Redes de Computadores

Parte 2

Evandro J.R. Silva

ejrs.profissional@gmail.com

Bacharelado em Ciência da Computação
Faculdade Estácio Teresina

05 a 06 de Agosto

Estácio

Sumário

Estácio

Protocolos de Camada de Aplicação

- Um protocolo de camada de aplicação define como **processos** de uma aplicação, que funciona em sistemas finais diferentes, passam mensagens entre si. Particularmente:
  - Os tipos de mensagens trocadas (ex.: requisição, resposta).
  - A sintaxe dos tipos de mensagem.
  - A semântica dos campos.
  - Regras para determinar quando e como um processo envia e responde mensagens.

Estácio

Protocolos de Camada de Aplicação

- Um protocolo de camada de aplicação define como **processos** de uma aplicação, que funciona em sistemas finais diferentes, passam mensagens entre si. Particularmente:
  - Os tipos de mensagens trocadas (ex.: requisição, resposta).
  - A sintaxe dos tipos de mensagem.
  - A semântica dos campos.
  - Regras para determinar quando e como um processo envia e responde mensagens.
- Existem protocolos cujas especificações são públicas (ex.: HTTP, SMTP) e outros cujas especificações são proprietárias (ex.: protocolo utilizado pelo Skype).

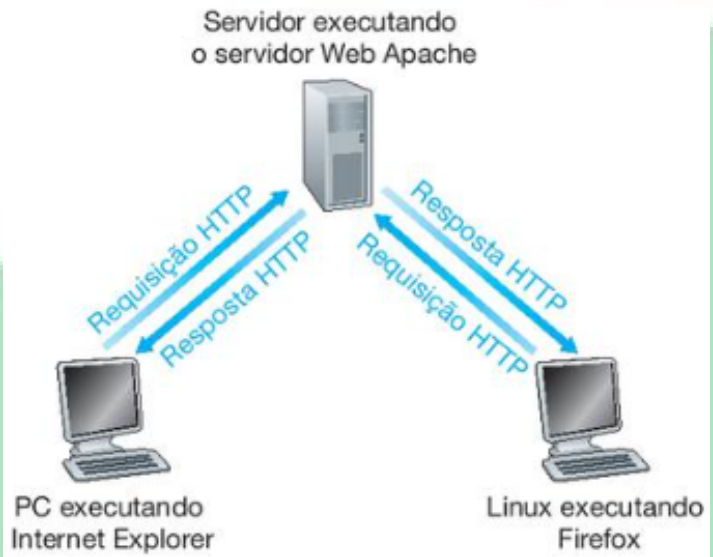
Estácio

HTTP

Estácio

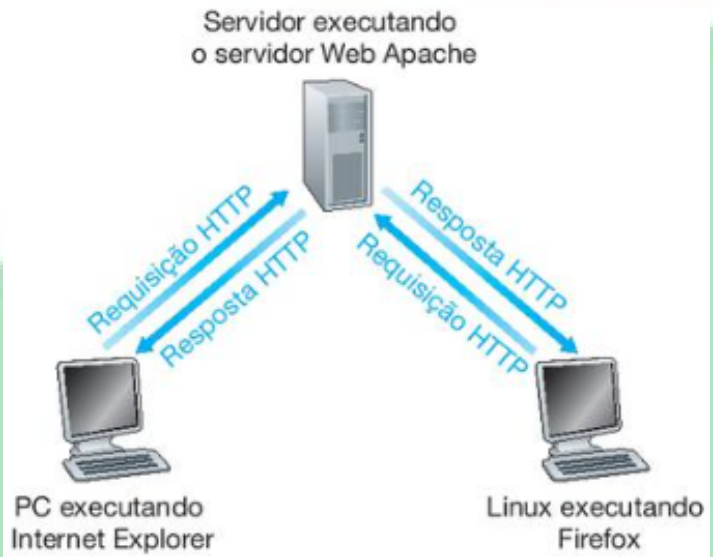
Descrição Geral

- Definido no RFC 1945 e RFC 2616.



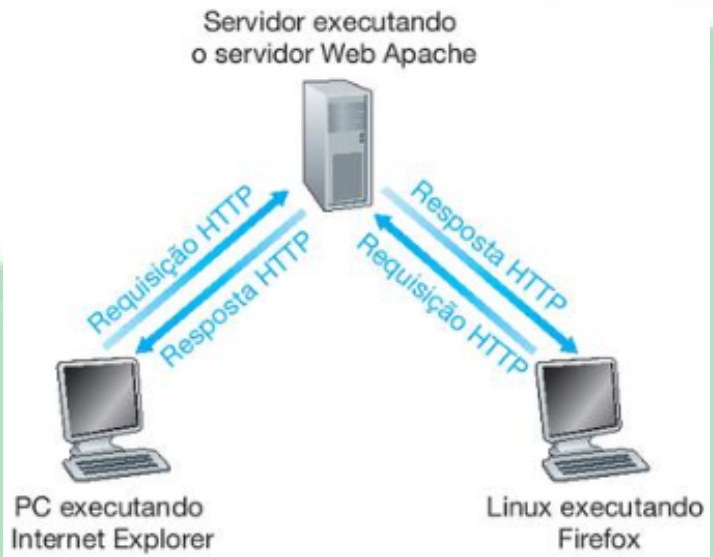
Descrição Geral

- Definido no RFC 1945 e RFC 2616.
- O protocolo é executado em dois programas: um cliente e um servidor.



Descrição Geral

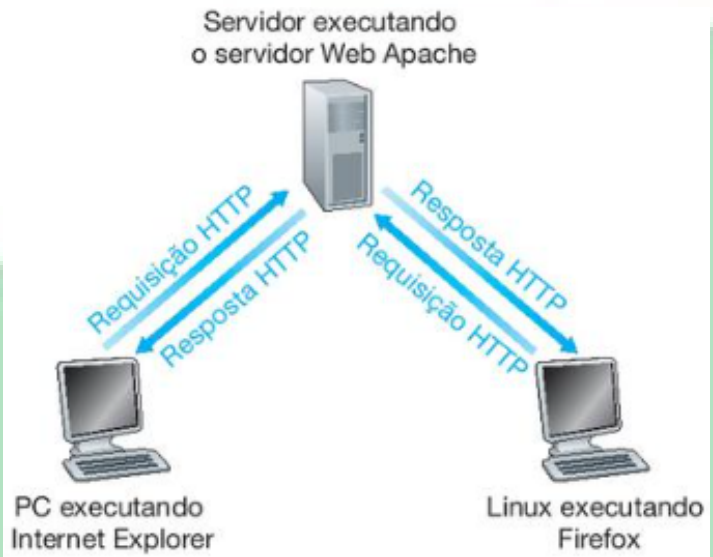
- Definido no RFC 1945 e RFC 2616.
- O protocolo é executado em dois programas: um cliente e um servidor.
- A troca de mensagens entre os dois programas é feita através de mensagens HTTP. O protocolo define a estrutura dessas mensagens e o modo são trocadas.



Estácio

Descrição Geral

- Definido no RFC 1945 e RFC 2616.
- O protocolo é executado em dois programas: um cliente e um servidor.
- A troca de mensagens entre os dois programas é feita através de mensagens HTTP. O protocolo define a estrutura dessas mensagens e o modo são trocadas.
  - Como clientes requisitam páginas aos servidores;
  - Como servidores transferem objetos aos clientes;



Descrição Geral

- O HTTP utiliza o TCP como protocolo de transporte
  - Portanto é uma aplicação cujas pontas criam uma conexão;
  - Não precisa se preocupar como será feita a entrega de dados, ou sobre a garantia de entrega (trabalho do TCP).

Estácio

HTTP e conexões não persistentes

- Em uma **conexão não persistente** as mensagens entre cliente e servidor acontecem através de várias conexões distintas.
- Entretanto, o HTTP tem como padrão a conexão persistente.

Estácio

HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:

Estácio

HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:
    1. O processo cliente HTTP inicia uma conexão TCP para o servidor `www.exemplo.com` na porta 80 (associados à conexão TCP teremos um *socket* no cliente e outro no servidor).

Estácio

HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:
    1. O processo cliente HTTP inicia uma conexão TCP para o servidor `www.exemplo.com` na porta 80 (associados à conexão TCP teremos um *socket* no cliente e outro no servidor).
    2. O cliente envia uma mensagem de requisição HTTP ao servidor por meio do seu *socket*. A mensagem inclui o nome do caminho, ou seja: `/meuExemplo/home.index`.

Estácio

HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:

2. O cliente envia uma mensagem de requisição HTTP ao servidor por meio do seu *socket*. A mensagem inclui o nome do caminho, ou seja: `/meuExemplo/home.index`.
3. O processo servidor HTTP recebe a mensagem pelo seu *socket*, extrai de seu armazenamento o objeto pedido, e o envia através de seu *socket* como uma mensagem de resposta.

HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:

3. O processo servidor HTTP recebe a mensagem pelo seu *socket*, extrai de seu armazenamento o objeto pedido, e o envia através de seu *socket* como uma mensagem de resposta.
4. O processo servidor ordena ao TCP que a conexão seja encerrada (o TCP vai encerrar somente quando tiver certeza de que o cliente recebeu a mensagem de resposta intacta).

Estácio

HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:

4. O processo servidor ordena ao TCP que a conexão seja encerrada (o TCP vai encerrar somente quando tiver certeza de que o cliente recebeu a mensagem de resposta intacta).
5. O cliente HTTP recebe a mensagem, e a conexão TCP é encerrada. A mensagem é um arquivo HTML, o qual contem referências às dez imagens.

Estácio

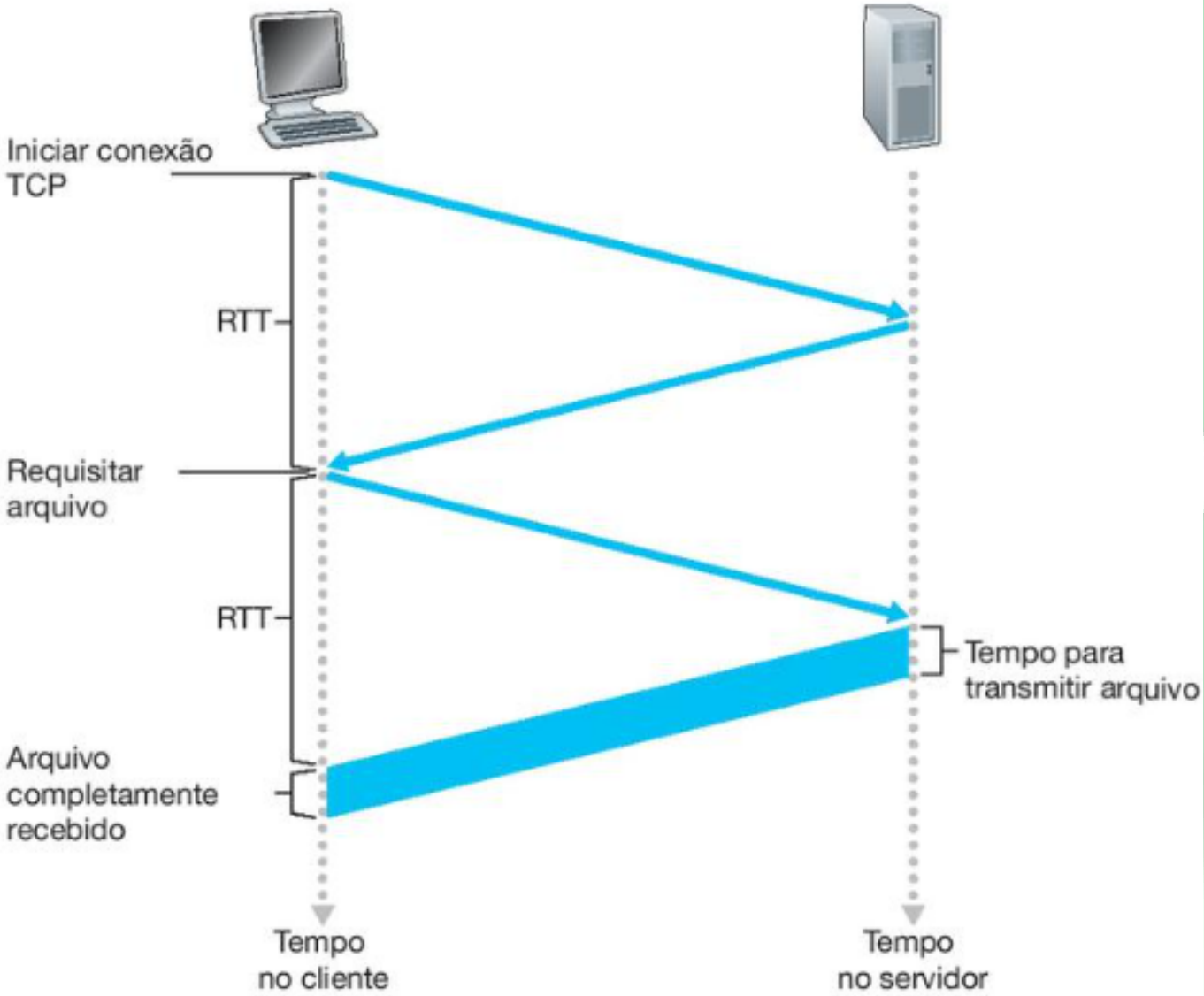
HTTP e conexões não persistentes

- EXEMPLO
  - Suponha o acesso a uma página contendo 11 objetos: 1 arquivo HTML e 10 imagens. Suponha também que todos os objetos estão no mesmo servidor, cuja url é `http://www.exemplo.com/meuExemplo/home.index`. Vamos ver o passo-a-passo:

5. O cliente HTTP recebe a mensagem, e a conexão TCP é encerrada. A mensagem é um arquivo HTML, o qual contem referências às dez imagens.
6. Os passos 1 - 4 são repetidos para cada imagem referenciada.

Estácio

HTTP e conexões não persistentes



HTTP e conexões persistentes

- Conexões não persistentes possuem algumas desvantagens
  - A necessidade de uma conexão para cada troca de mensagem pode sobrecarregar o servidor.
  - O carregamento de vários objetos leva mais tempo.

Estácio

HTTP e conexões persistentes

- Conexões não persistentes possuem algumas desvantagens
  - A necessidade de uma conexão para cada troca de mensagem pode sobrecarregar o servidor.
  - O carregamento de vários objetos leva mais tempo.
- Em conexões persistentes vários objetos e mensagens podem ser transferidos em uma única conexão (ex.: várias páginas em um mesmo site).

Estácio

HTTP e conexões persistentes

- Conexões não persistentes possuem algumas desvantagens
  - A necessidade de uma conexão para cada troca de mensagem pode sobrecarregar o servidor.
  - O carregamento de vários objetos leva mais tempo.
- Em conexões persistentes vários objetos e mensagens podem ser transferidos em uma única conexão (ex.: várias páginas em um mesmo site).
- As requisições em uma conexão persistente podem ser feitas mesmo antes de chegar a resposta para requisições pendentes (paralelismo $\rightarrow$ padrão no HTTP). Desta forma, a transferência pode ocorrer de forma ininterrupta.

Estácio

HTTP e conexões persistentes

- Conexões não persistentes possuem algumas desvantagens
  - A necessidade de uma conexão para cada troca de mensagem pode sobrecarregar o servidor.
  - O carregamento de vários objetos leva mais tempo.
- Em conexões persistentes vários objetos e mensagens podem ser transferidos em uma única conexão (ex.: várias páginas em um mesmo site).
- As requisições em uma conexão persistente podem ser feitas mesmo antes de chegar a resposta para requisições pendentes (paralelismo $\rightarrow$ padrão no HTTP). Desta forma, a transferência pode ocorrer de forma ininterrupta.
- Um servidor HTTP encerra uma conexão quando deixa de ser usada por algum tempo (configurável).

Estácio

Mensagem de requisição

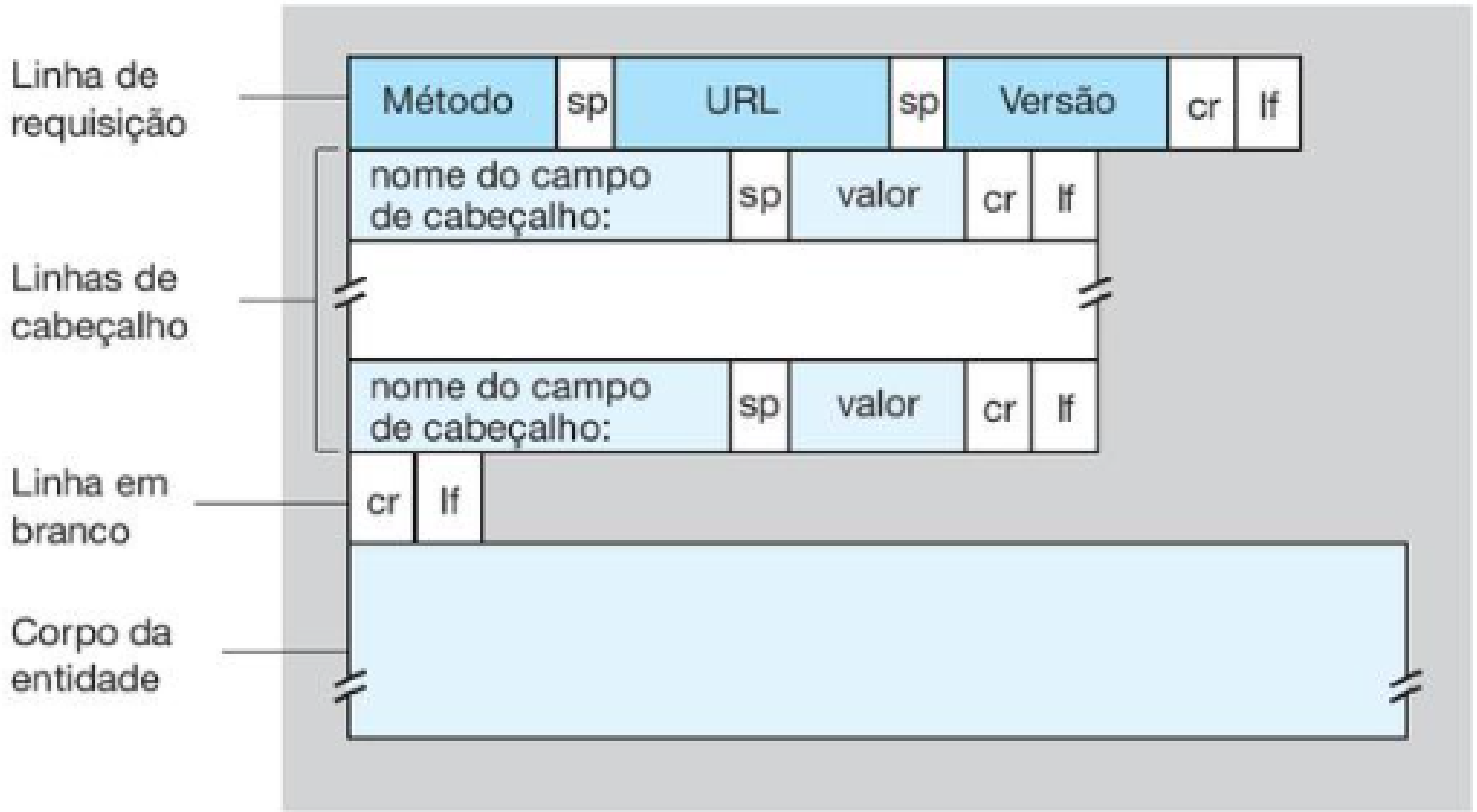
```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

- Essa mensagem tem 5 linhas (porém é possível ter mais linhas, ou menos também).
- **linha de requisição**: a primeira linha. Possui três campos: *método*, *url* e *versão* do HTTP. Outros métodos possíveis são POST, HEAD, PUT e DELETE.
- **linhas de cabeçalho**: as demais 4 linhas.
  1. Especifica o hospedeiro onde está o objeto (essa é uma informação exigida pelo *proxy*).
  2. Indica que a conexão deverá ser não persistente.
  3. Especifica o agente de usuário (o navegador) utilizado.
  4. Indica que o usuário prefere receber uma versão em Francês do objeto. Caso não exista, o servidor retornará com a versão *default*.

Estácio

Mensagem de requisição

FORMATO GERAL DE UMA MENSAGEM DE REQUISIÇÃO HTTP



Mensagem de resposta

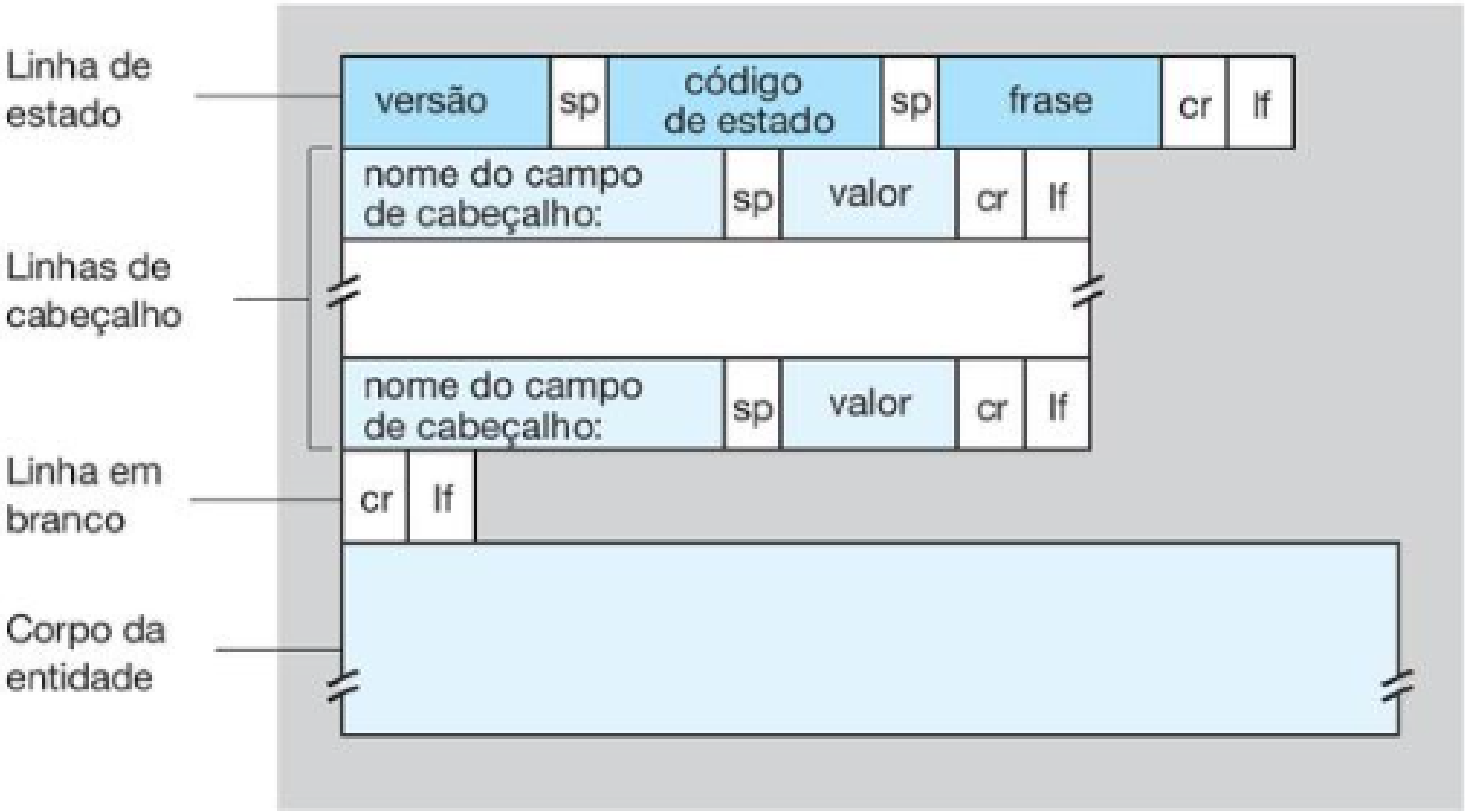
```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Aug 2011 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT
Content-Length: 6821
Content-Type: text/html

(dados dados dados dados dados ...)
```

- Três seções: **linha de estado**, **linhas de cabeçalho** e **corpo da entidade**.
- **linha de estado**: três campos → versão do protocolo, código de estado e mensagem de estado correspondente.
- **linhas de cabeçalho**
  1. O servidor informa que encerrará a conexão TCP após o envio da mensagem.
  2. Indica a data e hora em que a resposta foi criada e enviada.
  3. Mostra que a mensagem foi gerada por um servidor Apache.
  4. Indica a data e hora em que o objeto foi criado, ou sua última modificação.
  5. Indica o número de bytes do objeto que está sendo enviado.
  6. Mostra o tipo do objeto (neste caso um text/html).

Estácio

Mensagem de resposta

FORMATO GERAL DE UMA MENSAGEM DE RESPOSTA HTTP



SMTP

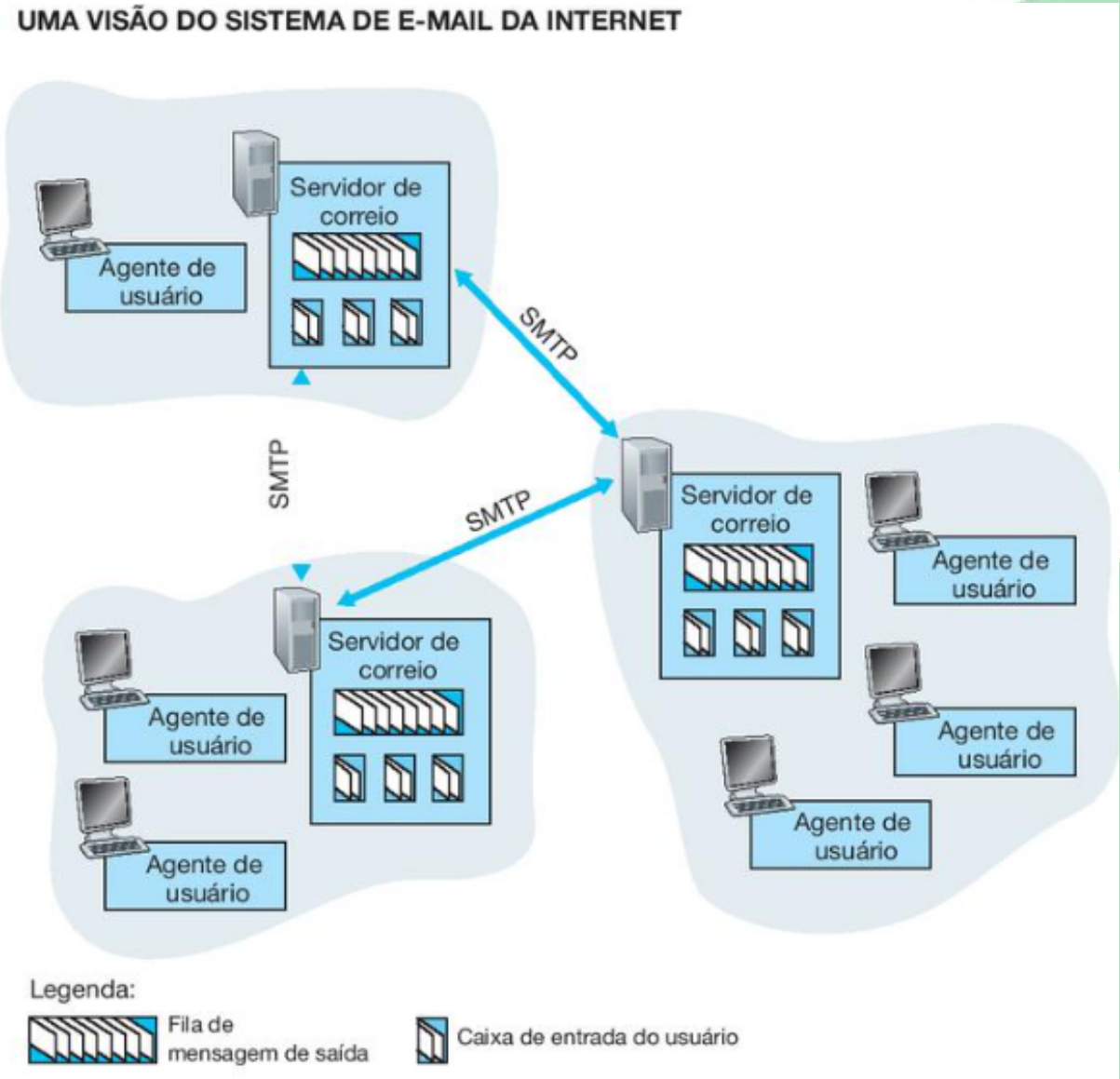
Estácio

Descrição Geral

- *Simple Mail Transfer Protocol*
- É o principal protocolo da camada de aplicação do correio eletrônico da Internet.
- Utiliza o TCP como protocolo de transporte.
- É mais antigo que o HTTP. Sua idade é perceptível pelas suas restrições (ex.: para os nossos tempos seu limite de dados é muito pequeno).

Estácio

Descrição Geral



Exemplo

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Estácio

DNS

Descrição Geral

- *Domain Name System*
- É um banco de dados distribuído, executado em uma hierarquia de **servidores de DNS** e também um protocolo da camada de aplicação (o qual permite que hospedeiros consultem o banco de dados distribuído).
- Utiliza o UDP como procolo da camada de transporte, e a porta 53.
- É utilizado também pelos outros protocolos da camada de aplicação (isso mesmo, o HTTP e SMTP, por exemplo).

Estácio

Como funciona

- A máquina do usuário executa o lado cliente do DNS.
- O navegador extrai o nome da URL e passa para o cliente DNS.
- O cliente DNS envia uma consulta com o nome do hospedeiro para um servidor DNS.
- O cliente DNS recebe uma resposta, incluindo o IP do hospedeiro.
- Outras aplicações (e protocolos) podem utilizar o IP para abrir uma conezão TCP.

Estácio

Outros serviços

- **Apelidos (*aliasing*)**: um hospedeiro pode ter um ou mais apelidos. Nesses casos até mesmo o **nome canônico** é difícil de decorar. O DNS pode fornecer o nome canônico associado a algum apelido, e também o IP.

Estácio

Outros serviços

- **Apelidos de servidor de correio**: imagine o email aluno@estacio.com; apesar de simples, o nome canônico do *estacio.com* pode ser mais complicado. O DNS também fornece o nome canônico e IP de um servidor de email.

Estácio

Outros serviços

- **Distribuição de carga**: alguns servidores são replicados, ou seja, o mesmo conteúdo está presente em mais de um servidor, cada um com seu IP. O DNS guarda o conjunto de IPs relacionado ao nome e passa para o cliente. No lado cliente o DNS faz um rodízio da ordem deles e, na prática, distribui a carga de requisições.

Estácio

Escalabilidade

- O DNS utiliza um grande número de servidores, organizados de maneira hierárquica, por isso, nenhum servidor terá, em si, todos os mapeamentos para todos os IPs.
- Três classes de servidores: raiz, de domnínio de alto nível (*top-level domain* - TLD) e autoritativos.
- **Servidores Raiz**: 13 servidores (denominados de A a M). Cada servidor destes pode ser um conjunto de servidores replicados (redundância).

Estácio

Escalabilidade

- O DNS utiliza um grande número de servidores, organizados de maneira hierárquica, por isso, nenhum servidor terá, em si, todos os mapeamentos para todos os IPs.
- Três classes de servidores: raiz, de domnínio de alto nível (*top-level domain* - TLD) e autoritativos.
- **Servidores Raiz**: 13 servidores (denominados de A a M). Cada servidor destes pode ser um conjunto de servidores replicados (redundância).
- **Servidores de Domínio de Alto Nível**: responsáveis por domínios como *com*, *org*, *net*, *edu*, *gov* e os de alto níveis de cada país, por exemplo *br*.

Estácio

Escalabilidade

- O DNS utiliza um grande número de servidores, organizados de maneira hierárquica, por isso, nenhum servidor terá, em si, todos os mapeamentos para todos os IPs.
- Três classes de servidores: raiz, de domnínio de alto nível (*top-level domain* - TLD) e autoritativos.
- **Servidores Raiz**: 13 servidores (denominados de A a M). Cada servidor destes pode ser um conjunto de servidores replicados (redundância).
- **Servidores de Domínio de Alto Nível**: responsáveis por domínios como *com*, *org*, *net*, *edu*, *gov* e os de alto níveis de cada país, por exemplo *br*.
- **Servidores Autoritativos**: servidores de organizações (universidades e empresas de grande porte) que possuem hospedeiros que podem ser acessados publicamente.

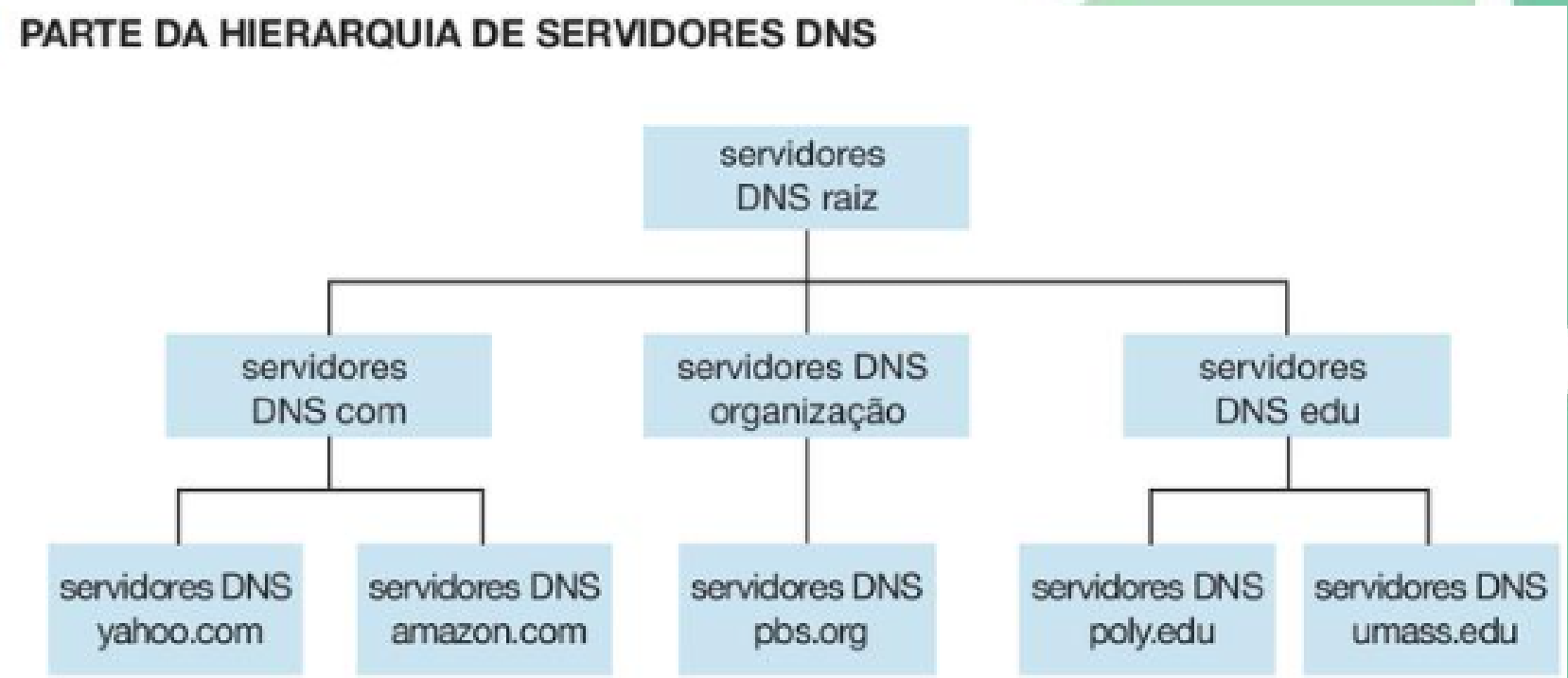
Estácio

Escalabilidade

- O DNS utiliza um grande número de servidores, organizados de maneira hierárquica, por isso, nenhum servidor terá, em si, todos os mapeamentos para todos os IPs.
- Três classes de servidores: raiz, de domnínio de alto nível (*top-level domain* - TLD) e autoritativos.
- **Servidores Raiz**: 13 servidores (denominados de A a M). Cada servidor destes pode ser um conjunto de servidores replicados (redundância).
- **Servidores de Domínio de Alto Nível**: responsáveis por domínios como *com*, *org*, *net*, *edu*, *gov* e os de alto níveis de cada país, por exemplo *br*.
- **Servidores Autoritativos**: servidores de organizações (universidades e empresas de grande porte) que possuem hospedeiros que podem ser acessados publicamente.
- Ainda existem também os servidores locais, os quais não fazer parte, estritamente, da hierarquia.

Estácio

Escalabilidade



Escalabilidade

Estácio

Camada de Transporte

Serviços de transporte disponíveis para aplicações

Estácio

Serviços de transportes para aplicações

- Transferência confiável de dados
  - Uma aplicação *sensível* (e-mail, transferência de arquivo, movimentação financeira, etc.) deve utilizar um protocolo que garanta a transferência confiável dos dados. A perda de dados para essas aplicações podem ser catastróficas.
  - Aplicações **tolerantes a perda** são pouco afetados se algumas informações não chegam ao destino.

Estácio

Serviços de transportes para aplicações

- Transferência confiável de dados

- Vazão
  - Vazão é a taxa pela qual o processo remetente pode enviar bits ao destinatário.
  - A vazão pode ser variável ao longo do tempo (outras aplicações enviando e recebendo dados). Um protocolo pode garantir uma vazão a uma taxa específica.
  - Uma aplicação pode solicitar uma vazão garantida e o protocolo vai garantir, no mínimo, a vazão solicitada.
  - Aplicações que possuem necessidae de vazão são conhecidas como **sensíveis à largura de banda**.

Estácio

Serviços de transportes para aplicações

- Temporização
  - Serviço que garante um tempo máximo para a entrega dos dados.
  - Exemplos: MS Teams, jogos online, etc.

Estácio

Serviços de transportes para aplicações

- Temporização
- Segurança
  - Uma aplicação pode necessitar de segurança para a transferência dos dados.
  - Um protocolo pode fornecer sigilo entre dois processos/usuários ao codificar a mensagem antes de ser enviada.

Estácio

Serviços de transportes para aplicações

Aplicação	Perda de dados	Vazão	Sensibilidade ao tempo
Transferência / download de arquivo	Sem perda	Elástica	Não
E-mail	Sem perda	Elástica	Não
Documentos Web	Sem perda	Elástica (alguns kbits/s)	Não
Telefonia via Internet/ videoconferência	Tolerante à perda	Áudio: alguns kbits/s – 1Mbit/s Vídeo: 10 kbits/s – 5 Mbits/s	Sim: décimos de segundo
Áudio/vídeo armazenado	Tolerante à perda	Igual acima	Sim: alguns segundos
Jogos interativos	Tolerante à perda	Poucos kbits/s – 10 kbits/s	Sim: décimos de segundo
Mensagem instantânea	Sem perda	Elástico	Sim e não

Estácio

Camada de Transporte
Serviços providos pela Internet

Serviços providos pela Internet

- A Internet (TCP/IP) disponibiliza dois protocolos de transporte para aplicações: TCP e UDP. Cada protocolo oferece um conjunto diferente de serviços.
- TCP
- UDP

Estácio

Serviços providos pela Internet

- A Internet (TCP/IP) disponibiliza dois protocolos de transporte para aplicações: TCP e UDP. Cada protocolo oferece um conjunto diferente de serviços.
- TCP
  - É orientado para conexão e confiabilidade de transferência de dados.
  - **Conexão**: o TCP faz cliente e servidor trocarem informações de controle de cama de transporte antes que as mensagens da camada de aplicação comecem a fluir.
  - **Transporte**: o TCP garante a entrega de todos os dados sem erro e na ordem correta, e também sem duplicação.
  - Além disso, o TCP também faz o controle de congestionamento.
- UDP

Estácio

Serviços providos pela Internet

- A Internet (TCP/IP) disponibiliza dois protocolos de transporte para aplicações: TCP e UDP. Cada protocolo oferece um conjunto diferente de serviços.
- TCP
- UDP
  - É um protocolo simplificado e leve. Não garante a transferência confiável de dados, a ordem correta de chegada e nem o controle de tráfego.

Estácio

Serviços providos pela Internet

- A Internet (TCP/IP) disponibiliza dois protocolos de transporte para aplicações: TCP e UDP. Cada protocolo oferece um conjunto diferente de serviços.
- TCP
- UDP
- Nenhum dos protocolos de transporte da internet provê serviço de vazão e temporização. As aplicações lidam de forma inteligente com essas limitações.

Estácio

Serviços providos pela Internet

Aplicação	Protocolo de camada de aplicação	Protocolo de transporte subjacente
Correio eletrônico	SMTP [RFC 5321]	TCP
Acesso a terminal remoto	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
Transferência de arquivos	FTP [RFC 959]	TCP
Multimídia em fluxo contínuo	HTTP (por exemplo, YouTube)	TCP
Telefonia por Internet	SIP [RFC 3261], RTP [RFC 3550] ou proprietária (por exemplo, Skype)	UDP ou TCP

Estácio

UDP

Estácio

UDP

- Definido no RFC 768;
- É um protocolo não orientado a conexão;
- Basicamente seu funcionamento é pegar mensagens do processo da aplicação, anexar os campos de número da porta de origem e de destino para o serviço de multiplexação/demultiplexação, adicionar dois outros pequenos campos e passar o segmento resultante para a camada rede.

Estácio

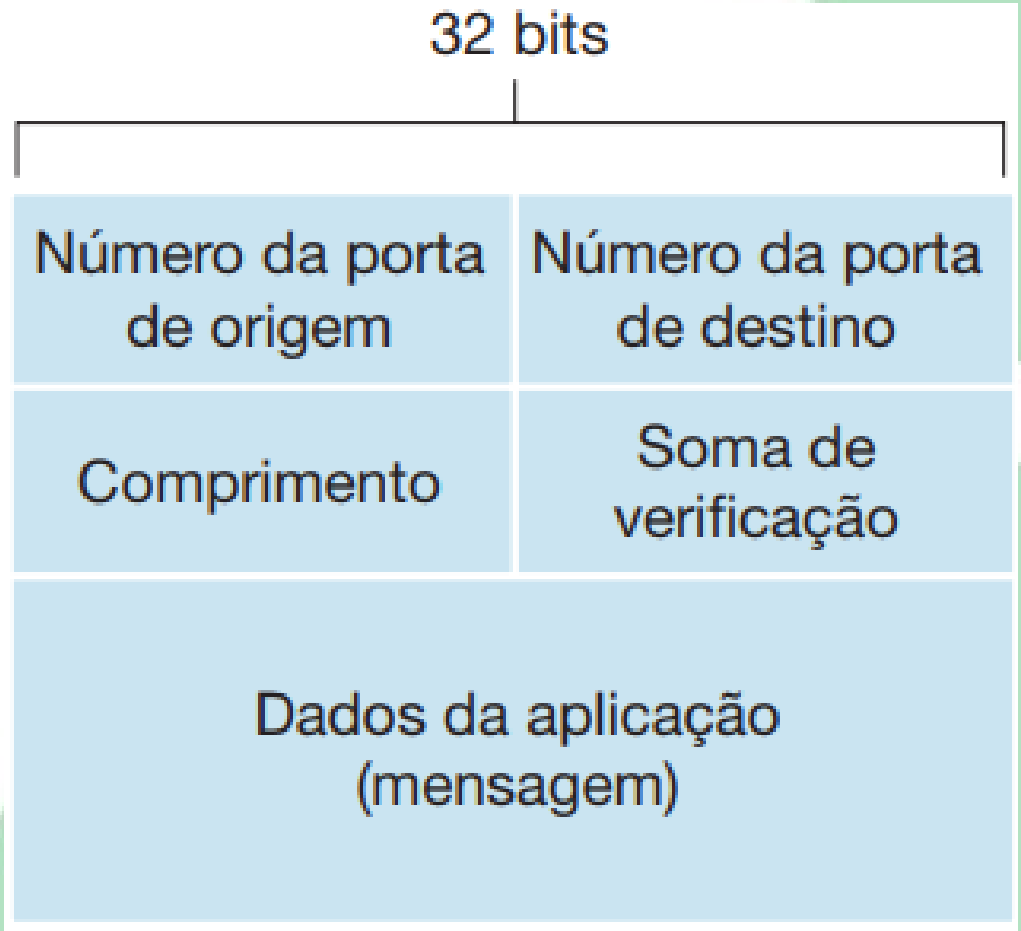
UDP

- Definido no RFC 768;
- É um protocolo não orientado a conexão;
- Basicamente seu funcionamento é pegar mensagens do processo da aplicação, anexar os campos de número da porta de origem e de destino para o serviço de multiplexação/demultiplexação, adicionar dois outros pequenos campos e passar o segmento resultante para a camada rede.

Estácio

UDP

- Estrutura do Segmento UDP



UDP

- **Soma de verificação** (*checksum*): serve para o UDP detectar se algum bit do segmento foi alterado. É opcional.
- **Como funciona**: no remetente todas as palavras de 16 bits são somadas. Ao fim da soma, teremos um resultado. O complemento deste resultado é a soma de verificação. No destinatário todas as palavras são somadas novamente, desta vez acrescentando a soma de verificação. O resultado deve ser 1111111111111111 (16 bits 1). Se houver qualquer bit 0, há um erro no segmento.
- Existem ainda mais detalhes que podem ser visualizados nos RFCs, ou no livro do Frouzan (a partir da página 712).

Estácio

TCP

Estácio

TCP

- É orientado a conexão (**ponto a ponto**);
- Provê um serviço ***full-duplex***, ou seja, os dados de remetente e destinatário podem fluir ao mesmo tempo. Também é um s
- A conexão é feita através de uma apresentação de três vias, ou *3-way handshake*, ou seja, o cliente inicia enviando ao servidor um segmento TCP especial. O servidor responde com outro segmento TCP especial e, por fim, o cliente responde com um terceiro segmento especial. Neste ponto os parâmetros para transferência de dados são estabelecidos.

Estácio

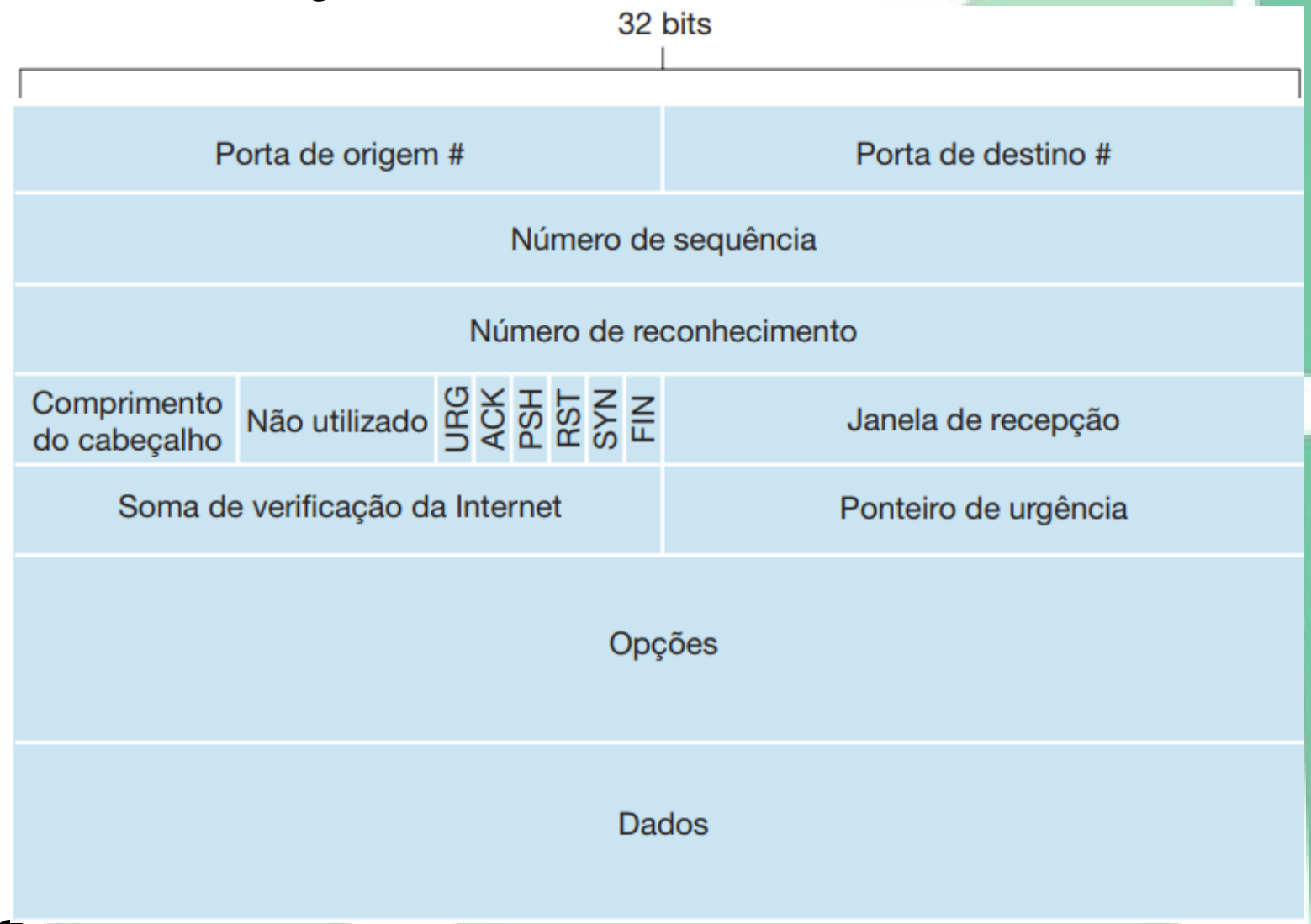
TCP

- Estabelecida a conexão, o TCP recebe a mensagem vinda do *socket* e o repassa ao **buffer de envio**, onde será dividido em partes menores, chamados segmentos, os quais são passados para a camada de rede.
- A **unidade máxima de transmissão** (MTU — *Maximum transmission unit*) é o valor do maior quadro de enlace que pode ser enviado pelo remetente. A partir desse valor é possível calcular o MSS (*Maximum Segment Size*), ou **tamanho máximo do segmento**, ou seja, a quantidade máxima de dados da camada de aplicação que cabem no segmento de forma que, junto com o cabeçalho do TCP, o segmento inteiro caiba no quadro.

Estácio

TCP

- Estrutura do segmento TCP



TCP

- **Porta de destino/origem**: valor da porta utilizada pelo processo de aplicação.
- **Número de sequência** e **de reconhecimento**: números (de 32 bits cada) utilizados pelo TCP para o serviço confiável de transferência de dados.
- **Janela de recepção**: 16 bits usados para controle de fluxo. Indica o número de bytes que um destinatário está disposto a aceitar.
- **Comprimento do cabeçalho**: 4 bits que especificam o comprimento do cabeçalho TCP em palavras de 32 bits.
- **Soma de verificação**: análogo ao *checksum* do UDP.

Estácio

TCP

- **Opções**: opcional e de comprimento variável. Usado quando remetente e destinatário negociam o MSS, ou como um fator de aumento de escala da janela para utilização em redes de alta velocidade.
- **Flag**: 6 bits:
  - **ACK**: Indica se o segmento contém um reconhecimento para um segmento que foi recebido com sucesso.
  - **RST**, **SYN** e **FIN**: usados para estabelecer e encerrar uma conexão.
  - **PSH**: indica que o destinatário deve passar os dados para a camada superior imediatamente.
  - **URG**: indica que há dados nesse segmento que a entidade da camada superior do remetente marcou como urgente. A localização do último byte dos dados urgentes é indicado no campo **Ponteiro de urgência**.

Estácio

Camada de Rede
Internet Protocol (IP)

Estácio

IP

- Um **endereço IPv4** é um endereço de 32 bits.
- $2^{32} = 4.294.967.296$ valores possíveis.
- Duas notações: **binária** e **decimal pontuada**.
  - 01110101 10010101 00011101 00000010
  - 117.149.29.2
- De início o IPv4 usava o **endereçamento com classes**: A, B, C, D e E.

	Primeiro byte	Segundo byte	Terceiro byte	Quarto byte
Classe A	0–127			
Classe B	128–191			
Classe C	192–223			
Classe D	224–239			
Classe E	240–255			

Estácio

IP

Classe	*Número de Blocos*	*Tamanho do Bloco*	*Aplicação*
A	128	16.777.216	Unicast
B	16.384	65.536	Unicast
C	2.097.152	256	Unicast
D	1	268.435.456	Multicast
E	1	268.435.456	Reservado

Estácio

IP

- **Endereços classe A**: eram designados a grandes organizações com um grande número de hosts ou roteadores conectados.
- **Endereços classe B**: destinavam-se às organizações de médio porte com dezenas de milhares de hosts ou roteadores conectados.
- **Endereços classe C**: destinavam-se a pequenas organizações com um pequeno número de hosts ou roteadores.
- **Endereços classe D**: foram projetados para multicast. Cada endereço nessa classe é usado para definir um grupo de hosts.
- **Endereços classe E**: reservados para uso futuro.

- Muitos endereços eram desperdiçados!

Estácio

IP

- **Netid** e **Hostid**
  - No endereçamento com classe, os tipos A, B e C são divididos em netid e hostid.
  - Classe A: 1 byte para netid e 3 bytes para hostid.
  - Classe B: 2 bytes para netid e 2 bytes para hostid.
  - Classe C: 3 bytes para netid e 1 byte para hostid.
- Esgotamento de endereços
  - Dado o tempo, a quantidade de endereços das classes A e B acabaram. Enquanto isso, um bloco de endereços C é muito pequeno para a maioria das organizações de porte médio.

Estácio

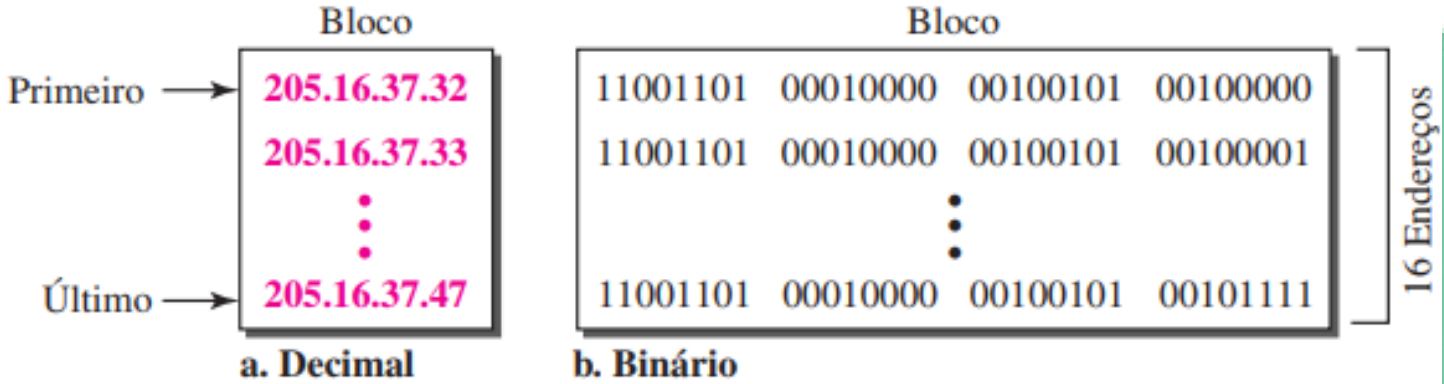
IP

- **Netid** e **Hostid**
  - No endereçamento com classe, os tipos A, B e C são divididos em netid e hostid.
  - Classe A: 1 byte para netid e 3 bytes para hostid.
  - Classe B: 2 bytes para netid e 2 bytes para hostid.
  - Classe C: 3 bytes para netid e 1 byte para hostid.
- Esgotamento de endereços
  - Dado o tempo, a quantidade de endereços das classes A e B acabaram. Enquanto isso, um bloco de endereços C é muito pequeno para a maioria das organizações de porte médio.
  - Solução: endereçamento sem classes.

Estácio

Endereçamento sem Classes

- Quando uma entidade precisa ser conectada à Internet, um bloco de endereços é concedido.
- O tamanho do bloco é variável.
- Restrições:
  - Os endereços em um bloco devem ser contíguos.
  - O número de endereços deve ser uma potência de 2.
  - O primeiro endereço tem de ser igualmente divisível pelo número de endereços.



- No exemplo os endereços são contíguos (205.16.37.32 ... 205.16.37.47). A quantidade de enderços (16) é potência de 2. O primeiro endereço em binário, quando convertido em decimal é divisível por 16.

Endereçamento sem Classes

- **Máscara**
  - Usado para definir um bloco de endereços.
  - É um número de 32 bits, no qual os *n* números mais à esquerda sao 1, e os 32 - *n* números mais à direita são 0.
  - Notação: x.y.z.t/*n*, onde x.y.z.t define um dos endereços e /*n* estipula a máscara.
- **Primeiro endereço**: Os 32 - *n* bits mais à direita são configurados como 0.
- **Último endereço**: Os 32 - *n* bits mais à direita são configurados como 1.

Estácio

Endereçamento sem Classes

- Exemplo:
  - Um dos endereços é 205.16.37.39/28. Como saber o primeiro endereço no bloco?
  - O endereço em binário é

    11001101 00010000 00100101 00100111

    Se configurarmos os 32 - n (32 - 28 neste caso) bits mais à direito como 0, teremos

    11001101 00010000 00100101 001**0000**

    que em decimal é 205.16.37.32.
  - Como encontrar o último endereço? Em vez 0, configuramos como 1:

    11001101 00010000 00100101 0010**1111**

    o que dá 205.16.37.47.
  - O número de endereços pode ser calculado como $2^{32-n} = 2^{32-28} = 2^4 = 16$.

Estácio

Endereçamento sem Classes

- Outra forma é utilizar a máscara inteira. Ou seja, quando $n = 28$ isso significa que a máscara é:

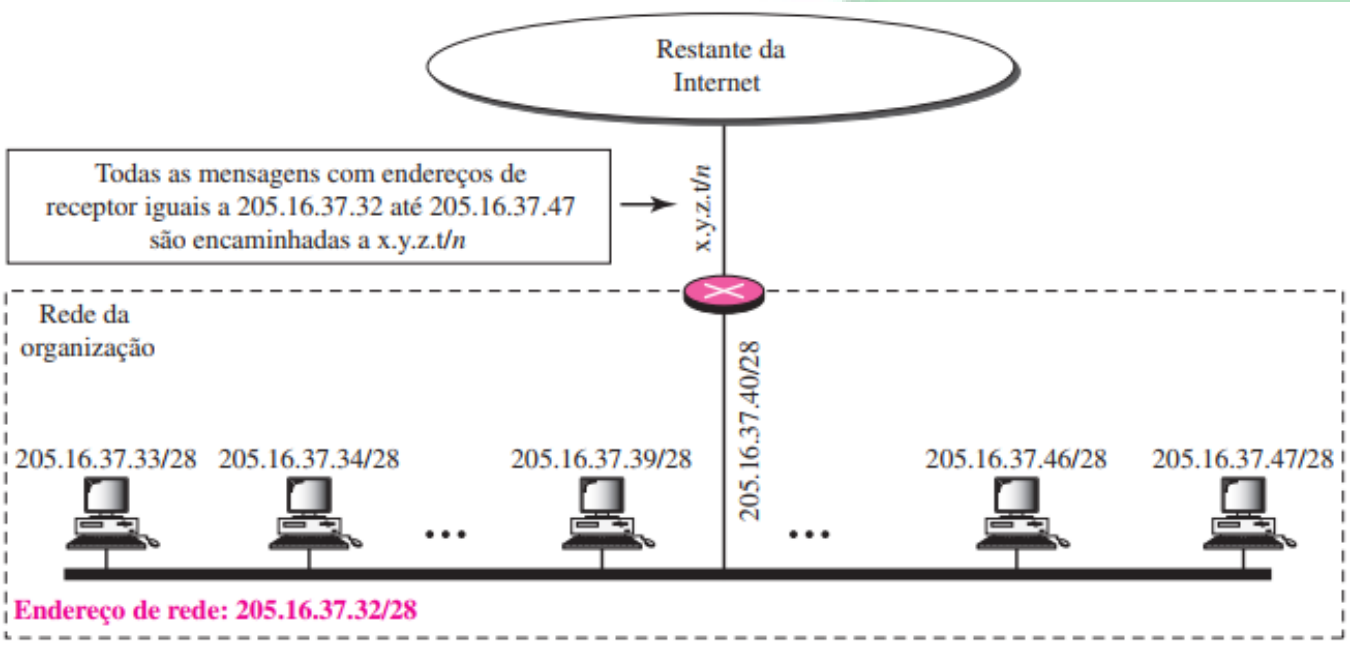
  11111111 11111111 11111111 11110000
- O primeiro endereço é descobertao ao ser aplicar o **AND** entre o endereço fornecido e a máscara.
- O último endereço é obtido aplicando o **OR** com o endereço fornecido e o complemento da máscara, ou seja:

  00000000 00000000 00000000 00001111
- A quantidade de endereços é o valor decimal do complemento da máscara + 1. No nosso caso, os quatro últimos bits são 1111. Então temos $2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$. Adicionado 1 a 15 temos 16 endereços no bloco.

Estácio

Endereços de Rede

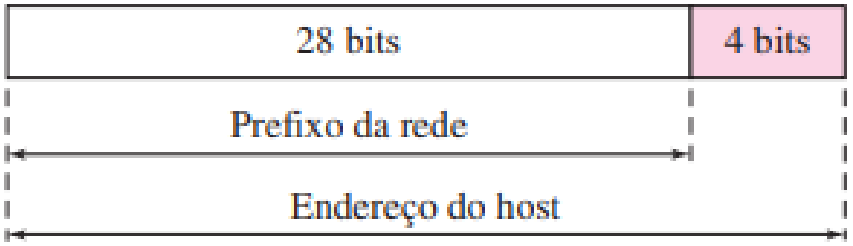
- Quando uma organização recebe um bloco de endereços IP, o primeiro deles é o endereço da rede (comumente, mas nem sempre).
- O endereço da rede é o que estabelece a organização para o restante do mundo.
- Exemplo: configuração de rede para o bloco 205.16.37.32/28



- A rede da organização é conectada à Internet por meio de um roteador, que tem dois endereços: um pertence ao bloco concedido; os demais pertencem à rede que se encontra do outro lado do

Hierarquia

- Os endereços IP apresentam níveis de hierarquia.
- **Hierarquia de Dois Níveis** (sem o uso de sub-redes)
  - Os *n* bits mais à esquerda do endereço x.y.z.t/n designam a rede (**prefixo**).
  - Os 32 − *n* bits mais à direita estabelecem o host particular (**sufixo**).



Hierarquia

- **Hierarquia de Três Níveis** (uso de sub-redes)
  - Quando alguma organização recebe uma grande quantidade de endereços, é possível que ela queira dividi-los em blocos menores (ou *clusters*), as chamadas sub-redes.
  - A Internet enxerga apenas uma rede, porém internamente podem existir várias.
  - A mensagem chega ao roteador da rede, o qual envia a mensagem para o roteador da sub-rede.

Estácio

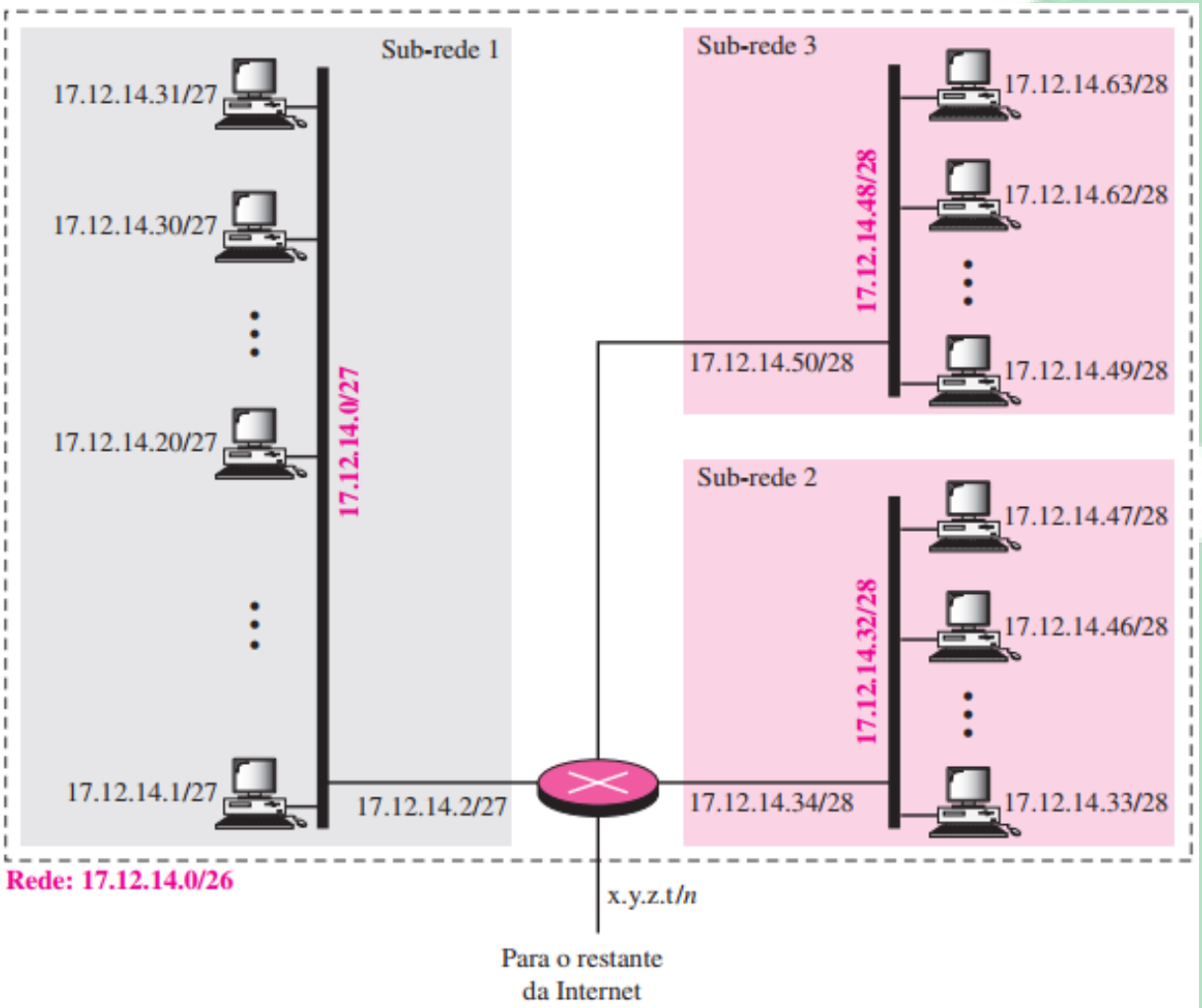
Hierarquia

- **Hierarquia de Três Níveis** (uso de sub-redes)

- Exemplo: Supondo que uma empresa tenha recebido o bloco 17.12.40.0/26 com 64 endereços.
  A empresa tem 3 escritórios e decide dividir os endereços da seguinte forma: 32, 16, 16. As novas máscaras podem ser descobertas da seguinte forma:
  A primeira sub-rede terá 32 endereços então sua máscara n1 tem de ser igual a 27 ($32 - 27 = 5 \therefore 2^5 = 32$).
  Como as outras duas sub-redes terão 16 endereços, então n2 = n3 = 28.
  Portanto temos as máscaras 27, 28 e 28 com a máscara da empresa sendo 26.

Estácio

Hierarquia


Sub-rede 1
17.12.14.31/27
17.12.14.30/27
17.12.14.20/27
17.12.14.1/27
17.12.14.0/27
17.12.14.2/27
Sub-rede 3
17.12.14.63/28
17.12.14.62/28
17.12.14.49/28
17.12.14.48/28
17.12.14.50/28
Sub-rede 2
17.12.14.47/28
17.12.14.46/28
17.12.14.33/28
17.12.14.32/28
17.12.14.34/28
Rede: 17.12.14.0/26
x.y.z.t/n
Para o restante
da Internet

Hierarquia

- **Hierarquia de Três Níveis** (uso de sub-redes)
  - Descobrindo os endereços **de** sub-rede a partir de um dos endereços **na** sub-rede.
    Na sub-rede 1, o endereço 17.12.14.29/27 nos fornece o endereço de sub-rede quando aplicamos a máscara /27

17.12.14.29/27

Host:	00010001 00001100 00001110 000**11101**
Máscara:	/27
Sub-rede:	00010001 00001100 00001110 00000000 $\rightarrow$ 17.12.14.0

Na sub-rede 2, o endereço 17.12.14.45/28 nos fornece o endereço de sub-rede quando aplicamos a máscara /28

17.12.14.45/28

Host:	00010001 00001100 00001110 0010**1101**
Máscara:	/28
Sub-rede:	00010001 00001100 00001110 00100000 $\rightarrow$ 17.12.14.32

Na sub-rede 3, o endereço 17.12.14.50/28 nos fornece o endereço de sub-rede quando aplicamos a máscara /28

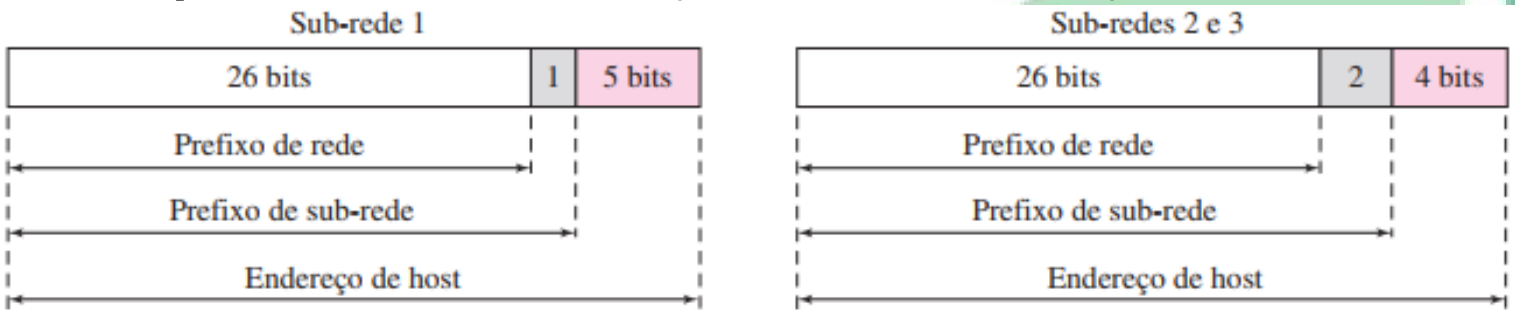
17.12.14.50/28

Host:	00010001 00001100 00001110 0011**0010**
Máscara:	/28
Sub-rede:	00010001 00001100 00001110 00110000 $\rightarrow$ 17.12.14.48

Estácio

Hierarquia

- **Hierarquia de Três Níveis** (uso de sub-redes)



- A estrutura de endereçamento sem classes não restringe o número de níveis hierárquicos.
- Uma organização pode dividir o bloco de endereços concedido em sub-blocos.
- Cada sub-bloco pode, por seu lado, ser dividido em sub-blocos menores, e assim por diante.

Estácio

Alocação de Endereços

- ICANN (*Internet Corporation for Assigned Names and Numbers*) ou *Corporação da Internet para Nomes e Números Designados* é a responsável por alocar os endereços.
- Normalmente a ICANN concede grandes blocos de endeços a ISPs, as quais fornecem seus serviços às empresas e usuários finais.

Estácio

Alocação de Endereços

- Exemplo: Um ISP recebe um bloco de endereços iniciando em 190.100.0.0/16 (65.536 endereços). Esse ISP precisa distribuir esses endereços para três grupos de clientes, como segue:
  a) O primeiro grupo apresenta 64 clientes; cada um deles precisa de 256 endereços.
  b) O segundo grupo tem 128 clientes; cada um deles precisa de 128 endereços.
  c) O terceiro grupo contém 128 clientes; cada um deles precisa de 64 endereços.

Estácio

Alocação de Endereços