mínimo um RTT para todos os objetos referenciados

Mensagem de requisição HTTP

- dois tipos de mensagens HTTP: *requisição, resposta*
- *mensagem de requisição HTTP:*
 - ❖ ASCII (formato de texto legível)

linha de requisição
(comandos GET,
POST, HEAD)

linhas de
cabeçalho

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language: fr
```

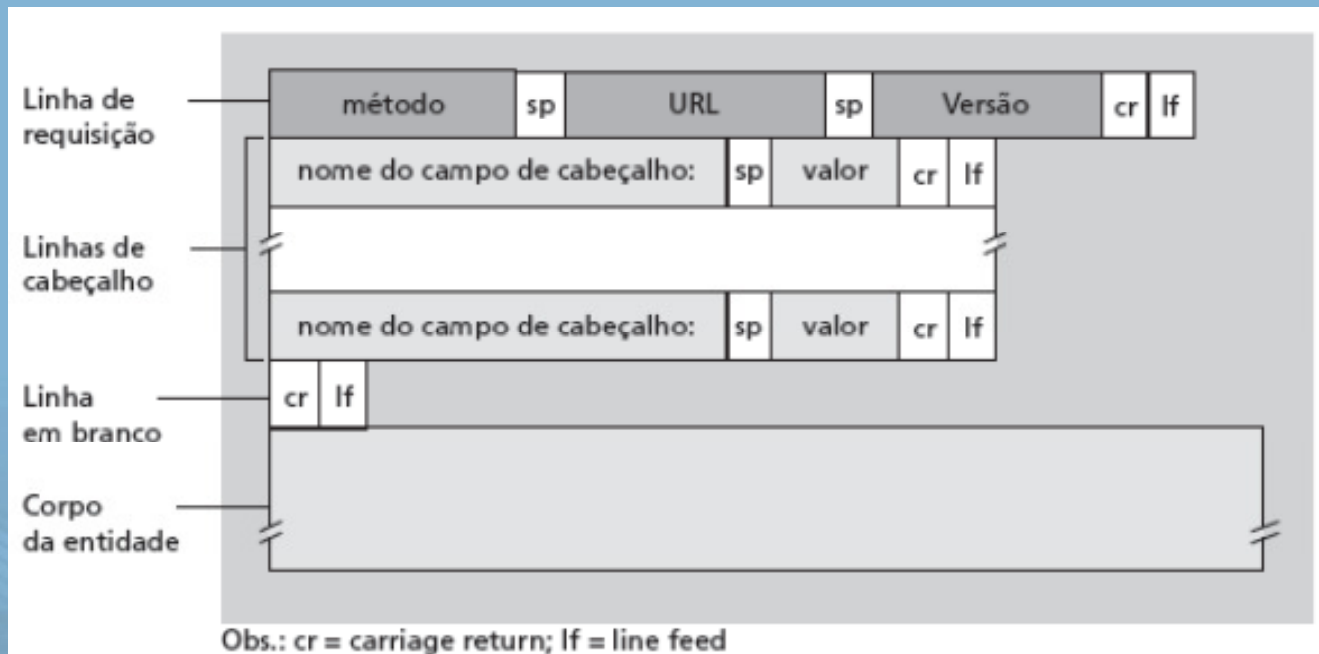
carriage return,
line feed
indica final
da mensagem

(carriage return, line feed extras)

Mensagem de requisição HTTP: formato geral

REDES DE
COMPUTADORES
E A INTERNET 5ª edição

Uma Abordagem Top-Down



Upload da entrada do formulário

método POST:

- ☐ página Web geralmente inclui entrada do formulário
- ☐ entrada é enviada ao servidor no corpo da entidade

método do URL:

- ☐ usa o método GET
- ☐ entrada é enviada no campo de URL da linha de requisição:

`www.umsite.com/buscaanimal?macacos&banana`

Tipos de método

HTTP/1.0

- ❑ GET
- ❑ POST
- ❑ HEAD
 - ❖ pede ao servidor para deixar objeto requisitado fora da resposta

HTTP/1.1

- ❑ GET, POST, HEAD
- ❑ PUT
 - ❖ envia arquivo no corpo da entidade ao caminho especificado no campo de URL
- ❑ DELETE
 - ❖ exclui arquivo especificado no campo de URL

Mensagem de resposta HTTP

linha de status
(protocolo
código de estado
frase de estado)

linhas de
cabeçalho

dados, p. e.,
arquivo HTML
requisitado

HTTP/1.1 200 OK

Connection close

Date: Thu, 06 Aug 1998 12:00:15 GMT

Server: Apache/1.3.0 (Unix)

Last-Modified: Mon, 22 Jun 1998

Content-Length: 6821

Content-Type: text/html

dados dados dados dados dados ...

Códigos de estado da resposta HTTP

primeira linha da mensagem de resposta servidor->cliente
alguns exemplos de código:

200 OK

- ❖ requisição bem-sucedida, objeto requisitado mais adiante

301 Moved Permanently

- ❖ objeto requisitado movido, novo local especificado mais adiante na mensagem (Location:)

400 Bad Request

- ❖ mensagem de requisição não entendida pelo servidor

404 Not Found

- ❖ documento requisitado não localizado neste servidor

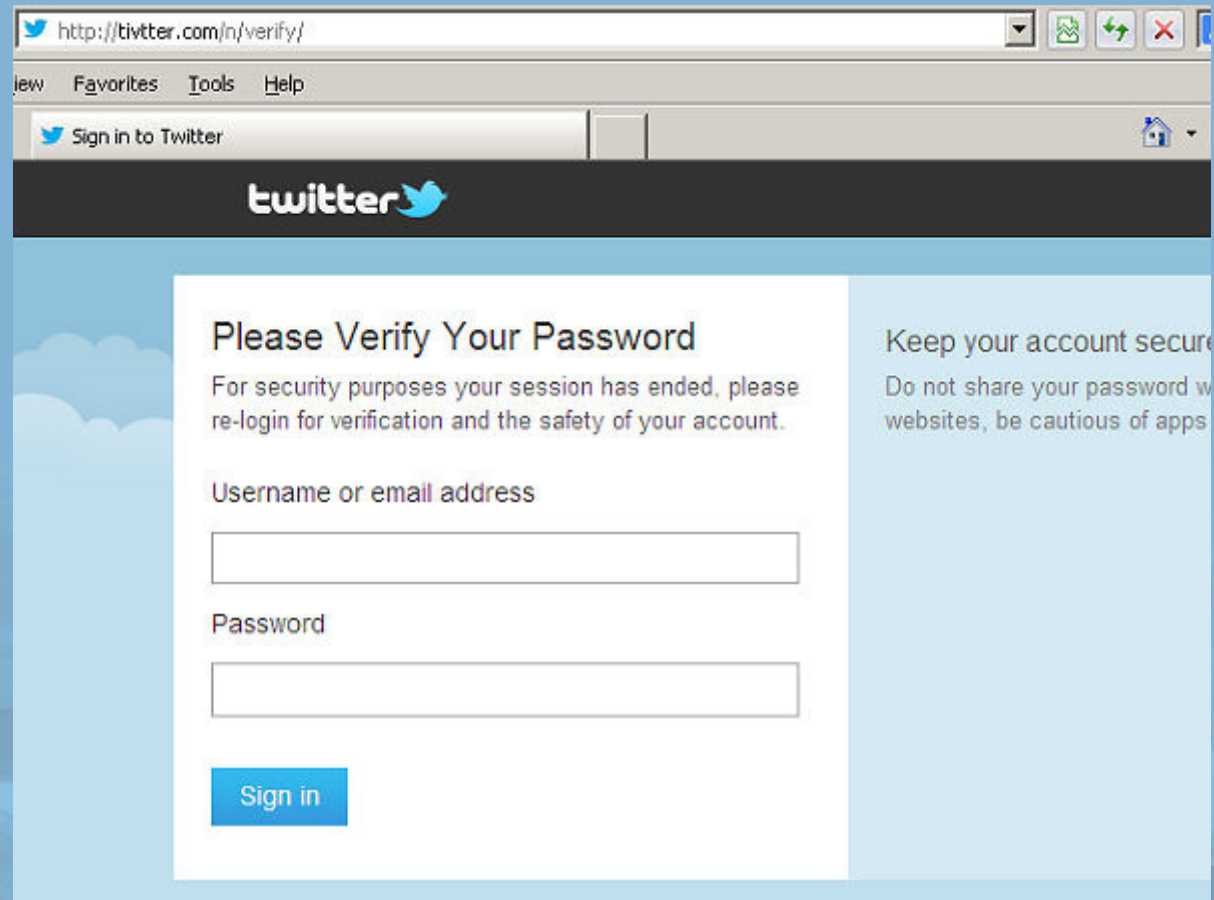
505 HTTP Version Not Supported

Segurança em foco: Phishing

REDES DE
COMPUTADORES
E INTERNET 5ª edição

Uma Abordagem Top-Down

- ❑ O golpe consiste no recebimento de links arbitrários por meio de Mensagens Direta.
- ❑ Os endereços são encurtados, o que dificulta identificar rapidamente uma URL falsa.
- ❑ Depois de clicar no link, o usuário é redirecionado para uma página falsa do Twitter.



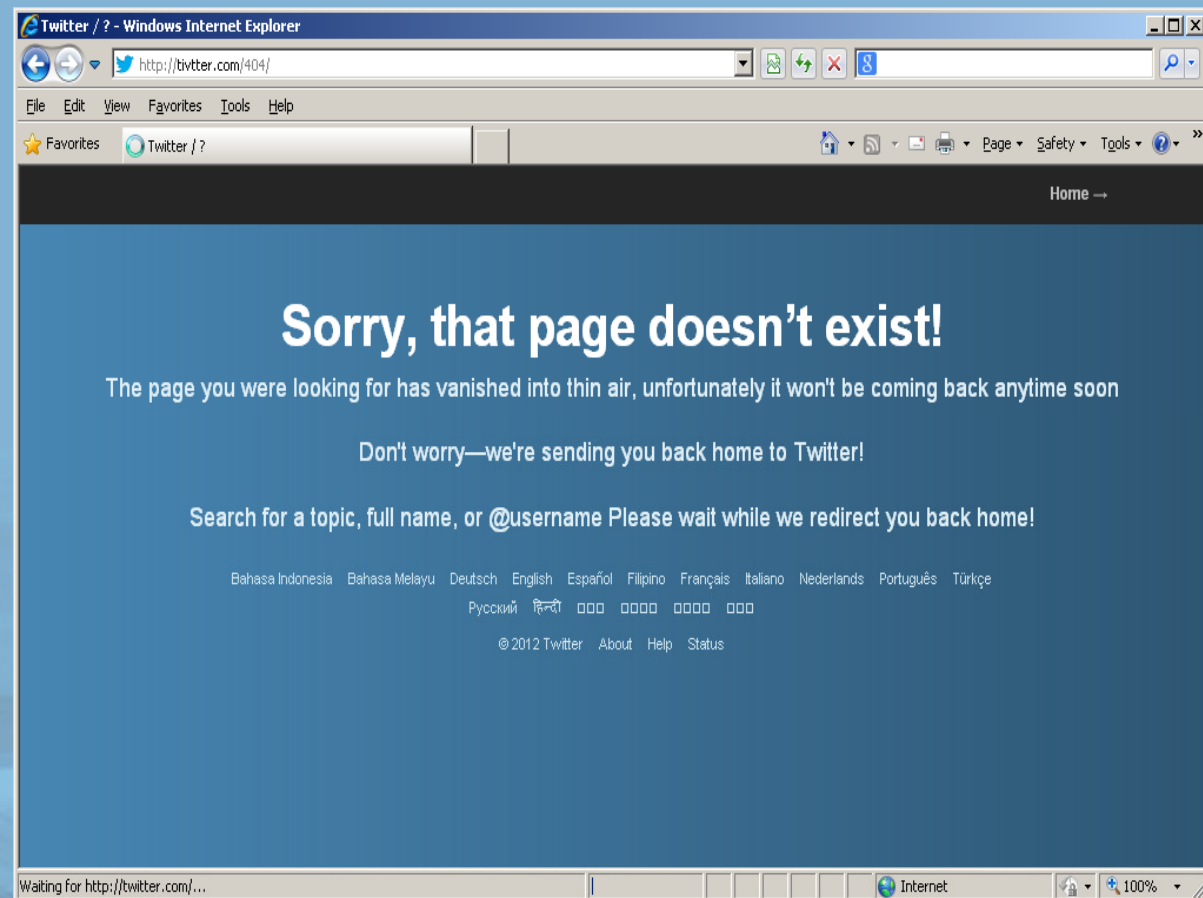
*Página do Twitter forjada, utilizada no ataque phishing
(Imagem: Websense)*

Segurança em foco: Phishing

REDES DE
COMPUTADORES
E INTERNET 5ª edição

Uma Abordagem Top-Down

- ❑ Se a vítima não desconfiar, irá pensar apenas que a sua sessão na página da rede social expirou e irá colocar novamente os dados de acesso para poder visualizar a página solicitada no início de tudo.
- ❑ Antes de redirecionar o usuário ao site legítimo, aparecerá uma mensagem de "página não encontrada", ou o conhecido "Erro 404". Essa é a confirmação de que os dados do usuário já foram roubados.



Disponível em:

<http://idgnow.uol.com.br/internet/2013/02/26/twitter-sofre-onda-de-ataques-phishing-por-meio-de-mensagem-direta/>

Estado usuário-servidor: cookies

Muitos sites importantes
usam cookies

Quatro componentes:

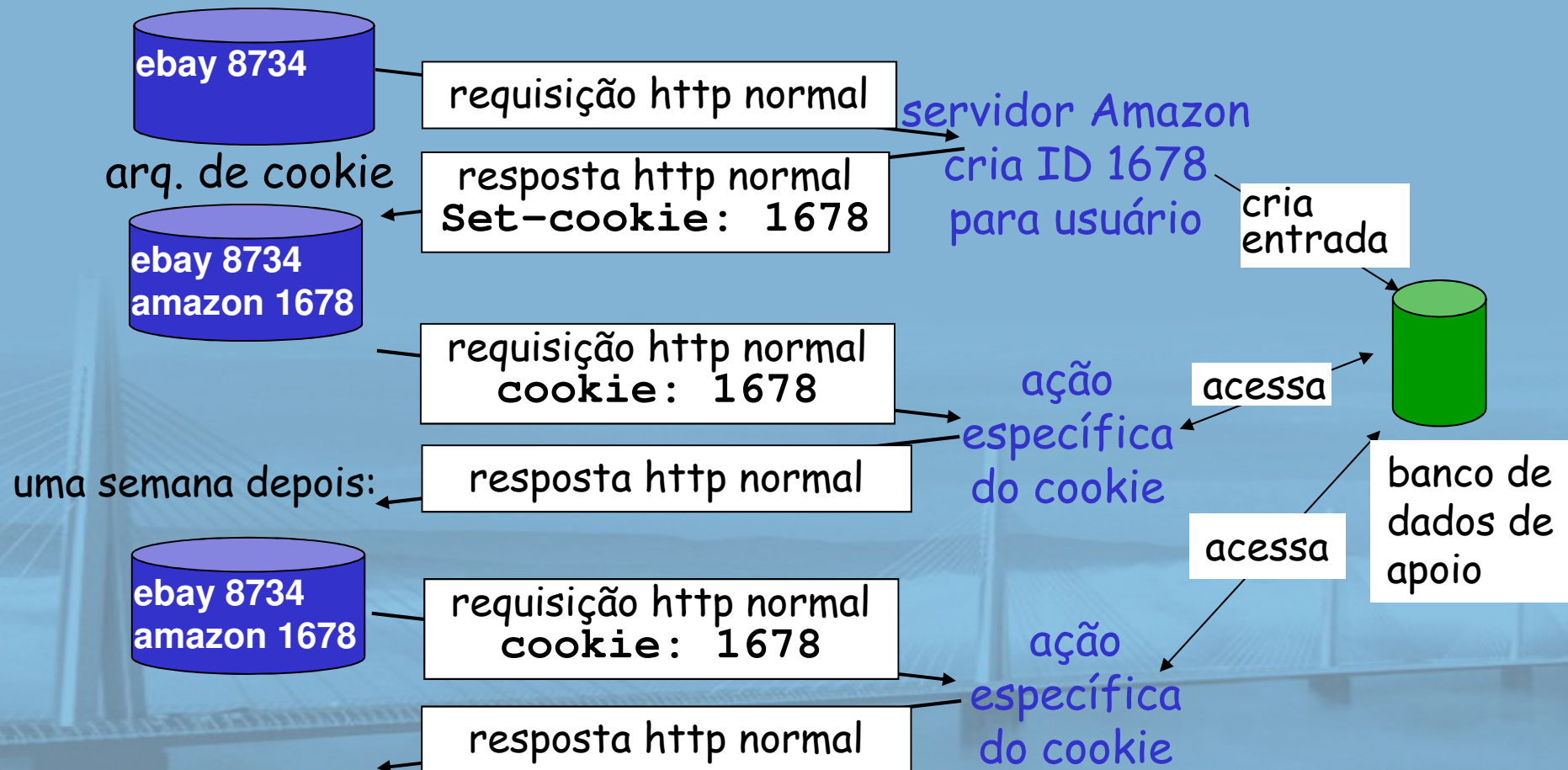
- 1) linha de cabeçalho de cookie da mensagem de *resposta* HTTP
- 2) linha de cabeçalho de cookie na mensagem de *requisição* HTTP
- 3) arquivo de cookie na máquina do usuário, controlado pelo navegador do usuário
- 4) banco de dados de apoio no site Web

Exemplo:

- ❑ Susana sempre acessa a Internet pelo PC
- ❑ visita um site de comércio eletrônico pela primeira vez
- ❑ quando as primeiras requisições HTTP chegam ao site, este cria:
 - ❖ ID exclusivo
 - ❖ entrada no banco de dados de apoio para o ID

cliente

servidor



O que os cookies podem ter:

- ☐ autorização
- ☐ carrinhos de compras
- ☐ recomendações
- ☐ estado da sessão do usuário (e-mail da Web)

Como manter o "estado":

- ☐ extremidades do protocolo: mantêm estado no emissor/receptor por múltiplas transações
- ☐ cookies: mensagens HTTP transportam estado

aparte

Cookies e privacidade:

- ☐ cookies permitem que os sites descubram muito sobre você
- ☐ você pode fornecer nome e e-mail aos sites

GET condicional

- ❑ **objetivo:** não enviar objeto se o cache tiver versão atualizada
- ❑ **cache:** especifica data da cópia em cache na requisição HTTP

If-modified-since:
<data>

- ❑ **servidor:** resposta não contém objeto se a cópia em cache estiver atualizada:

HTTP/1.0 304 Not
Modified

cache

servidor

