REDES DE COMPUTADORES E INTERNET

Camada de aplicação

Prof. Rogério Leão

Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P compartilhamento de arquivos

Camada de Aplicação.

Objetivos:

- conceitos, implementação, aspectos dos protocolos das aplicações de rede
  - Modelos de serviço da camada de transporte
  - Paradigma cliente-servidor
  - Paradigma peer-to-peer – p2p
- Aprender sobre protocolos examinando aplicações populares
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS
- Programando aplicações de rede
  - socket API

Algumas aplicações de rede

- E-mail
- Web
- Mensagem instantanea
- Login remoto
- P2P compartilhamento de arquivos
- Jogos de rede multi-usuarios
- Visualização de videos pela rede
- VOIP – telefone pela rede internet
- Conferencia com video em tempo-real
- Computação paralela

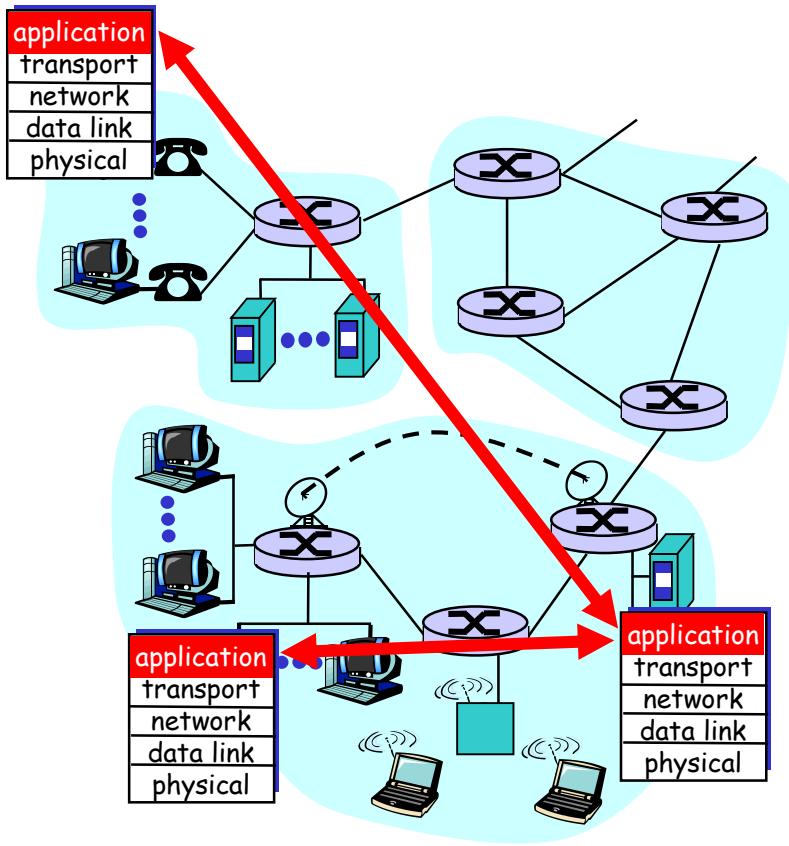
Criando uma aplicação de rede

Escrever programas que:

- Rodam em sistemas finais diferentes
- Comunicam-se pela rede
- e.x., Web: Software Web servidor(Apache) comunica-se com software navegador(netscape,...)

Nenhum software codifica dispositivos do nucleo da rede

- Dispositivos do nucleo da rede não são funcionais para camada de aplicação
- Este modelo facilita o desenvolvimento das apps



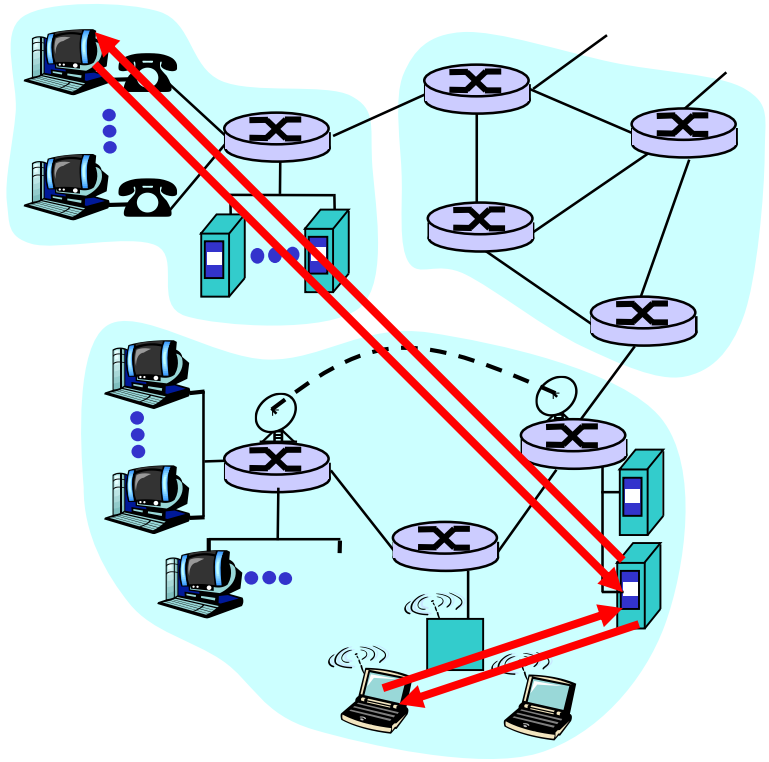
Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 DNS
- 2.5 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.6 P2P compartilhamento de arquivos

Arquitetura das aplicações de rede

- Cliente-servidor
- Peer-to-peer (P2P)
- Hibridas de cliente-servidor e P2P

Arquitetura Cliente-servidor



servidor:

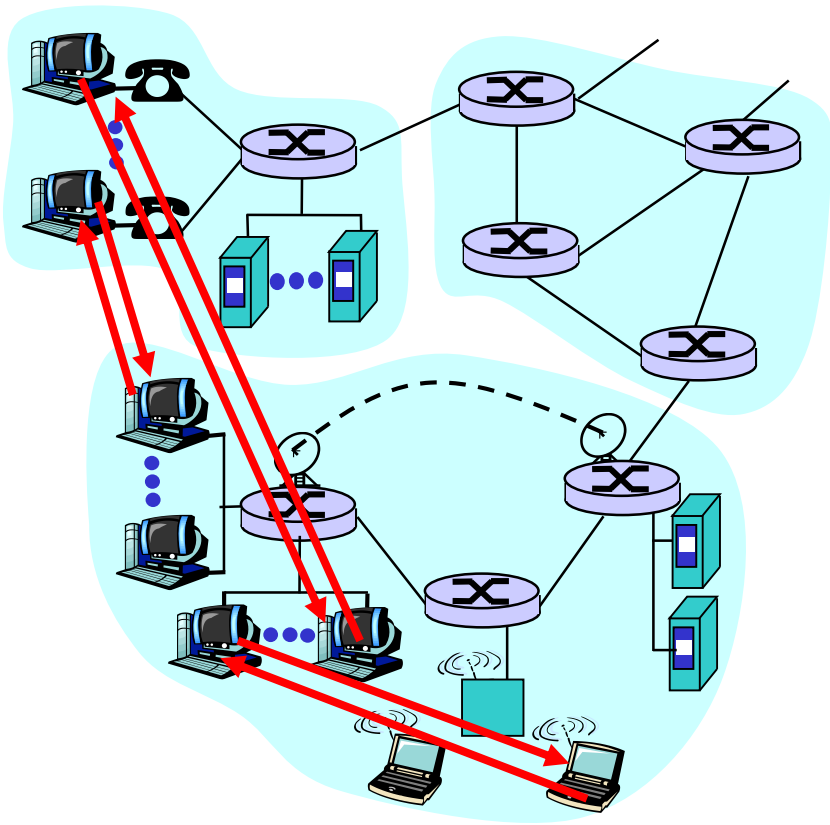
- Sempre ativo (on-line)
- Endereço de IP fixo

clientes:

- Comunica-se com o servidor
- Pode conectar-se constantemente ou de modo intermitente
- Endereço de IP dinâmico
- Não comunica-se diretamente com outro cliente.

Arquitetura P2P pura

- Não possui servidor (on-line)
- Sistemas finais conectam-se diretamente de forma arbitratia
- Pares ficam intermitentemente conectados e trocam endereços de IP esporadicamente.
- exemplo: Gnutella



Altamente escalável,
mas de dificil gestão.

Híbrido de cliente-servidor e P2P

Napster

- P2P
- Busca dos arquivos centralizada:
  - Usuário registra seu conteudo em servidores centrais.
  - Usuários consultam algum servidor para busca de arquivos.

Mensagem instantânea

- Bate-papo entre dois usuários é P2P.
- Descoberta de quais usuários estão on-line é centralizada:
  - Usuários registram seus endereços IP em servidores centrais quando estão on-line
  - Usuários contactam esses servidores para encontrar os amigos on-line e seus respectivos endereços IP

Comunicação de processos

Processo: programas rodando dentro de um host.

- No mesmo host, dois processos comunicam-se através comunicação inter-processo (definida pelo S.O.).
- processos em diferentes hosts comunicam-se por troca de mensagens pela rede

Processo cliente:
processo que inicia a comunicação

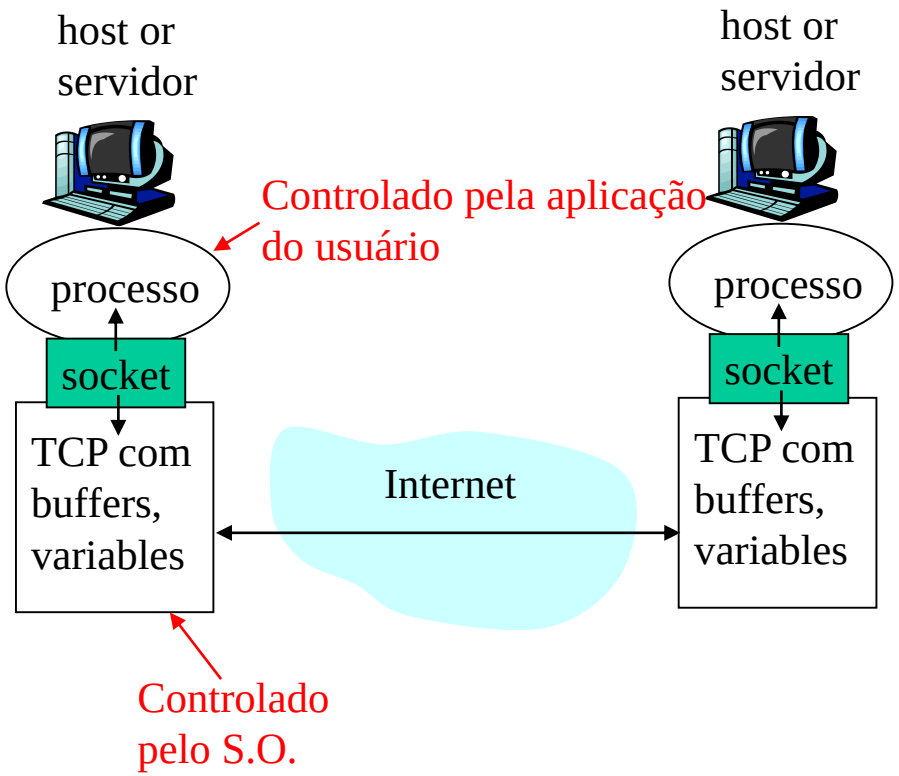
Processo servidor:
processo que espera o contato de algum cliente

- Nota: aplicações de arquitetura P2P têm processos clientes e processos servidores

Sockets

- processos enviam/recebem mensagens para/de seu socket

- socket é análogo a uma porta
  - Envia mensagens para fora
  - O envio das msgs depende da infraestrutura do outro lado da porta, que é responsável por levar as mensagens ao socket destino



Endereçamento de processos

- Para cada processo que recebe mensagens, deve haver um identificador
- Um host tem um único IP de 32-bit
- Q: O endereço IP do host que roda o processo é suficiente para identificar o processo ?
- R: Não, vários processos rodam no mesmo host.

- Identificação utiliza-se de endereço IP e número da porta associadas ao processo.
- Exemplos de números de portas:
  - HTTP Server: 80
  - Mail Server: 25

Protocolos da camada de aplicação definem

- Tipos de troca de mensagem, ex, mensagens de requisição e resposta
- Sintaxe da mensagem: quais são os campos da mensagem, tamanho, ordem, tipo
- Semântica dos campos, significado, para que serve
- Ações: quando e como os processos enviam e recebem as mensagens

Protocolos de dominio-publico

- Definidos por RFCs
- Permite interoperabilidade
- ex, HTTP, SMTP

Protocolos proprietarios:

- ex, KaZaA

O que o serviço de transporte de uma app precisa?

Perda de dados

- Alguns apps (e.x., audio) toleram perda de dados
- Outros apps (e.x., FTP, telnet) requerem 100% de integridade na transferência

Largura de banda

- Alguns apps (e.x., multimidia) requerem banda larga para se efetivarem
- outros apps conseguem trafegar com qualquer largura de banda

Praticando.....

Comandos:

Netstat: relação de processos/porta/endereço/status atuais.

ps: Gerenciador de processos do S.O Linux.

Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P compartilhamento de arquivos

Web e HTTP

Alguns conceitos:

- Páginas web consistem em objetos
- Objetos podem ser arquivos HTML, imagens JPEG, Scripts Java, arquivos de audio/video,...
- Um página web contem um arquivo base HTML que inclui referencias a outros objetos (imagens...)
- Cada objeto é endereçado por uma URL
- Exemplo de URL:

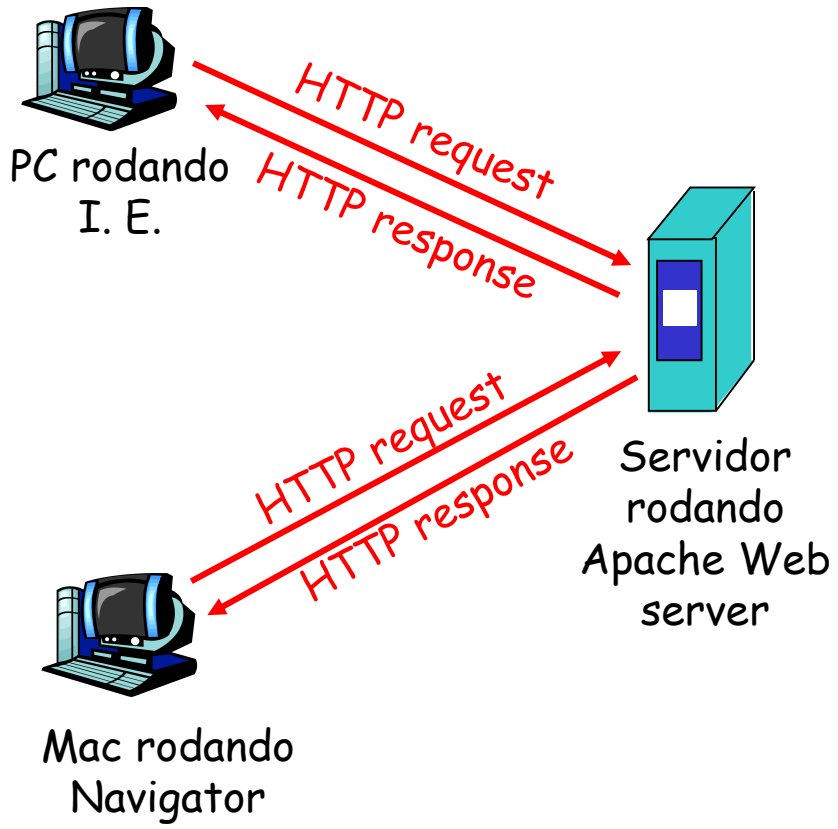
`www.fatecjales.com.br/imagem.gif`

host | Caminho do arquivo

HTTP visão

HTTP: hypertext transfer protocol

- É um protocolo da camada de aplicação.
- Modelo cliente/servidor
  - *cliente:* browser é quem requisita, recebe, mostra os objetos Web
  - *servidor:* envia objetos em resposta as requisições dos clientes.
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068



HTTP visão

Usando TCP:

- cliente inicia conexão TCP (cria um socket) para o servidor na porta 80
- servidor aceita conexão TCP do cliente
- Mensagens HTTP (na camada de aplicação) são trocadas entre o browser (HTTP client) e o Servidor Web (HTTP server)
- Conexão TCP é encerrada

HTTP é "stateless" (sem estado)

- O servidor não mantem informações de requisições passadas.

Tipos de Métodos

HTTP/1.0

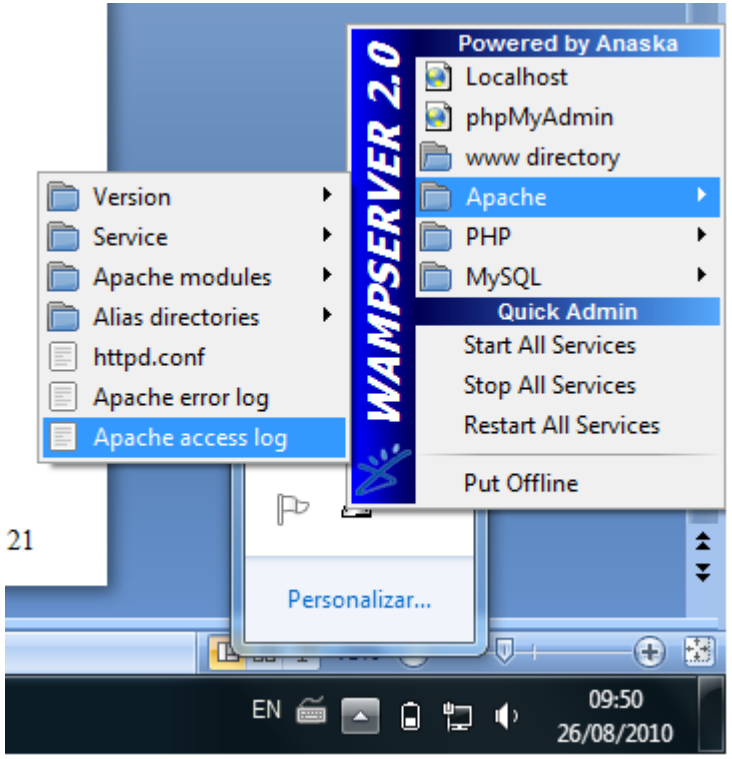
- GET
- POST
- HEAD
  - Similar ao método GET, mas utilizado somente para verificar se dados foram atualizados no servidor. (não traz do servidor imagens/textos, por exemplo).

HTTP/1.1

- GET, POST, HEAD
- PUT
  - Faz uploads de arquivos em um URL especifico.
- DELETE
  - Apaga arquivos em URL especifico.

Respostas e logs HTTP

Abra o arquivo de logs de acesso do servidor web (Apache) e verifique as mensagens de conexão com os navegadores.



Códigos de status de respostas HTTP

Alguns exemplos em conexões servidor-cliente:

200 OK

- Requisição Ok - respondida

301 Objeto movido permanentemente

- O novo local é mostrado nas mensages posteriores (Location:)

400 Requisição desconhecida

- O servidor não conseguiu interpretar a requisição

404 Não encontrado

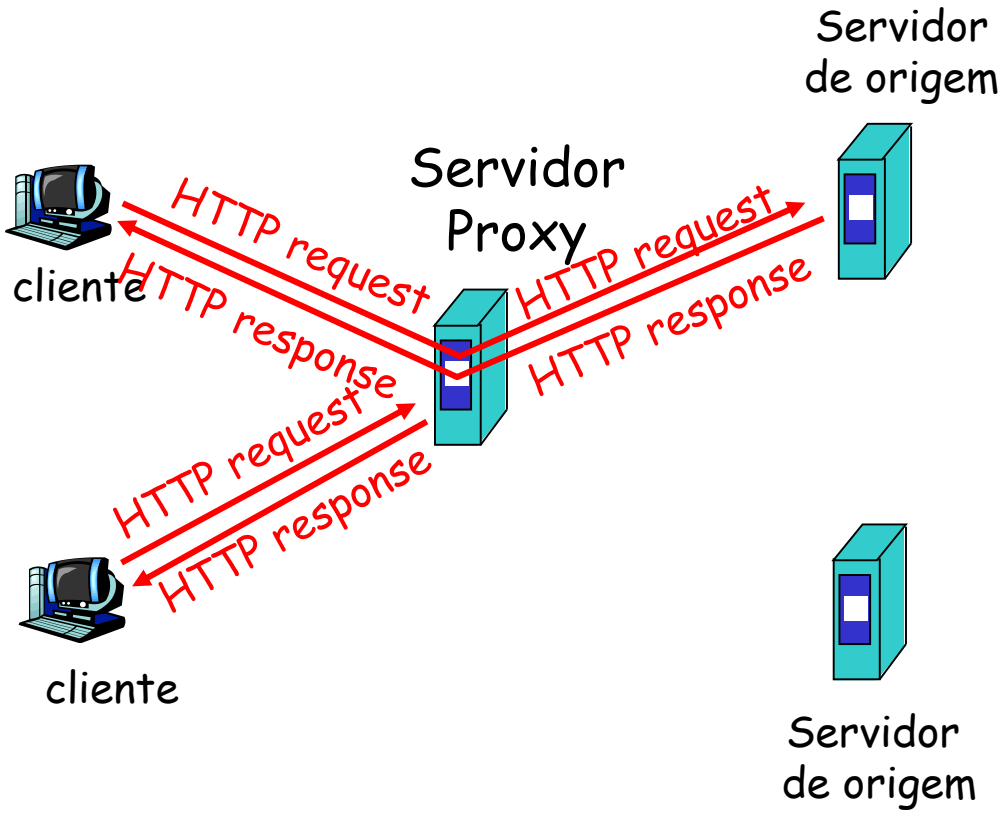
- Documento requisitado não encontrado no servidor

505 Versão HTTP não suportada

Web caches (Servidor proxy)

Objetivo: satisfazer a requisição do cliente sem envolver o servidor original.

- Configuração feita no navegador.
- browser envia todas requisições para o proxy.
  - Se o objeto requerido estiver em cache ele é retornado;
  - Se não estiver em cache o servidor proxy requisita-o no servidor original e retorna para o cliente.



Mais informações sobre Web caching

- Proxy (cache) atua tanto como servidor quanto como cliente.
- Tipicamente é instalado em ISP (universidades, companias)

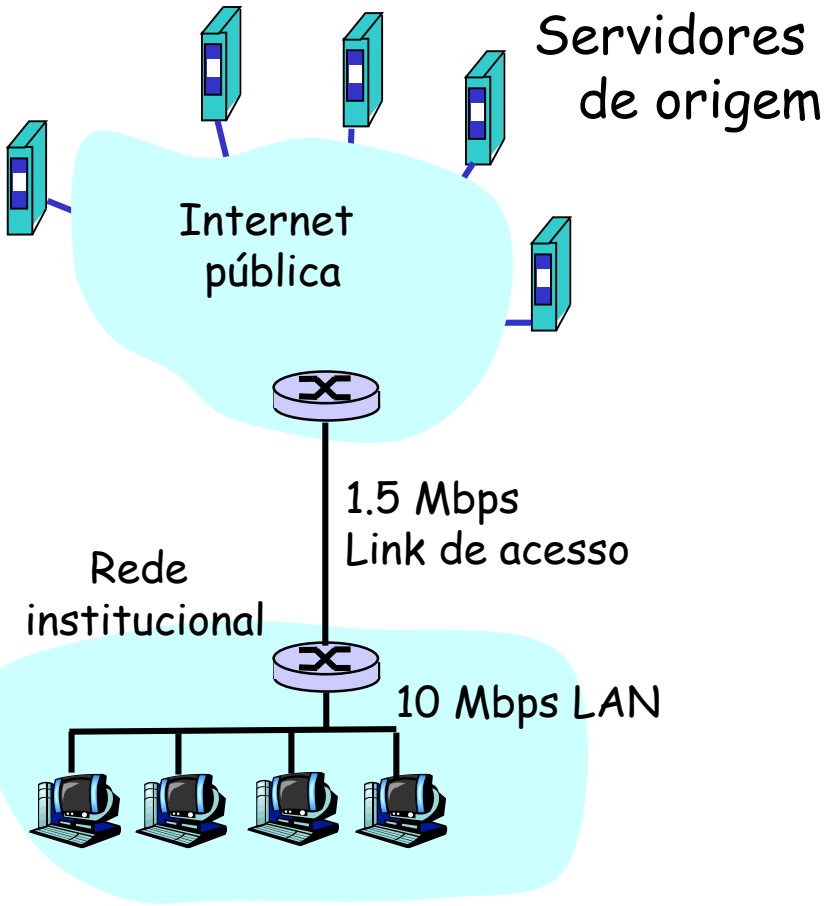
Para que utilizar cache ?

- Reduzir tempo de respostas para os clientes.
- Reduzir trafego na rede.

Exemplo de caching - Proxy

Computadores navegando diretamente em servidores de origem.

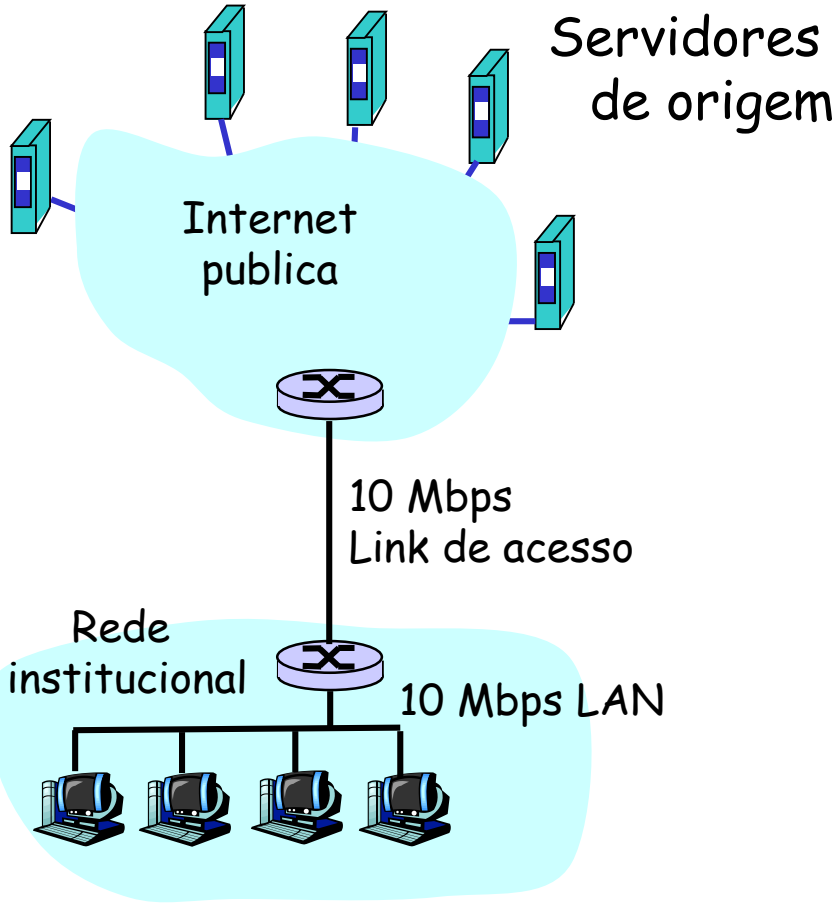
- Congestionamento do link de acesso.



Exemplo de caching - Proxy

Computadores navegando diretamente em servidores de origem.

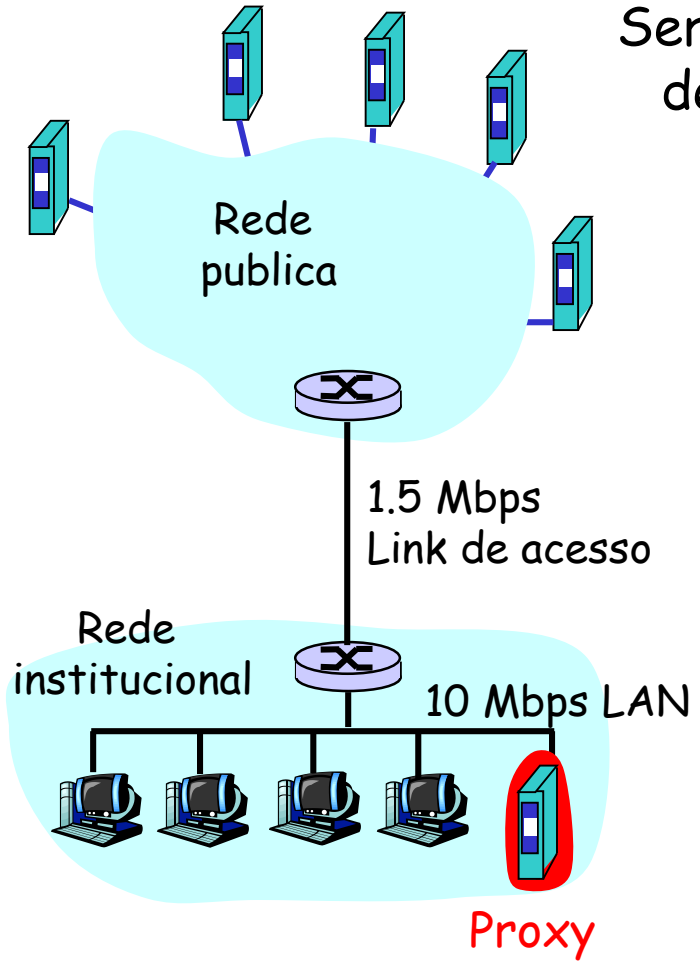
- Aumento do link de acesso – solução cara.



Exemplo de caching - Proxy

Computadores navegando por intermedio de um servidor proxy (cache).

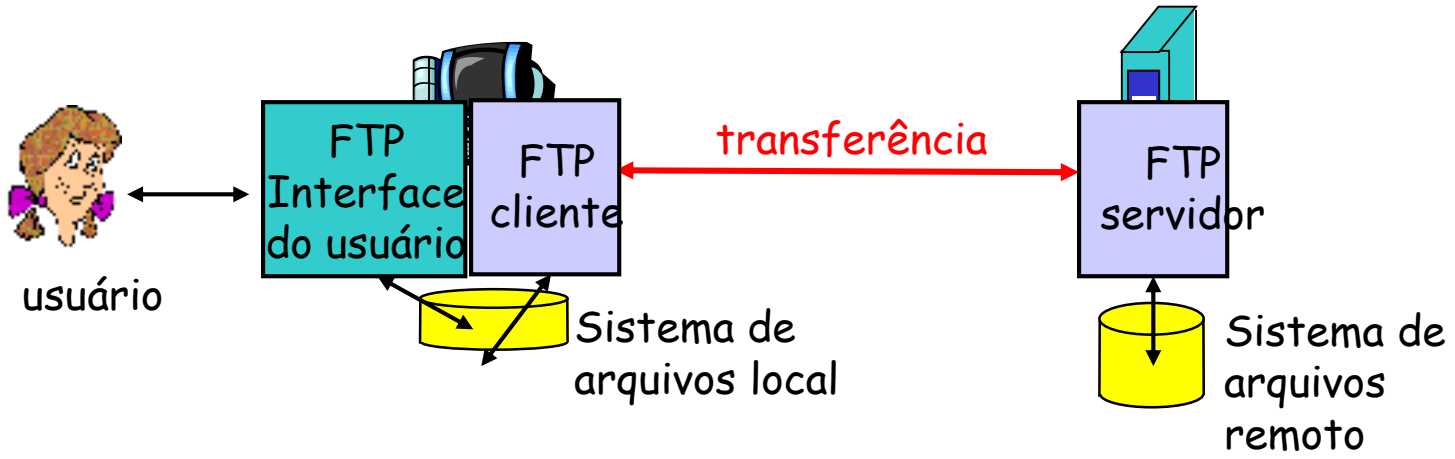
- Solução mais barata e eficiente.



Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P compartilhamento de arquivos

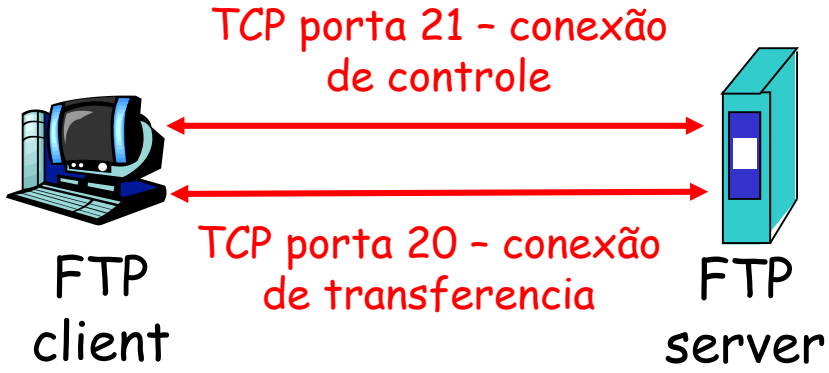
FTP: Protocolo de transferência de arquivos



- transferir arquivos para/de algum host remoto
- Modelo cliente/servidor
  - *cliente:* lado que inicia a transferencia
  - *server:* host remoto – escutando algum cliente
- ftp: RFC 959
- ftp server: port 21

FTP: dupla conexão

- FTP cliente contacta FTP servidor na porta 21, especificando TCP como protocolo de transporte.
- Cliente obtem autorização
- O cliente navega nos diretorios atraves de comandos de controle.
- Quando servidor recebe um comando para transferir algum arquivo, é aberta uma nova conexão TCP.
- Depois da transferencia a conexão é finalizada.



Comandos e respostas FTP

Comandos:

- **USER *username***
- **PASS *password***
- **LIST** retorna lista de arquivos e pastas do diretorio atual.
- **RETR filename** recebe (gets) arquivo do servidor
- **STOR filename** envia (puts) arquivo para o servidor

Respostas – código.

- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

Praticando.....

- Instalar e configurar Servidor FTP
- Instalar e configurar Cliente FTP
- Realização conexões, transferir arquivos e observar logs.

Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P compartilhamento de arquivos

DNS: Sistema de nome de domínios

pessoas: identificadores:

- RG, nome, CPF...

Host de internet, roteadores:

- End. IP (32 bit) – usados pelos computadores
- "nome", e.x., www.yahoo.com – usado pelos humanos

Q: tradução entre end. IP e os nomes quem faz ?

Domain Name System:

- *Banco de dados distribuido* implementado de forma hierarquica
- *Protocolo da camada de aplicação* hosts, roteadores, precisam traduzir os nomes em IPs para conseguirem o acesso.

DNS

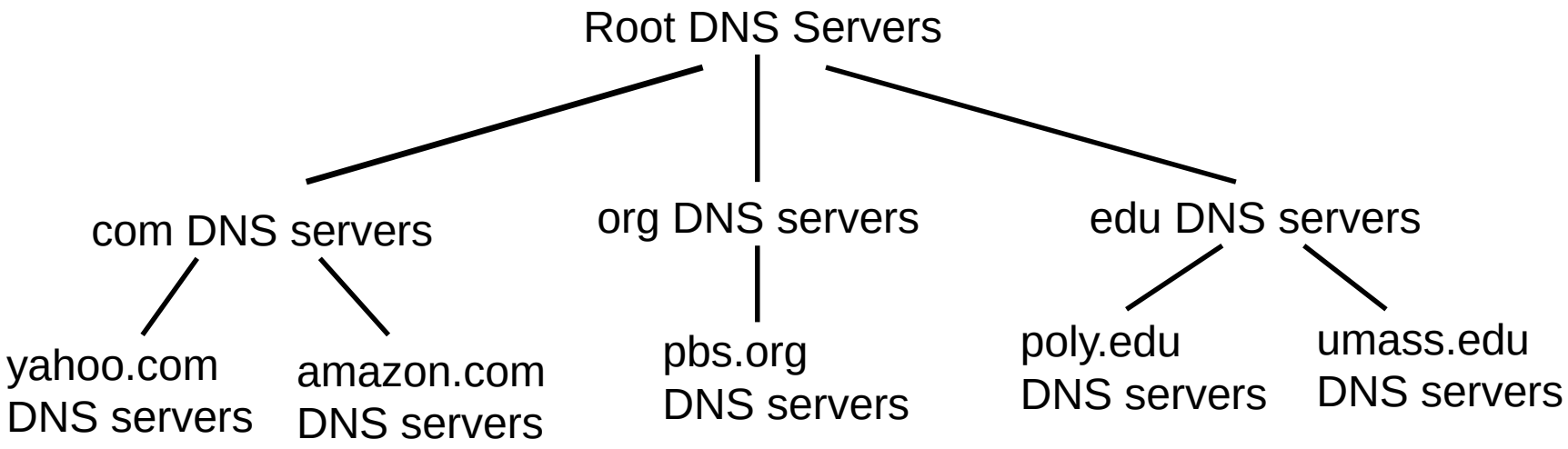
Serviços DNS

- Tradução de hostname para endereço IP
- Apelido de host
- Tradução de endereço IP para hostname (DNS reverso)

Porque não centralizar DNS?

- Um único ponto pode falhar e derrubar todo sistema
- Volume de tráfego
- Distancia do banco de dados centralizado
- Manutenção dificil

Banco de dados distribuido e hierárquico



Armazenamento dos IPs e DNS

- Os endereços ficam armazenados nos respectivos servidores DNS e são replicados a outros servidores como forma de backup do serviço.

Servidor de nomes local

- Cada ISP(provedor) pode ter um
  - Chamado comumente de "default name server"
- Quando um host faz uma consulta DNS, a requisição primeiro é enviada a este servidor local
  - Ele atua como um proxy, quando ele não tem a tradução desejada pergunta aos servidores Root .

DNS: caching e atualização de registros

- Toda vez que um servidor DNS aprende um registro, ele o armazena em cache
  - Este cache é apagado algum tempo depois(definido por configuração)
  - Ou é apagado quando não requisitado mais

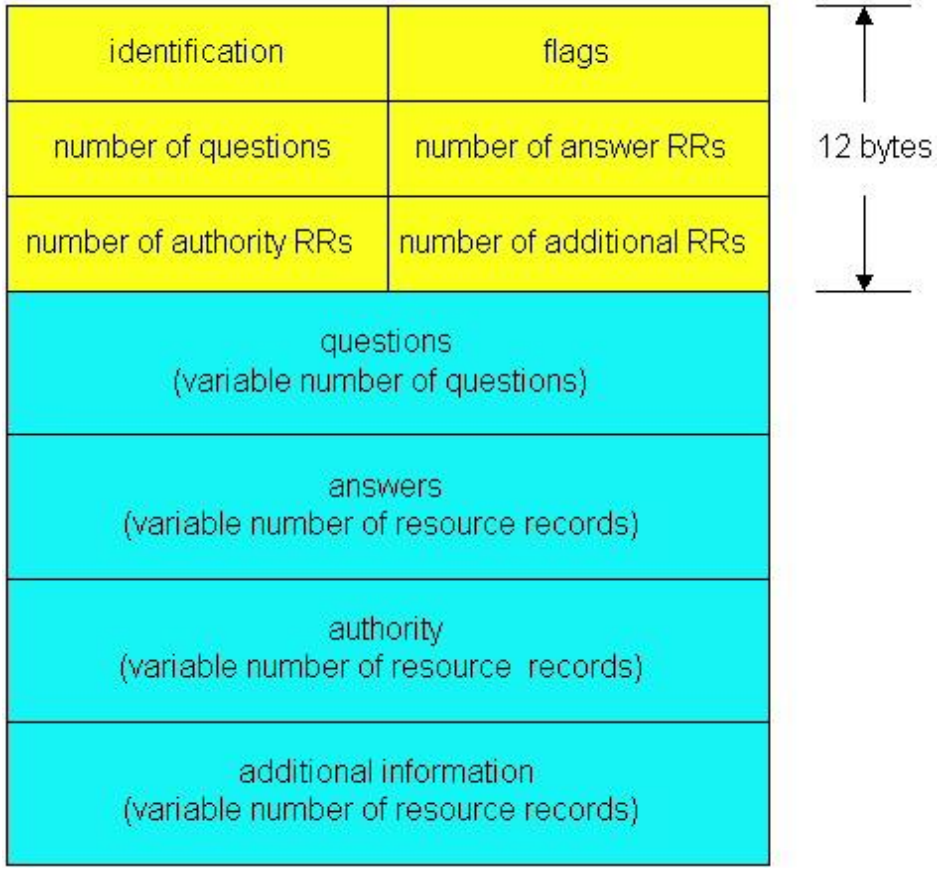
- Este mecanismo de atualização é padronizado pela IETF através do
  - RFC 2136

DNS - protocolo, mensagens

DNS protocolo : *consulta e responde* mensages

msg header

- id: 16 bit
- flags:
  - Consulta ou resposta



DNS – definições e segurança

- Porta padrão: 53
- Tipo transporte: UDP

Discussão sobre segurança:

Como seria um ataque DNS ?

Exercícios:

- Mudar configuração DNS e navegar.
- Executar comandos no prompt (ping, nslookup)

Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P compartilhamento de arquivos

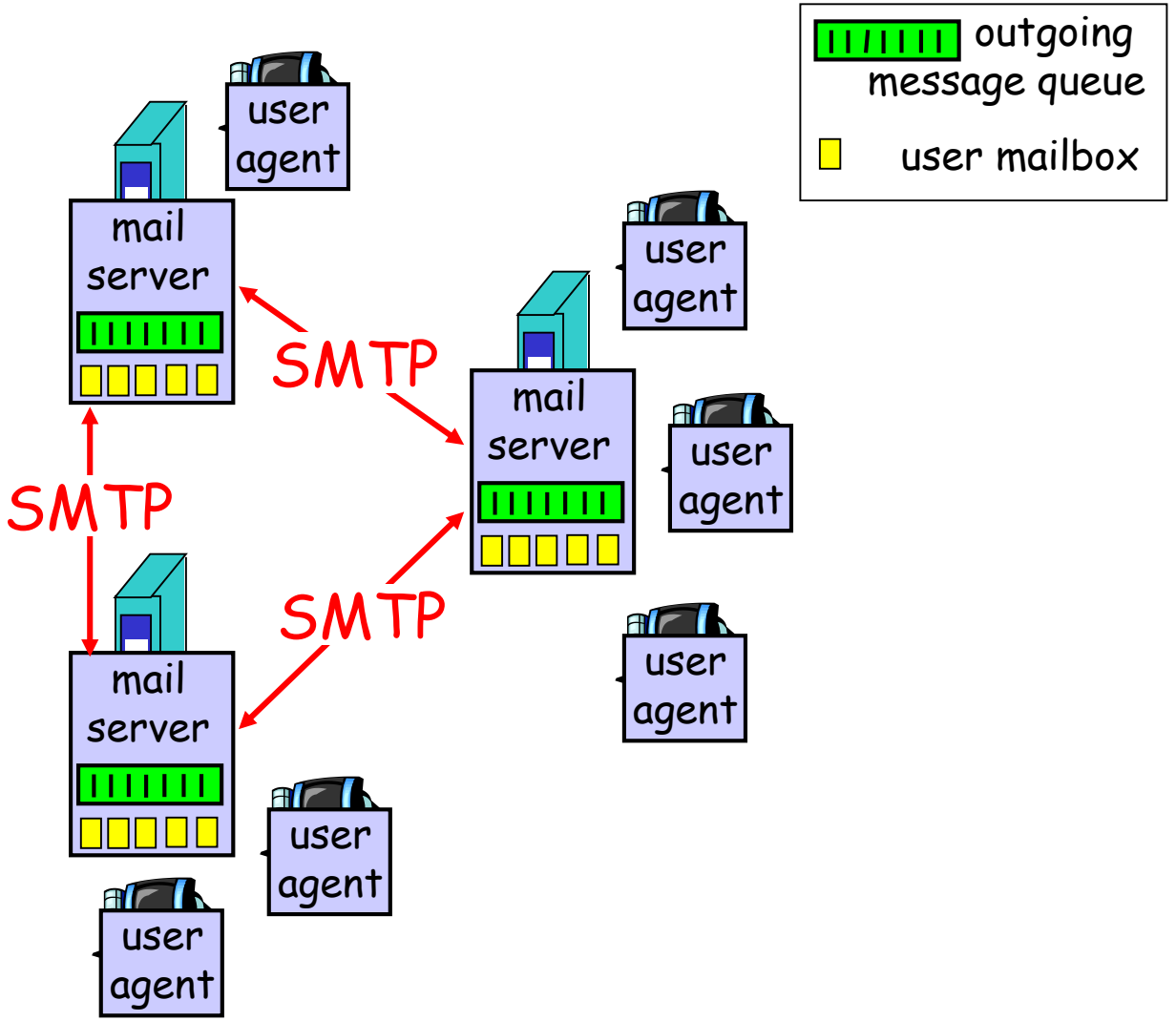
Correio eletrônico

3 grandes componentes:

- Agentes de usuário
- Servidores de e-mail
- Protocolo simplificado de transferência de e-mail: SMTP

Agente de usuário

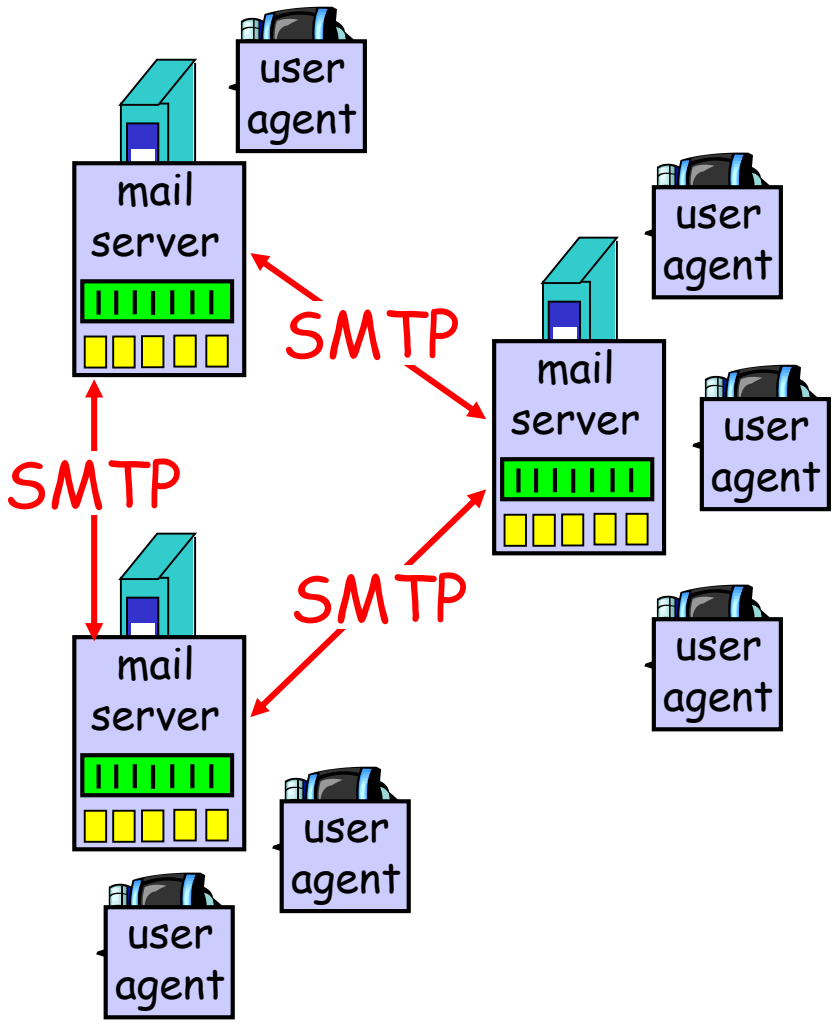
- Interface com usuário.
- Criar, editar e ler mensagens de e-mail
- e.x., Eudora, Outlook, elm, Netscape Messenger
- Enviar e receber mensagens gravadas nos servidores.



Correio eletrônico: servidores de e-mail.

Servidores de e-mail:

- Caixa de correios contém mensagens enviadas para os usuários
- Fila de mensagens de saida (a serem enviadas) para outro servidor
- Protocol SMTP transfere mensagens entre servidores de e-mail
  - cliente: envia e-mail para o servidor
  - "servidor": recebe e-mail de outro servidor

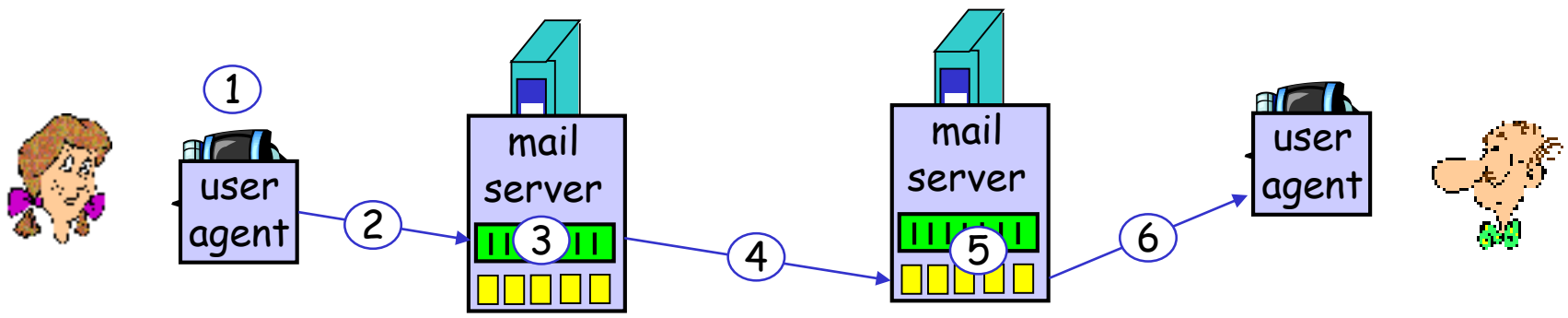


Correio eletrônico: SMTP [RFC 2821]

- usa TCP para transferir e-mail de clientes para servidores, porta 25
- Transferência direta: entre servidores diretamente
- Três fases para isso:
  - handshaking
  - Transferência de mensagens
  - encerramento
- Comandos/respostas – interação.
  - comandos: texto ASCII
  - resposta: códigos de status

Cenário: Alice envia mensagem para Bob

1) Alice usa agente de usuário para criar a mensagem e "to" `bob@fatec.edu.br`
2) O agente de usuário de Alice envia a mensagem para seu proprio servidor de e-mail; a mensagem é colocada na fila de envio.
3) O lado cliente SMTP abre conexão TCP o servidor de e-mail de Bob
4) O cliente SMTP envia a mensagem através da conexão TCP criada
5) O servidor de e-mail de Bob recebe a mensagem e a coloca na caixa de e-mail correspondente
6) Bob invoca seu agente de usuário que busca a mensagem no seu servidor para ser lida

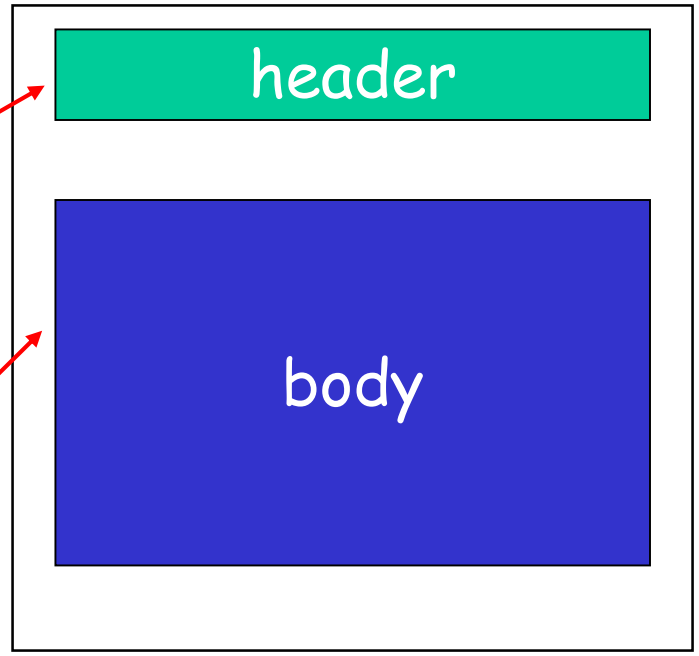


Formato da mensagem de e-mail.

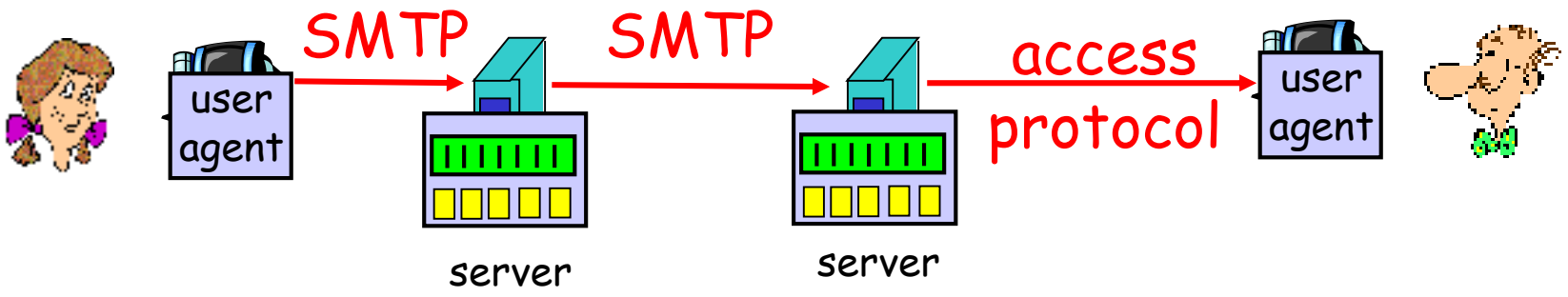
SMTP: protocolo para envio de mensagens e-mail

RFC 822: padrão para o formato de e-mails texto:

- header ex.,
  - Para:
  - De:
  - Assunto:
- body
  - A "mensagem", em caracteres ASCII somente.



Protocolos de acesso aos e-mails



- SMTP: faz a entrega de mensagens entre os servidores.
- Protocolos de acesso a e-mails: recuperam as mensagens do servidor para o cliente (user agent)
  - POP: Post Office Protocol [RFC 1939]
    - Autoriza o cliente (agent) a fazer o download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - Mais recursos(mais complexidade)
    - Manipula as mensagens diretamente no servidor
  - HTTP: Hotmail , Yahoo! Mail, etc.

Protocolo POP3

Fase de autorização

- Comandos clientes:
  - `user`: username
  - `pass`: password
- Respostas do servidor
  - `+OK`
  - `-ERR`

Fase de transação, cliente:

- `list`: lista mensagens
- `retr`: download das mensagens
- `dele`: apaga (delete)
- `Quit: sair`

```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 e IMAP

Mais sobre o POP3

- No exemplo anterior "download e apaga" as mensagens serão excluidas do servidor
- Se Bob mudar de programa cliente não conseguirá recuperar as mensagens anteriores.
- Se a configuração no prog. Cliente deixar a mensagem no servidor, ele conseguirá le-las

IMAP

- Deixa todas mensagens no servidor
- Permite que os usuarios organizem mensagens em pastas
- IMAP permite que o cliente acesse de varios programas se qualquer alteração nas mensagens.

Camada de Aplicação.

- 2.1 Principios das aplicações de rede
- 2.2 Web e HTTP
- 2.3 FTP
- 2.4 Correio eletrônico
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P compartilhamento de arquivos

P2P compartilhamento de arquivos.

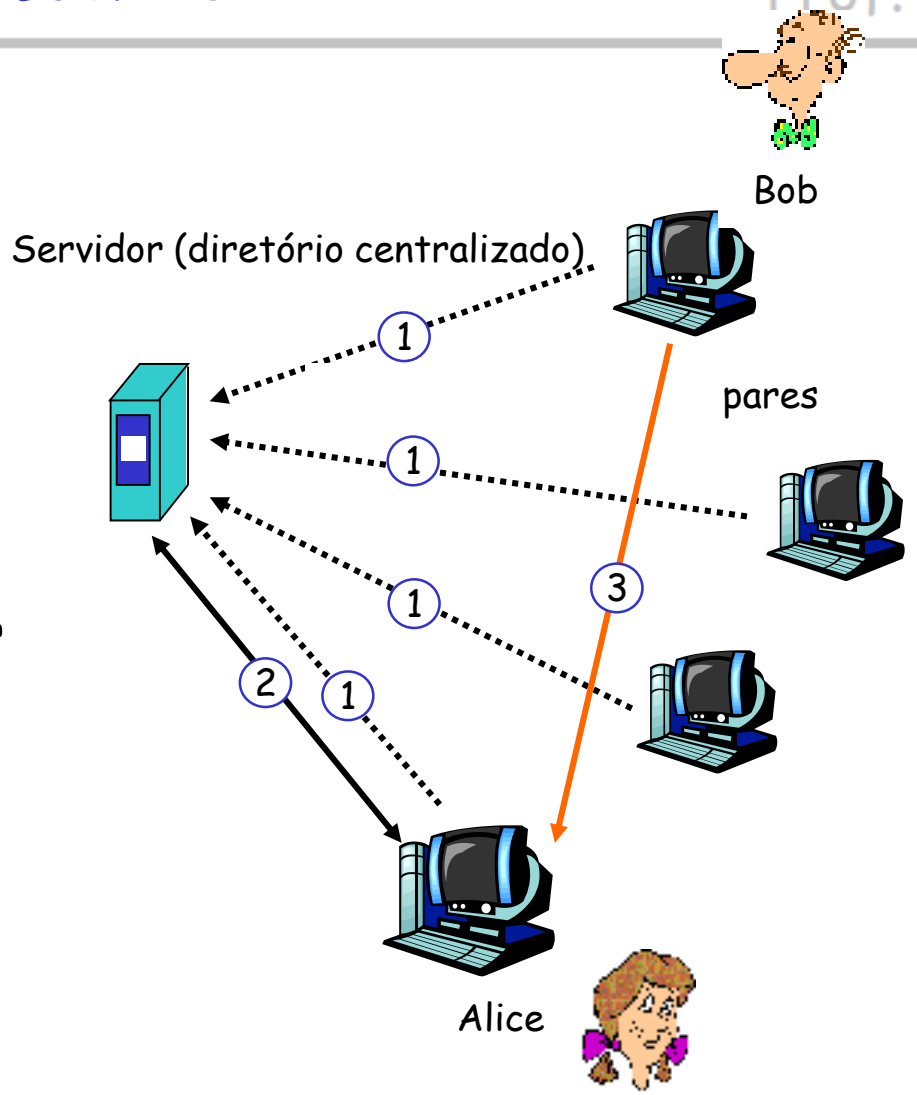
Exemplo

- Alice roda uma app cliente P2P em seu notebook
- De forma intermitente conecta a internet; recebe um novo end. IP a cada conexão
- Consulta por uma musica
- Sua app conecta outros pares que possuem a musica
- Alice escolhe um deles, Bob.
- E o arquivo é copiado do PC de Bob para o notebook de Alice
- Enquanto Alice faz o download, outros usuarios podem fazer uploading de Alice
- O notebook de Alice é cliente e servidor simultaneamente

P2P: diretório centralizado

Design original do "Napster"

1) Quando um par conecta, ele informa um servidor central:
   - Seu end. IP
   - Conteúdo que possui
2) Quando algum par quer consultar um determinado arquivo, ele pergunta ao servidor central



P2P: problemas com diretório centralizado

- Único ponto pode falhar (servidor)
- Gargalo de performance

transferencia de arquivos é direta entre os pares (descentralizada), mas as informações de conteudo de cada par é armazenada de forma centralizada.

Gnutella: consulta distribuida

- Servidores de informações de conteudo e pares são distribuidos.
  - Não existe servidor central único
  - Pares conectam a vários servidores
- Protocolo de domínio publico

Sumário

Finalizamos o estuda da camada de aplicação

- Arquitetura
  - cliente-servidor
  - P2P
  - Hibrida
- Protocolos:
  - HTTP
  - FTP
  - SMTP, POP, IMAP
  - DNS

Sumário

Mais detalhes: sobre os protocolos

- Tipicamente requerem/recebem mensagens:
  - cliente requisita dados ou serviço
  - servidor responde com dados ou códigos de status