1 Layering – Arquiteturas de Protocolos ..... 3
1.1 Modelo RM-OSI da ISO (Reference Model for Open Systems Interconnection, da International Standardization Organization) ..... 3
1.2 Pilhas: Uma arquitetura de Implementação ..... 5
1.3 Controle de fluxo ..... 7
2 Ligação Inter-redes ..... 9
2.1 Protocolos para ligação inter-redes ..... 10
3 IP – Endereçamento Inter-rede ..... 11
3.1 Endereçamento – conceitos básicos ..... 11
3.2 A divisão do espaço de endereçamento ..... 13
3.2.1 Um exemplo de endereçamento ..... 13
3.2.2 Máscaras de endereçamento ..... 14
3.3 Endereços especiais ..... 14
3.4 Endereços de roteadores ..... 15
3.5 NAT, Subredes e CIDR ..... 15
3.5.1 NAT - Network Address Translation ..... 15
3.5.2 Subredes ..... 17
3.5.3 CIDR – Classless InterDomain Routing ..... 19
4 Binding de endereços ..... 20
4.1 Resolução de endereços ..... 20
4.2 Técnicas de resolução de endereços ..... 21
4.2.1 Pesquisa em Tabela ..... 21
4.2.2 Forma Fechada ..... 22
4.2.3 Troca de mensagens ..... 22
4.3 Protocolo ARP – Address Resolution Protocol ..... 22
4.3.1 Formato de mensagem ARP ..... 23
5 Datagrama IP e Encaminhamento de Datagramas ..... 23
5.1 Serviço sem Conexão ..... 23
5.2 Datagrama IP ..... 24
5.2.1 Encaminhamento de Datagramas ..... 24
5.2.2 O Campo Máscara e Encaminhamento de Datagramas ..... 24
5.2.3 Best-Effort Delivery ..... 25
5.2.4 Formato do Cabeçalho IP ..... 25
6 Fragmentação e Remontagem de Datagramas IP ..... 26
6.1 MTU – Maximum Transmission Unit ..... 27
6.2 Remontagem ..... 27
7 IPv6 ..... 28
8 ICMP – Internet Control Message Protocol ..... 30
9 TCP: Transporte Confiável ..... 32
9.1 Serviço fim-a-fim ..... 33
9.2 Perda de Pacotes e Retransmissão ..... 33
9.3 Controle de Fluxo ..... 34
9.4 Three-way Handshake ..... 35
9.5 Formato do TCP ..... 36
9.6 UDP – User Datagram Protocol ..... 36
9.7 Encapsulamento de transporte ..... 37
10 Camada de Aplicação – Alguns exemplos importantes ..... 37

10.1 DNS – Domain Name System ..... 37
10.2 Correio eletrônico ..... 39

**Apostila de Redes de Computadores II**
**Professor Frederico Sauer, M.Sc.**
**Livro-texto – Redes de Computadores e Internets – 2ª edição – Douglas Comer**

**Atenção** – este texto é apenas uma referência básica para o curso de Redes de Computadores II dos cursos de graduação em Tecnologia em Processamento de Dados e Bacharelado em Sistemas de Computação. É totalmente baseado no livro texto e não pretende esgotar o assunto, sendo necessária pesquisa complementar no próprio livro texto ou em outras referências de Redes de Computadores, de acordo com o assunto. Uma bibliografia recomendada é apresentada ao final deste documento.

# 1 Layering – Arquiteturas de Protocolos

## *1.1 Modelo RM-OSI da ISO (Reference Model for Open Systems Interconnection, da International Standardization Organization)*

Com a dificuldade existente para possibilitar a maior vantagem da utilização das Redes de Computadores, que é a interoperação, os usuários ansiavam por uma solução que os permitisse interligar seus equipamentos, mesmo que fossem de diferentes fabricantes. Naquela época, equipamentos de um mesmo fabricante, muitas vezes, não "se falavam". A IBM propôs a sua solução, chamada de modelo SNA (Systems Network Architecture), e a Digital lançou a DNA (Digital Network Architecture). Apesar de permitirem a troca de informações entre equipamentos diferentes, os mesmos precisariam ser todos do mesmo fabricante (IBM com IBM, Digital com Digital). Apesar de já ser um grande avanço, ainda não era o ideal. Pela fidelização forçada, mas com a vantagem de interoperação, os equipamentos ficaram mais caros ainda.

A ISO então lançou uma proposta inovadora. Definiu um modelo onde apenas duas entidades precisavam ser detalhadamente descritas, porém, sem vínculo algum com fabricantes ou peculiaridades de hardware que dificultassem a interoperação. Estas entidades são os protocolos de comunicação e as interfaces. Os protocolos definem um conjunto de regras comunicação e formatos de mensagens usados para uma função de interoperação qualquer. As interfaces são os pontos de comunicação entre as entidades que estão se comunicando, que chamamos de "entidades pares". Como fruto deste trabalho, surgiu o RM-OSI (figura 1).

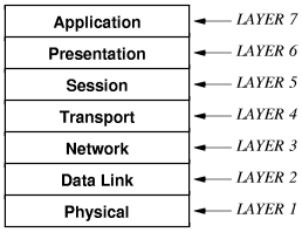


**Figura 1 - RM-OSI**

Neste modelo, cada camada possui um conjunto de funcionalidades, que são descritas apenas da forma "o que fazer", e não "como fazer". As funcionalidades são agrupadas, de acordo com certas afinidades, dentro de uma das sete camadas do modelo, de forma que uma camada use serviços providos pelas camadas inferiores a ela e forneça serviços à camada imediatamente acima. Desta forma, ao receber uma determinada unidade de dados, a camada pressupõe que todas as funcionalidades das camadas abaixo foram devidamente executadas, bastando apenas que a camada em execução exerça apenas as funções solicitadas pela aplicação em execução. Algumas funções básicas de cada camada:

- Camada 1 – Física – exclusivamente criação e detecção de bits. Padrões universais como a RS-232 (interface serial) é um exemplo;
- Camada 2 – Enlace – os bits formam quadros (ou *frames*, em inglês). Devido a isto, tem a capacidade de manipular endereços de interfaces. Pode detectar erros também;
- Camada 3 – Rede – os dados desta camada são representados numa unidade chamada pacote, que será encapsulada, como veremos adiante, dentro do quadro. Roteamento e endereçamento são algumas das suas funções;
- Camada 4 – Transporte – Esta camada lida com a transferência fim-a-fim dos dados. A sua unidade de dados é a mensagem, ou, de acordo com o seu tamanho, o segmento de mensagem;
- Camada 5 – Sessão – a Camada de sessão permite a criação de sessões entre equipamentos remotos. Protocolos de autenticação, por exemplo, fazem parte desta camada.
- Camada 6 – Apresentação – Fornece uma interface para representação de dados, de forma a permitir a interoperação entre equipamentos diferentes, sistemas diferentes e até mesmo idiomas diferentes. Outras funcionalidades são a criptografia e a compactação dos dados;
- Camada 7 – Aplicação – No modelo OSI, serve apenas para a interação com o usuário local.

## 1.2 Pilhas: Uma arquitetura de Implementação

Há uma grande confusão quando se diz que um computador usa "o protocolo TCP/IP". Na verdade, o que estamos querendo dizer é que o computador implementa a pilha de protocolos da família TCP/IP, da qual fazem parte não só os protocolos TCP (tranmission Control Protocol) e o IP (Internet Protocol), mas inúmeros outros. O DNS, http, UDP, ICMP, ARP e tantos outros também fazem parte dessa família.
As pilhas são assim chamadas porque, na abstração dos desenvolvedores, elas se organizam em camadas sobrepostas (empilhadas), onde um protoclo se relaciona com o outro segundo uma determinada hierarquia. Observe na figura 2 como se processa a comunicação de um aplicativo no computador 1 com o seu correspondente no computador 2 (um cliente e um servidor ftp, por exemplo).

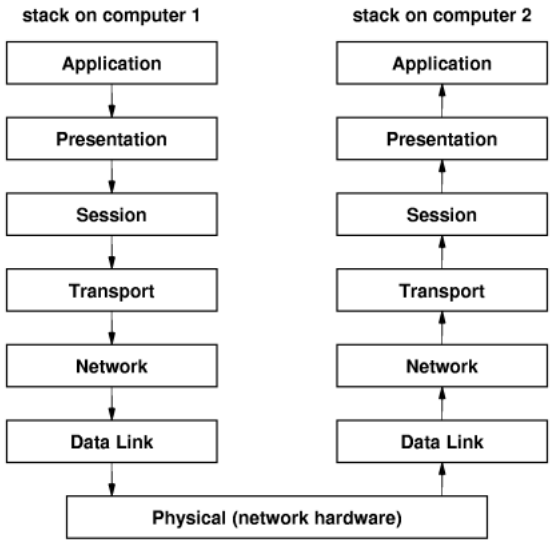


**Figura 2 - Unidade de dados passando pelas camadas**

Para que as unidades de dados de cada uma destas camadas possa identificar corretamente o que deve processar, ela deve, ao receber a unidade, ser capaz de ler o cabeçalho da mesma e executar os procedimentos descritos. Por isso, a cada camada por onde os dados passam antes de serem despachados para os seus destinos, um cabaçalho é agregado mais externamente, e a sua "carga útil" é exatamente aquilo que ela recebeu da camada anterior. Chamamos isso de aninhamento (*nesting*) de cabeçalhos, ou mais comumente, encapsulamento. A figura 3 mostra como a unidade de dados fica antes de ser despachada para o meio físico, no modelo OSI.

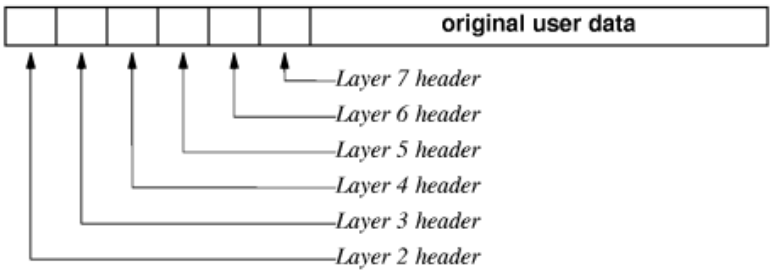


**Figura 3 - Aninhamento (encapsulamento) de cabeçalhos**

Outra abstração interessante do projeto de protocolos é que cada camada gera uma unidade de dados que deverá ser rigorosamente igual à unidade recebida pela camada correspondente no computador de destino (na verdade, num ambiente mais amplo, isso ocorre apenas à partir da camada de transporte, e antes disso a correspondência é nó-a-nó)
A figura 4 apresenta essa correspondência entre as unidades, e a figura 5 a ilustração dessa correspondência num sistema de *internetworking*.

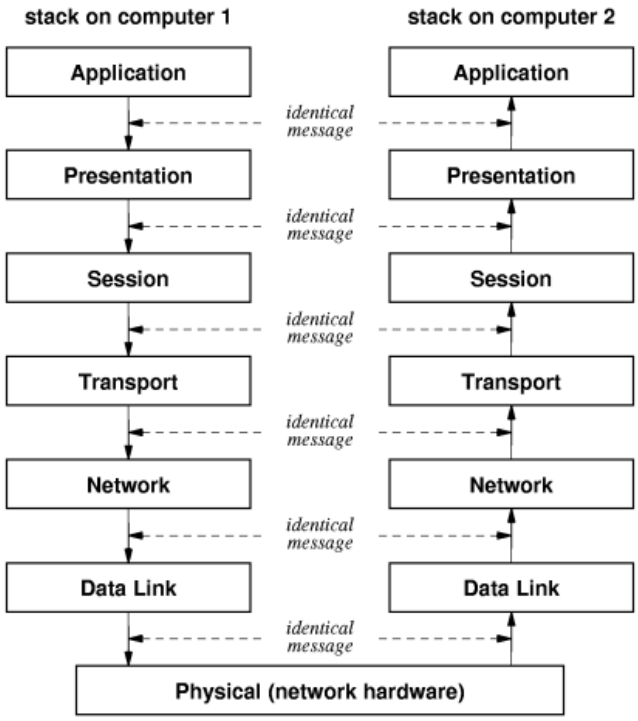


**Figura 4 - Dados idênticos no mesmo nível**

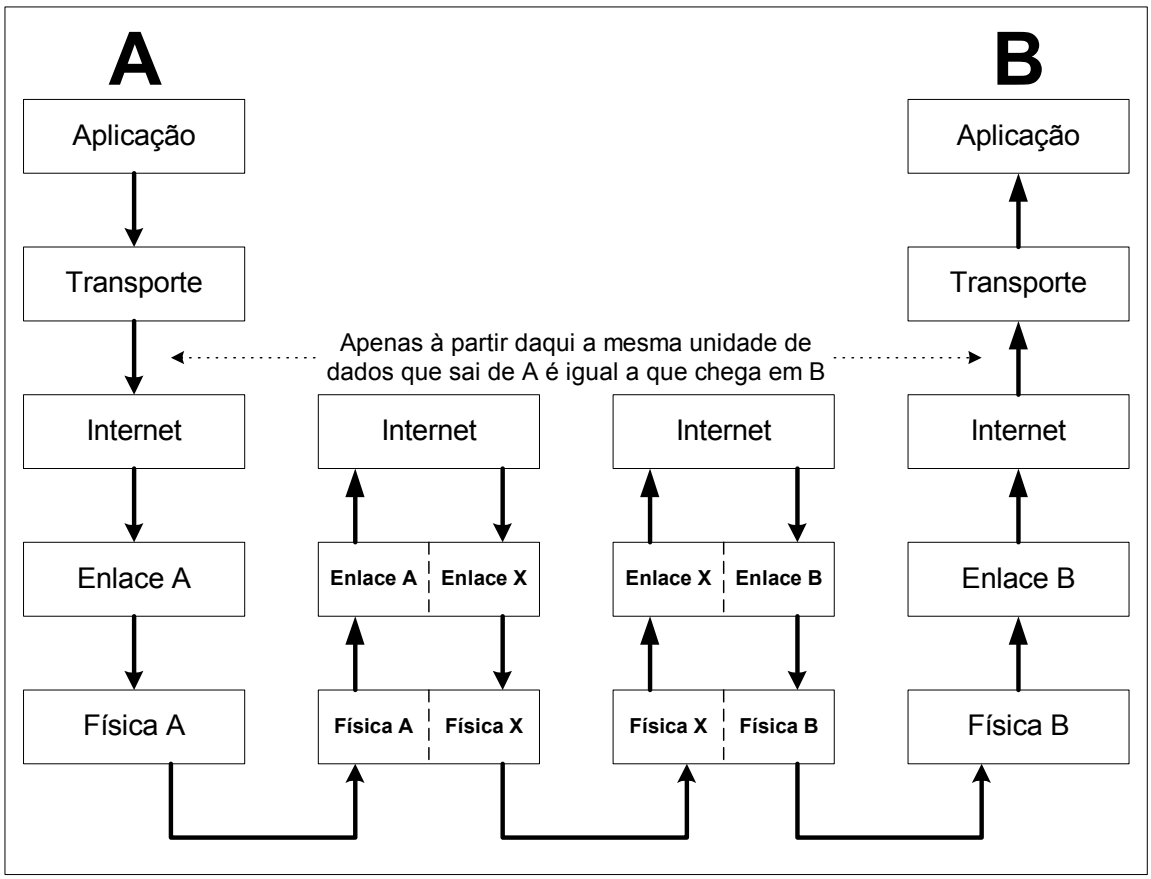


**Figura 5 – Unidades de Dados em ambiente de *Internetworking***

Na figura 5, que representa o ambiente real, o que observamos é que, devido às características das redes pelas quais os dados passam até chegar ao destino, alterações serão feitas. Apenas à titilo de exemplo, pode-se citar:

- ✓ Na camada física, os bits são formatados de acordo com o meio físico e as interfaces. Supondo que os enlaces entre A e o primeiro roteador seja de fibra, entre o primeiro roteador e o segundo de cobre, e entre o segundo roteador e o nó B *wireless*. Em cada um destes segmentos os bits serão codificados diferentemente;
- ✓ Na camada de enlace, como cada interface deve possuir um endereço físico único, os cabeçalhos serão diferentes; e
- ✓ Na camada de rede, o campo TTL (*Time to Live*) será decrementado em cada roteador pelo qual o pacote passar.

## *1.3 Controle de fluxo*

Os computadores não operam na mesma velocidade. Para que a interoperação seja feita sem atropelos, é vital que haja um controle de fluxo. Uma das técnicas mais usadas é o protocolo *sliding windows*. (figura 6).

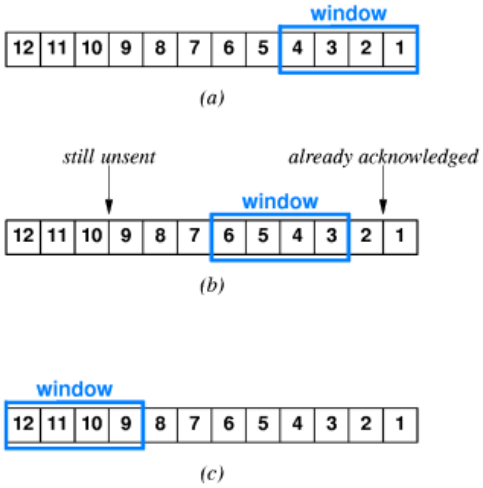


**Figura 6 - *Sliding Windows***

Nele, Um tamanho de janela é definido entre emissor e receptor. O emissor apenas envia os dados contidos na janela de emissão, e fica aguardando do receptor a confirmação do seu recebimento. Apenas após a reordenação dos mesmos na janela de recepção, é que o receptor envia uma confirmação para os dados (*ack*), deslizando sua janela para aguardar novos dados. O emissor então, recebendo os acks, pode deslizar sua janela e continuar a transmitir dados. Esse tamanho de janela pode ser configurado dinamicamente, para permitir a adequação da taxa de transmissão (*throughput*) às reais capacidades de emissor e receptor, num determinado momento. A figura 7 mostra configurações diferentes do protocolo para envio de *acks*.

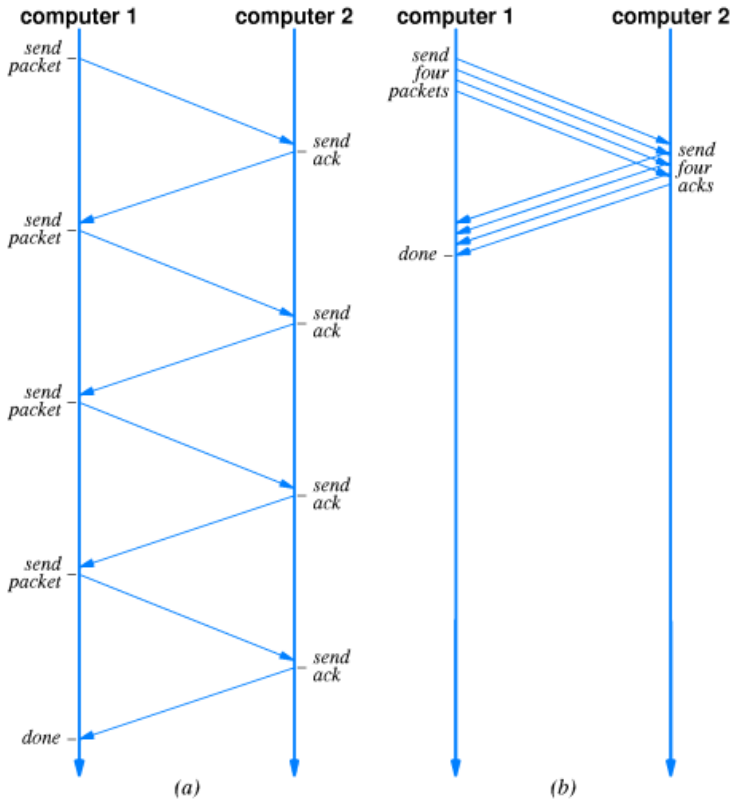


**Figura 7 - diferentes configurações para *ACK***

# 2 Ligação Inter-redes

Atualmente há uma grande confusão sobre o que seria a Internet. Muitos dizem “estou na Internet”, como se o usuário fizesse parte dessa rede. Na verdade, a Internet é uma grande rede mundial de equipamentos de *Internetworking*, ou seja, apenas para interconectar redes (criar o ambiente inter-redes). Quando você está “conectado” ao IG, por exemplo, você recebe um endereço de rede pertencente ao IG. Você está **na rede** do IG. Esta rede, por sua vez, possui um *gateway* para algum dos *backbones* da Internet, de forma que você possa navegar por outras redes (como a do google, do yahoo ou qualquer outra). A Internet é apenas a “nuvem” de equipamentos que usamos para poder acessar outras redes locais. Esta infraestrutura é ilustrada na figura 8.

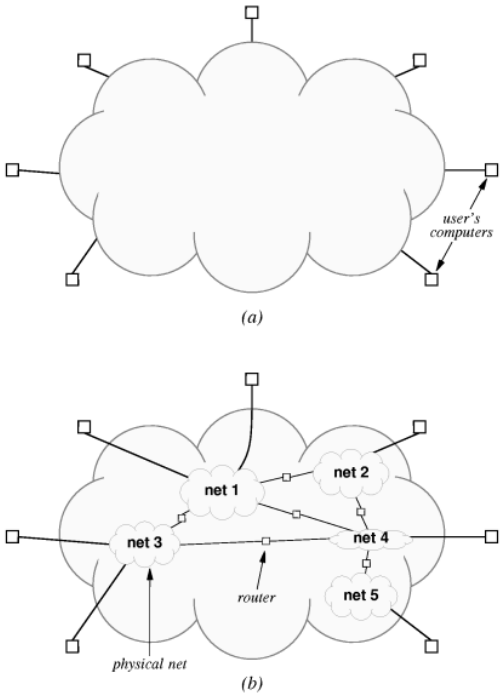


**Figura 8 - Ambiente de *Internetworking***

## 2.1 Protocolos para ligação inter-redes

Existem várias pilhas (ou suítes) de protocolos capazes de permitir a criação deste ambiente de *internetworking*, como o SNA e o IPX/SPX, mas, indiscutivelmente, a mais usada no mundo é a TCP/IP. Essa pilha surgiu através de uma pesquisa fomentada pelo departamento de defesa Americano (DoD), que queria criar um ambiente capaz de permanecer operando mesmo quando um de seus nós fosse eliminado, num possível (e naquela época – anos 70 – bastante provável, ataque nuclear). Esse trabalho, conduzido inicialmente pela ARPA (*Advanced Research Projects Agency*) e, posteriormente, continuado pela NSF (*National Science Foundation*) culminou com a criação das redes locais (*lans*) e o TCP/IP. A idéia do TCP/IP era basicamente criar o ambiente de *internetworking*, já proposto no modelo OSI, porém com mais simplicidade, objetividade, e menor rigor hierárquico entre as camadas. Das conclusões destes estudos, verificou-se que as funcionalidades das camadas de aplicação, apresentação e sessão poderiam ficar todas numa única camada, a própria camada de aplicação. A camada de transporte, que no modelo OSI previa apenas o serviço com conexões e 5 tipos de serviços diferentes, de acordo com o meio físico, passou a ter um serviço sem conexões (protocolo UDP – *User Datagram Protocol*) e um serviço *orientado* a conexões (protocolo TCP), onde são feitos controles sobre os dados enviados e recebidos apenas quando de sua recepção, se abstraindo do meio de transmissão (a "nuvem" entre os pares comunicantes). Assim segmentos de uma mensagem podem ser roteados por caminhos completamente diferentes, ficando a cargo do receptor a sua reordenação antes da verificação de erros. A figura 9 mostra a divisão em camadas do

modelo TCP/IP. Na prática, no entanto, esse modelo não é rigorosamente implementado em camadas compartimentadas, uma vez que o que se busca é o melhor desempenho.

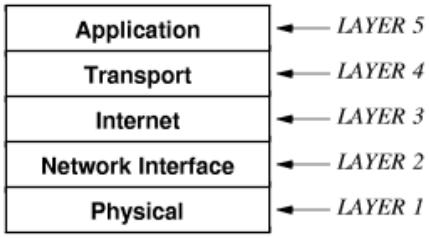


**Figura 9 - Suíte TCP/IP**

# 3 IP – Endereçamento Inter-rede

Neste tópico serão abordadas as técnicas usadas para que possa haver comunicação entre os milhões de usuários da Internet.

## *3.1 Endereçamento – conceitos básicos*

A principal funcionalidade do IP (*Internet Protocol*) é o endereçamento. É necessário assegurar que cada nó capaz de se comunicar com outro através de um ambiente de *Internetworking* possua um endereço unívoco. É obvio que seria impossível que uma única entidade controlasse todos os endereços de máquinas no mundo, então, foi idealizado um esquema onde o endereço é dividido em duas partes: um prefixo, que identifica a rede a qual um nó pertence, e um sufixo, que identifica o próprio *host*. O prefixo é determinado por uma autoridade local de distribuição de endereços, e o sufixo fica a critério do administrador da rede local. Observe que essa estratégia se coaduna com a idéia explamada no capítulo anterior, onde o ambiente de *internetworking* se presta apenas a rotear dados entre redes, e não entre máquinas.
O IP atualmente usado, versão 4, usa endereços de 32 bits (quatro *bytes*). Esses bytes são divididos em prefixo e sufixo de acordo com a classe a qual pertencem. As classes, por sua vez, definem quantas redes daquela respectiva classe poderão ser criadas no mundo e quantas máquinas cada uma destas redes poderá ter. Para que os roteadores possam identificar a qual classe um endereço pertence (e assim, separar apenas os bytes do prefixo para rotear para a rede de destino), são usados os primeiros quatro bits do endereço. A figura 10 ilustra as classes, de acordo com os primeiros bits.

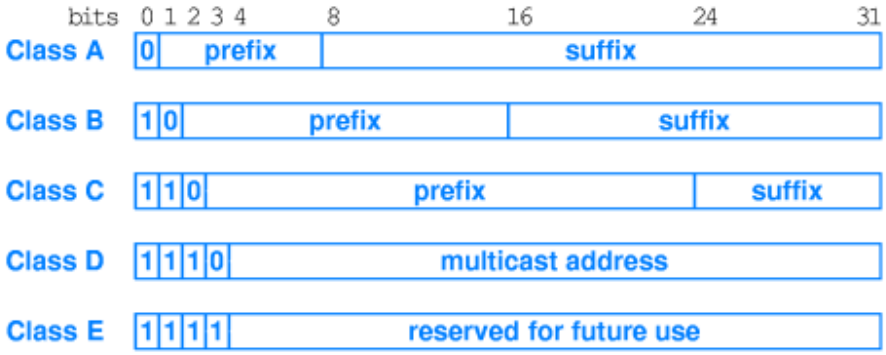


**Figura 10 - Classes de acordo com os 4 primeiros bits**

A figura 11 mostra todas as variações possíveis de classes de acordo com os primeiros 4 bits.

| First Four Bits Of Address | Table Index (in decimal) | Class of Address |
|---|---|---|
| 0000 | 0 | A |
| 0001 | 1 | A |
| 0010 | 2 | A |
| 0011 | 3 | A |
| 0100 | 4 | A |
| 0101 | 5 | A |
| 0110 | 6 | A |
| 0111 | 7 | A |
| 1000 | 8 | B |
| 1001 | 9 | B |
| 1010 | 10 | B |
| 1011 | 11 | B |
| 1100 | 12 | C |
| 1101 | 13 | C |
| 1110 | 14 | D |
| 1111 | 15 | E |

**Figura 11 - Variações de endereços e suas classes**

È claro que os computadores só lidam com bits. Mas nós, seres humanos, não teríamos facilidade em trabalhar com bits. Foi adotada, então, a dotação decimal pontilhada (*Dotted Decimal Notation*), onde os binários são convertidos para decimais para se tornarem mais

fáceis de serem utilizados, principalmente em configurações. A figura 12 apresenta alguns exemplos de endereços binários convertidos para decimal.

| 32-bit Binary Number | Equivalent Dotted Decimal |
|---|---|
| 10000001 00110100 00000110 00000000 | 129 . 52 . 6 . 0 |
| 11000000 00000101 00110000 00000011 | 192 . 5 . 48 . 3 |
| 00001010 00000010 00000000 00100101 | 10 . 2 . 0 . 37 |
| 10000000 00001010 00000010 00000011 | 128 . 10 . 2 . 3 |
| 10000000 10000000 11111111 00000000 | 128 . 128 . 255 . 0 |

**Figura 12 - Exemplos de endereços convertidos para decimal**

## *3.2 A divisão do espaço de endereçamento*

O endereçamento por classe possui algumas desvantagens. A primeira delas é que existe uma grande concentração de endereços disponíveis para as classes A e B, pouco usadas devido ao grande número de máquinas que cada uma suporta. Em decorrência a isso, os endereços disponíveis estão acabando. A figura 12 mostra a quantidade de redes e máquinas por rede que cada classe suporta.

| Address Class | Bits In Prefix | Maximum Number of Networks | Bits In Suffix | Maximum Number Of Hosts Per Network |
|---|---|---|---|---|
| A | 7 | 128 | 24 | 16777216 |
| B | 14 | 16384 | 16 | 65536 |
| C | 21 | 2097152 | 8 | 256 |

**Figura 13 - Quantidade de redes e máquinas por classe**

### 3.2.1 Um exemplo de endereçamento

Suponha que uma determinada máquina de uma rede possui o endereço 146.164.69.2.
a) A qual classe ele pertence ?
b) Um pacote destinado a esta máquina, será roteado para qual rede ?

Respostas: convertendo para binário, temos:
146.164.69.2 → 10010010.10100100.01000101.00000010
Logo, como começa com 10, é classe B. se é classe B, os dois primeiros octetos representam a rede e os demais para a máquina. Como o roteamento é feita para a rede de destino, então ele será roteado para a rede 146.164.0.0

### 3.2.2 Máscaras de endereçamento

Para que o cálculo seja feito rapidamente, os roteadores recebem o endereço de destino, deslocam de quatro bits para a direita e verificam a classe. Conhecida a classe, fazem então a operação lógica AND (•) entre o endereço e uma máscara de endereçamento, que é correspondente à classe, de acordo com a tabela 1:

| Classe | Máscara |
|---|---|
| A | 255.0.0.0 |
| B | 255.255.0.0 |
| C | 255.255.255.0 |

**Tabela 1- Máscaras de acordo com a classe**

Como sabemos, o número decimal 255 é correspondente a 11111111 em binário. Logo, a operação AND entre 146.164.62.2 e 255.255.0.0 vai resultar em:

10010010.10100100.01000101.00000010
• 11111111.11111111.00000000.00000000
10010010.10100100.00000000.00000000 → 146.164.0.0

## *3.3 Endereços especiais*

Há alguns endereços que são reservados pelo IP para funções especiais. Estes endereços não podem ser atribuídos a máquinas individuais. A figura 13 apresenta estes endereços.

| Prefix | Suffix | Type Of Address | Purpose |
|---|---|---|---|
| all-0s | all-0s | this computer | used during bootstrap |
| network | all-0s | network | identifies a network |
| network | all-1s | directed broadcast | broadcast on specified net |
| all-1s | all-1s | limited broadcast | broadcast on local net |
| 127 | any | loopback | testing |

**Figura 14 - Endereços especiais**

Entre estes, há dois tipos de endereços reservados importantíssimos. Dentro de cada faixa de endereços, em binário, os bits do sufixo tudo 0 (zero) identifica a rede, logo, não podem ser usados para serem configurados em máquinas. O mesmo ocorre com bits tudo 1 (um) no sufixo, que identifica o broadcast de uma rede específica, também não podendo ser usado para identificar máquinas. Isso implica, na prática, que numa classe C, por exemplo, das 256 combinações possíveis de bits do sufixo ($2^8$), apenas 254 máquinas poderão coexistir em cada rede.

## 3.4 Endereços de roteadores

Apesar de, até agora, ter sido dito que o endereço IP é usado para identificar uma máquina específica, é necessário que cada interface da máquina com uma rede tenha o seu próprio endereço. Isso significa que, se uma máquina possuir duas placas de rede, cada uma deverá ter seu próprio endereço. Isso ocorre com roteadores. Fazendo o papel de *gateways*, eles servem para interconectar, no mínimo, duas redes distintas. Conseqüentemente, cada interface sua deverá possuir um endereço da rede correspondente, conforme o exemplo da figura 14.

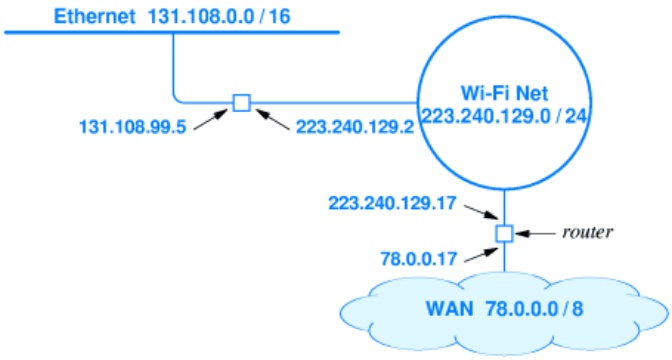


**Figura 15 - Endereços de interfaces de roteadores**

## 3.5 NAT, Subredes e CIDR

Com pouco tempo de utilização da Internet, percebeu-se que havia algumas inadequações às reais necessidades dos usuários. Com o crescimento explosivo, as redes passaram a ficar congestionadas. Além disso, o espaço de endereçamento não era suficiente para a demanda, e os problemas de segurança ficaram mais visíveis. Para minimizar estes problemas, foram criadas soluções como o NAT, o CIDR e as subredes, que serão abordadas a seguir.

### 3.5.1 NAT - Network Address Translation

Como o espaço de endereçamento começou a ficar escasso, surgiram alternativas para otimizar o esquema de endereçamento. Uma delas é o NAT, que além de permitir uma redução quantitativa do número de endereços válidos necessários para uma rede, agrega segurança por obrigar que todos os pacotes passem por um dispositivo lógico chamado de caixa-NAT, que pode implementar políticas de *firewall*. Para isso, a RFC 1918 separou os seguintes endereços (tabela 2):

| Faixas | Primeiro endereço | Último endereço |
|---|---|---|
| 1 | 10.0.0.0/8 | 10.255.255.255/8 |
| 2 | 172.16.0.0/12 | 172.31.255.255/12 |

| | | |
|---|---|---|
| 3 | 192.168.0.0/16 | 192.168.255.255/16 |

**Tabela 2 - endereços reservados - RFC 1918**

Obs.: essa representação "/n" significa o número de bits 1 usados na máscara. A faixa 2, por exemplo, usa a máscara 11111111.11110000.00000000.00000000 (255.240.0.0), que é suficiente para representar todas as redes reservadas para o uso do NAT na classe B. Ë muito comum a criação de redes locais com endereços reservados para compartilhamento de Velox, por exemplo, já implementada no próprio modem, que se transforma num modem-router para executar essa função. Ele recebe um único endereço da telemar, e, através do DHCP (*Dynamic Host Configuration Protocol*), que será visto capítulos adiante, aloca endereços como 192.168.0.1, 192.168.0.2, 192.168.0.3, etc. para cada máquina da rede. A figura 16 mostra a caixa NAT.

**Figura 16 - NAT compartilhando Velox**

Para que o esquema funcione, é necessário que seja feita uma "troca" dos endereços reservados, que não podem ser usados no ambiente de *internetworking*, pelo endereço válido, fornecido pelo provedor. Neste momento, são armazenadas na caixa NAT detalhes de cada conexão, como porta de origem e destino, além dos endereços de origem e destino, para que, na resposta externa, a caixa NAT saiba para quem direcioná-la, trocando o endereço de destino (que é o da sua interface externa) pelo endereço reservado da máquina requisitante. A figura 17 mostra esta tradução e a figura 18 apresenta detalhes das conexões, usados para saber de quem é a resposta recebida.

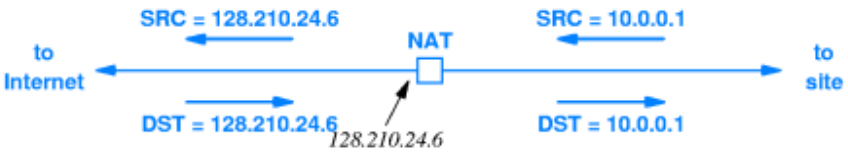


**Figura 17 - Tradução de endereços pela caixa NAT**

| Direction | Fields | Old Value | New Value |
|---|---|---|---|
| out | IP SRC:TCP SRC | 10.0.0.1:30000 | 128.10.19.20:40001 |
| out | IP SRC:TCP SRC | 10.0.0.2:30000 | 128.10.19.20:40002 |
| in | IP DEST:TCP DEST | 128.10.19.20:40001 | 10.0.0.1:30000 |
| in | IP DEST:TCP DEST | 128.10.19.20:40002 | 10.0.0.2:30000 |

**Figura 18 - Números de porta usados para entrega dos pacotes**

### 3.5.2 Subredes

Com o aumento do uso dos recursos computacionais pelas empresas, e o incremento do tráfego da rede com novas aplicações cliente/servidor e vários compartilhamentos de recursos, as redes começaram a ficar sobrecarregadas. Foi visto no curso de redes I que a substituição de *hubs* por *switches* já melhora bastante esta situação, pois, por implementar a camada 2 (enlace), permite a utilização de endereços físicos identificando as interfaces que estão conectadas à cada porta. Com isso, é reduzido o domínio de colisões, que passa a ser de todo um *hub* para uma única porta (a porta do *switch* não consegue identificar o endereço de destino dos bits recebidos – colisão – e descarta o quadro recebido). Já o *hub*, cujas portas apenas amplificam e retransmitem os dados recebidos a todas as outras portas, não conseguem evitar que colisões se propaguem. Mesmo para o switch, há, no entanto, um tipo de quadro que é retransmitido para todas as portas, que é o *broadcast*. Estes quadros são extensivamente utilizados pelos sistemas operacionais atuais, visando a ampla disseminação da informação sobre recursos compartilhados. Com isso, as redes tem se tornado congestionadas com pouco tempo de implementação.
A solução para reduzir este problema é a sublocação de endereços, com a criação de subredes autônomas, com o uso da máscara de subrede. Essa máscara é a mesma máscara de rede apresentada no item 3.2.2, porém, alguns bits que seriam usados para identificar hosts (bits do sufixo) são usados para identificar o número da subrede criada. A figura 19 ilustra uma situação típica de uso de subredes.

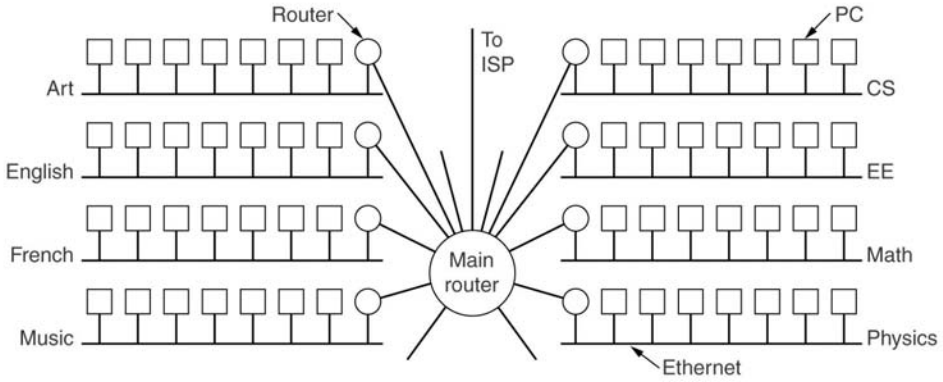

**Figura 19 - Situação típica de subredes**

Observe, como dito anteriormente, há um roteador que centraliza todas as conexões de rede local (o chamado backbone colapsado) que gerencia as subredes e converte os endereços antes de roteá-los para o ISP (*Internet Service Provider*).
Vamos exemplificar o esquema. Suponhamos que temos uma classe B inteira a nossa disposição (por exemplo, 130.50.0.0). Se utilizarmos uma máscara de subredes, digamos, do tipo 255.255.252.0 (ou /22, porque essa máscara tem 22 bits 1 consecutivos), no lugar da tradicional 255.255.0.0, estaremos possibilitando a criação de 64 subredes ($2^6$), correspondente aos seis bits retirados do sufixo para criação de subredes, Cada uma destas subredes pode ter até 1022 máquinas ($2^{10} - 2$). A máscara do exemplo está ilustrada na figura 20.

10nnnnnn.nnnnnnnn.ssssssshh.hhhhhhhh
n- network, s – subnet, h - host

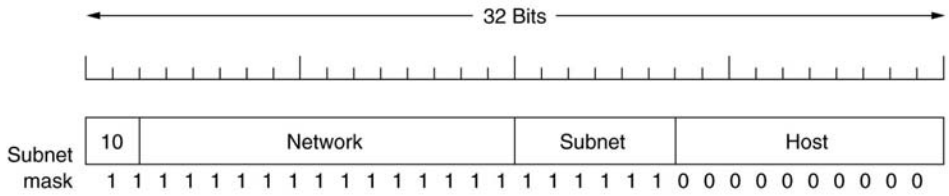


**Figura 20 - Máscara de subrede**

Em binário, teríamos as seguintes subredes:

Subrede 1: 10000010.00110010.000000 | 00.00000000 → 130.50.0.0
Subrede 2: 10000010.00110010.000001 | 00.00000000 → 130.50.4.0
Subrede 3: 10000010.00110010.000010 | 00.00000000 → 130.50.8.0
Subrede 4: 10000010.00110010.000011 | 00.00000000 → 130.50.12.0
...
Subrede 64: 10000010.00110010.111111 | 00.00000000 → 130.50.252.0

Estas subredes terão as seguintes características adicionais (tabela 3):

| Subrede | Primeiro endereço | Último endereço | Broadcast |
|---|---|---|---|
| 130.50.0.0 | 130.50.0.1 | 130.50.3.254 | 130.50.3.255 |
| 130.50.4.0 | 130.50.4.1 | 130.50.7.254 | 130.50.7.255 |
| 130.50.8.0 | 130.50.8.1 | 130.50.11.254 | 130.50.11.255 |
| 130.50.12.0 | 130.50.12.1 | 130.50.15.254 | 130.50.15.255 |
| 130.50.252.0 | 130.50.252.1 | 130.50.255.254 | 130.50.255.255 |

**Tabela 3 - Início e fim de cada subrede**

### 3.5.3 CIDR – Classless InterDomain Routing

O CIDR é uma proposta direcionada objetivamente para a eliminação das classes de endereços, fazendo com que a máscara passe a fazer parte definitivamente do endereço. Há diferenças fundamentais entre as subredes e o CIDR: As subredes não possibilitam uma redistribuição global de endereços, e sim apenas local. São visíveis apenas internamente à rede. O roteador externo converte estes endereços para as máscaras tradicionais, cuja classe foi concedida para utilização do domínio que foi dividido em subredes. Na prática, as subredes ainda gastam mais endereços, pois cada subrede terá o seu próprio endereço e o seu próprio broadcast, tirando endereços que poderiam estar sendo usados para identificar máquinas, se a rede não tivesse sido segmentada. O CIDR é uma solução criada para ser visível externamente ao domínio local, permitindo a um provedor de *backbone*, responsável pela distribuição de endereços, uma organização de forma a minimizar o desperdício atual. Uma rede pequena (50 máquinas, digamos) no esquema atual, usaria uma classe C, capaz de endereçar 254 máquinas, desperdiçando 204 endereços que poderiam estar sendo alocados para outras redes.
Elaboremos um exemplo para entender o CIDR. Suponha que um provedor possua disponível a faixa 194.24.0.0. por tratar-se de um classe B, da forma tradicional deveria ser alocado para uma única instituição de médio porte, como uma universidade (a UFRJ usa o classe B 146.164.0.0), com a capacidade de endereçar até $2^{16}$ – 2 máquinas (65.534 máquinas). O provedor verifica que deve atender, no complexo universitário, a três instituições com as respectivas necessidades:

- ✓ Cambridge – 2048 máquinas;
- ✓ Oxford – 4096 máquinas; e
- ✓ Edinburgh – 1024 endereços.

Após cálculos de adequabilidade, o provedor resolve implementar a distribuição de endereços, acompanhados de máscaras específicas, conforme a tabela 4.

| University | First address | Last address | How many | Written as |
|---|---|---|---|---|
| Cambridge | 194.24.0.0 | 194.24.7. | 2048 | 194.24.0.0/21 |
| Edinburgh | 194.24.8.0 | 194.24.11.255 | 1024 | 194.24.8.0/22 |
| (Available) | 194.24.12.0 | 194.24.15.255 | 1024 | 194.24.12/22 |
| Oxford | 194.24.16.0 | 194.24.31.255 | 4096 | 194.24.16.0/20 |

**Tabela 4 - CIDR do classe B 194.24.0.0**

A primeira instituição a ser atendida foi Cambridge, com a máscara 255.255.248.0 (/21). Com ela, conseguimos chegar aos 2048 endereços necessários, alocando um bloco contíguo, iniciando no endereço 194.24.0.0 e terminando no 194.24.7.255. Não podemos esquecer que o endereço 194.24.0.0 identifica a rede e o 194.24.7.255 o broadcast deste bloco. Uma vez alocado este bloco, parte-se para as próximas necessidades. Com a máscara /22, a partir do primeiro endereço disponível (194.24.8.0), chega-se a um bloco de 1024 máquinas. Pode-se então atender à necessidade de Edinburgh, iniciando no primeiro endereço disponível (194.24.8.0) e terminando no 194.24.11.254 (o 255 é broadcast). A

partir do 194.24.12.0, não é possível um único bloco de 4096 endereços, a partir de onde paramos. Reserva-se então a faixa de 1024 endereços que vai de 194.24.12.0 até 194.24.15.255, com a máscara /22, e então se aloca a faixa imediatamente seguinte com a máscara /20 para a Universidade de Oxford, ficando o primeiro endereço 194.24.16.0 (endereço de rede), e o último endereço (broadcast) 194.24.31.255.
Ficam óbvias duas desvantagens deste modelo: primeiro, a complexidade de se ajustar a necessidade das redes a um esquema amarrado, que disponibiliza conjuntos de endereços dependendo do que já foi alocado.Outra questão é que as tabelas de roteamento do mundo inteiro deverão ser alteradas para incluir essa nova máscara junto com o endereço. Na prática, entretanto, esse esquema já vem sendo utilizado por provedores para reduzir o desperdício de endereços.

# 4 Binding de endereços

Conforme aprendemos em Redes I, todas as comunicações em redes locais ocorrem nos níveis físico ou, no máximo de enlace. As ligações entre equipamentos de conectividade, apesar de utilizarem endereços lógicos (IP), precisam ter seus endereços convertidos para o correspondente endereço físico, para que os dados sejam transportados para a interface adequada.
No ambiente de inter-rede, apesar do endereçamento entre máquinas ser lógico[1], baseado em endereços IP, na prática, as comunicações são feitas ao nível de enlace. Neste nível, cuja unidade de dados é o quadro[2], o endereçamento é físico, e não lógico. Assim, é necessária uma estratégia para resolver (traduzir) endereços lógicos em físicos, e com isso "amarrar" (*binding*) estes pares de endereços para que a comunicação possa acontecer.

## *4.1 Resolução de endereços*

Resilver um endereço é sempre uma tarefa local. As comunicações fim-a-fim ocorrem, na verdade, nó-a-nó, até que os dados cheguem ao seu destino (fim). Na figura 21, por exemplo, se o computador A deseja enviar algo para F (ou seja, a comunicação fim-a-fim é sempre entre A e F), na verdade, ocorre da seguinte forma:

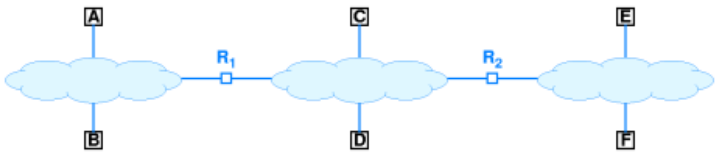


**Figura 21 - Comunicação fim-a-fim**

[1] Diz-se que é lógico por que é exclusivamente mantido por software, podendo ser alterado à qualquer tempo
[2] O pacote fica encapsulado no quadro

a) A verifica que o endereço de F não pertence à sua própria rede. Despacha então o pacote para o seu "*default gateway*", o roteador $R_1$, cujo endereço físico é conhecido, através da rede local – na forma de um quadro;
b) O roteador $R_1$ recebe o quadro e o desencapsula. Vê que o endereço de destino possui uma rota válida passando pelo roteador $R_2$. Encapsula o pacote num quadro endereçado para o endereço físico da interface de $R_2$ conectada a própria rede local, e o despacha; e
c) $R_2$ recebe o quadro, e ao desencapsulá-lo verifica que o endereço de destino pertence a uma rede que está conectada a ele próprio, em outra interface física. Através dela, o roteador RESOLVE o endereço de destino (F) num endereço físico, monta o quadro a ser entregue e o despacha.

## *4.2 Técnicas de resolução de endereços*

Há três categorias básicas de resolução de endereços.

### 4.2.1 Pesquisa em Tabela

Cada rede física possui a sua tabela de *binding*. Todos os endereços IP possuem a sua correspondência para um endereço físico. Na figura 22, é apresentado um exemplo de tabela.

| IP Address | Hardware Address |
|---|---|
| 197.15.3.2 | 0A:07:4B:12:82:36 |
| 197.15.3.3 | 0A:9C:28:71:32:8D |
| 197.15.3.4 | 0A:11:C3:68:01:99 |
| 197.15.3.5 | 0A:74:59:32:CC:1F |
| 197.15.3.6 | 0A:04:BC:00:03:28 |
| 197.15.3.7 | 0A:77:81:0E:52:FA |

**Figura 22 – Tabela de *Binding***

Nesta tabela, observa-se a correspondência direta entre os endereços virtuais (IP *Address*) aos endereços físicos. A figura 23 mostra um exemplo de pesquisa direta em tabela, onde o sufixo do endereço funciona como um *array* indicando a posição onde está o seu endereço resolvido. É tipicamente usada em WANs.

**Figura 23 - Pesquisa em tabela**

### 4.2.2 Forma Fechada

Há redes onde o endereço de hardware é configurável. Neste caso, pode se usar uma função matemática para resolver endereços. Apenas para redes configuráveis.

### 4.2.3 Troca de mensagens

Os métodos anteriores são diretos. Neste método uma mensagem é enviada a alguém para solicitar o endereço físico correspondente a um lógico qualquer. Uma forma é o uso de servidores de resolução. Na outra, mais usada, uma mensagem de *broadcast* é despachada na rede local perguntando quem possui um determinado endereço físico configurado. Todas as máquinas recebem a solicitação, mas apenas o possuidor do endereço de destino responde. É a mais usada em LANs.

## *4.3 Protocolo ARP – Address Resolution Protocol*

Faz parte da pilha TCP/IP, funcionando da forma explanada no item 4.2.3. Seu funcionamento é ilustrado na figura 24.

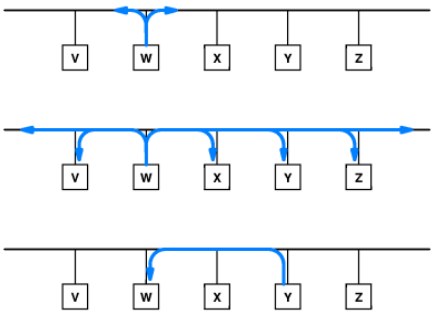


**Figura 24 - Troca de mensagens ARP**

### 4.3.1 Formato de mensagem ARP

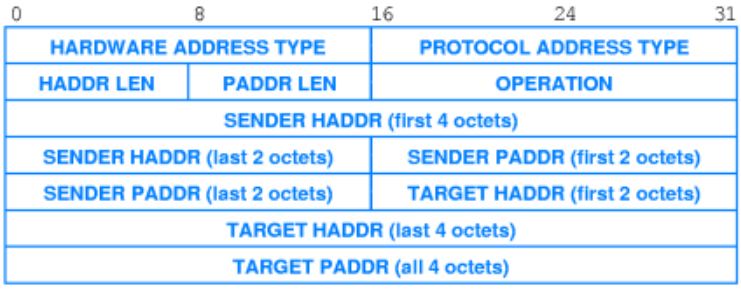


**Figura 25 - Formato da mensagem ARP**

Observe, na figura 25 os seguintes aspectos:

a) O primeiro campo especifica um tipo de endereço de hardware, e o terceiro campo o seu tamanho – isso dá versatilidade ao ARP para futuras extensões[3];
b) O campo OPERATION determina se a mensagem é uma requisição (1) ou uma resposta (2). Um único formato de mensagem então é necessário para qualquer transação do protocolo; e
c) Como o endereço de origem (físico) também é enviado, permite-se não só a descoberta de um endereço como também a amarração do endereço do emissor. As amarrações são armazenadas temporariamente em *cache*, para evitar novas solicitações[4].

# 5 Datagrama IP e Encaminhamento de Datagramas

Este capítulo descreve o formato dos pacotes e a forma como os mesmos são tratados pelos roteadores.

## *5.1 Serviço sem Conexão*

Como já foi dito, a Internet (Inter-redes) oferece um serviço sem conexões permanentes. Os protocolos de aplicação usam sua infra-estrutura, abstraíndo-se completamente de como seus dados chegam ao seu destino e suas respostas são recebidas. Os protocolos podem usar serviços sem conexão, ou no máximo, orientados à conexão.

[3] Embora quase sempre o ARP seja usado para amarrar IP (32 bits) ao MAC ETHERNET (48 bits)
[4] Se não são usadas por muito tempo (exemplo: 20 minutos) são descartadas

## 5.2 Datagrama IP

O datagrama IP é o pacote inter-rede da pilha TCP/IP, cujo formato geral é visto na figura 26.

**Figura 26 - Formato Geral do Datagrama IP**

### 5.2.1 Encaminhamento de Datagramas

Cada roteador do ambiente de inter-rede, ao receber o quadro, o passa para a camada de rede, que antes de qualquer coisa, verifica o endereço de destino e resolve a rota, com base em uma tabela de roteamento, conforme o exemplo da figura 27.

*(a)*

| Destination | Mask | Next Hop |
|---|---|---|
| 30.0.0.0 | 255.0.0.0 | 40.0.0.7 |
| 40.0.0.0 | 255.0.0.0 | deliver direct |
| 128.1.0.0 | 255.255.0.0 | deliver direct |
| 192.4.10.0 | 255.255.255.0 | 128.1.0.9 |

*(b)*

**Figura 27 - Tabela de Roteamento (roteador central)**

Mais uma vez, se observa neste exemplo que o roteamento é feito por redes, e não por host.

### 5.2.2 O Campo Máscara e Encaminhamento de Datagramas

O campo máscara é usado para extrair a parte da rede do endereço a ser roteado. Como já apresentado anteriormente, isto é feito com a operação lógica AND do endereço de destino

com a máscara correspondente. Convém ressaltar que, apesar de *next hop* (figura 27) ser um endereço IP do próximo roteador na rota para se chegar ao destino, esse endereço não é usado no datagrama, e sim apenas para o endereço de destino do quadro, após a sua resolução.

### 5.2.3 Best-Effort Delivery

O serviço de entrega de datagramas é descrito como de melhor esforço (*best-effort*), ou seja, não tratará dos seguintes problemas:

- ✓ Duplicação de datagramas;
- ✓ Entrega atrasada ou fora de ordem
- ✓ Adulteração de dados
- ✓ Perda de datagramas

Alem disso, caso o roteador fique sobrecarregado, simplesmente descartará pacotes. Qualquer controle para a recuperação do fluxo de comunicação fim-a-fim deverá ser feito em camadas superiores, por origem e destino.

### 5.2.4 Formato do Cabeçalho IP

A figura 28 apresenta o cabeçalho do datagrama IP.

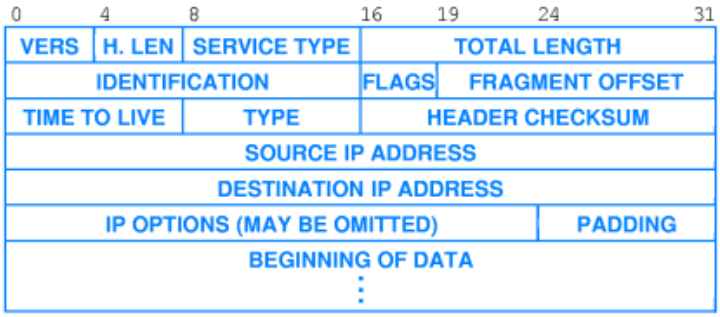


**Figura 28 - Cabeçalho do Datagrama IP**

Alguns detalhes relevantes:

- ✓ SERVICE TYPE - O emissor pode especificar se quer rotas de atraso mínimo ou de througput máximo, caso os roteadores suportem;
- ✓ ID, FLAGS e FRAGMENT OFFSET – referem-se à fragmentação de datagramas, que será discutida no próximo capítulo;

- ✓ TTL – Time to Live – é um inteiro positivo entre 1 e 255. É decrementado a cada roteador. Se chegar a 0, o roteador descarta o datagrama e gera uma mensagem de erro para o emissor;
- ✓ HEADER CHECKSUM – Computa e detecta erros **apenas no cabeçalho**;
- ✓ IP OPTIONS – Pode ou não estar presente, e são extensões do IPv4; e
- ✓ PADDING – usado para tornar, caso necessário, o cabeçalho múltiplo de 32.

# 6 Fragmentação e Remontagem de Datagramas IP

Os quadros apenas existem num segmento (enlace) de rede. Após chegar ao próximo roteador da rota, o cabeçalho do quadro é removido, o datagrama é tratado (para verificação de rota, por exemplo) para despacho ao próximo roteador e um novo quadro é criado, conforme mostrado na figura 29.

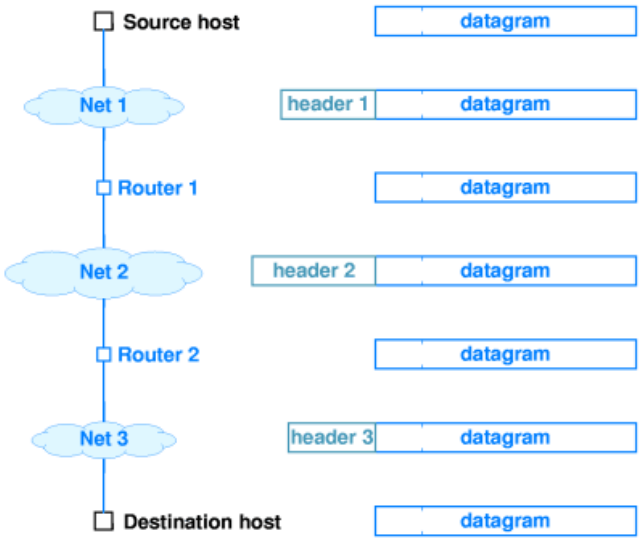


**Figura 29 - Entrega de quadros no enlace**

O quadro, com o datagrama encapsulado, é mostrado em detalhe na figura 30.

**Figura 30 - Datagrama encapsulado num quadro**

## 6.1 MTU – Maximum Transmission Unit

Todo enlace de dados possui um tamanho máximo que pode ser suportado. Como o ambiente de inter-rede pode ser bastante heterogêneo, é possível que enlaces consecutivos entre origem e destino possuam diferentes MTU, conforme ilustrado na figura 31.

**Figura 31- Enlaces com MTU diferentes**

Na figura 31, podemos observar que H1 pode transmitir datagramas de até 1500 bytes. Ao chegar ao roteador R, o datagrama deverá ser fragmentado para resolver o problema de MTU heterogêneo. Nesta técnica, usa-se o campo FLAGS (figura 28), onde um bit específico é "setado" para indicar se o datagrama é um fragmento ou se é completo. FRAGMENT OFFSET indica a posição do fragmento no datagrama original. O campo IDENTIFICATION permite a associação de um fragmento qualquer ao datagrama original, de forma que a remontagem possa ocorrer. A figura 32 mostra como se processa a fragmentação de um datagrama em três fragmentos. Cada fragmento começa com uma cópia do cabeçalho do datagrama original, acrescentando-se os três campos citados acima. Um dos fragmentos, o último, recebe um bit no campo FLAGS indicando ser ele o último.

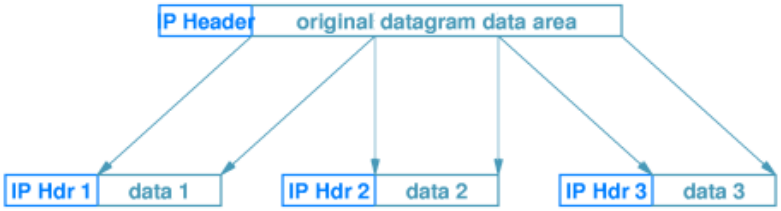


**Figura 32 - Fragmentação de um Datagrama em 3 fragmentos**

## 6.2 Remontagem

A remontagem é feita apenas no destino final. Na figura 33, por exemplo, o roteador $R_1$ dividiria o datagrama de 1500 bytes em dois, antes de despachá-lo para $R_2$, mas a remontagem não seria feita por $R_2$, apesar da rede 3 suportar o MTU de 1500 bytes. A remontagem seria feita apenas por $H_2$ (nó de destino).

**Figura 33 - Fragmentação de datagrama**

Ao receber um fragmento, o nó de destino inicia uma contagem de *time-out* para a recepção dos demais fragmentos. Se um único fragmento for perdido ou descartado no ambiente de inter-rede, ao estouro de *time-out* todos os segmentos são descartados. Não há pedido de retransmissão, pois como a fragmentação ocorre no ambiente de inter-redes, numa nova emissão, o datagrama original poderia seguir por enlaces diferentes, com MTUs diferentes, impossibilitando o reaproveitamento de outros fragmentos, remanescentes de outros enlaces com outros MTU.

# 7 IPv6

O IP versão 6, que no início de sua discussão era uma necessidade indiscutível, perdeu um pouco de sua emergência devido ao uso do CIDR, já apresentado no capítulo 3. O NAT, também discutido naquele capítulo, igualmente atenuou a necessidade de novos endereços, por permitir a substituição de classes inteiras de endereços válidos por endereços reservados e uma caixa de tradução para um único endereço válido. Independentemente disso, o IPv6 introduz vários aperfeiçoamentos em relação ao IPv4 (atual):

- ✓ 128 bits para endereçamento, *classless*, no lugar dos 32 bits atuais;
- ✓ Cabeçalhos de extensão que oferecem, por exemplo, suporte a:
  - Mobilidade
  - Privacidade criptográfica
  - Autenticação
- ✓ Permite a especificação de QoS (Qualidade de Serviço) para tráfego de áudio e vídeo

O cabeçalho principal tem o formato da figura 34.

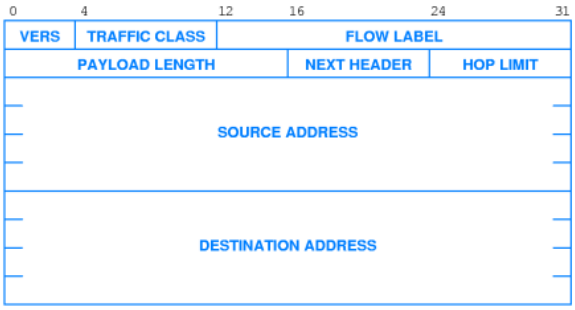


**Figura 34 - Cabeçalho do Datagrama IPv6**

Após a agregação de cabeçalhos de extensão, para o cumprimento de finalidades específicas, teria o formato da figura 35.

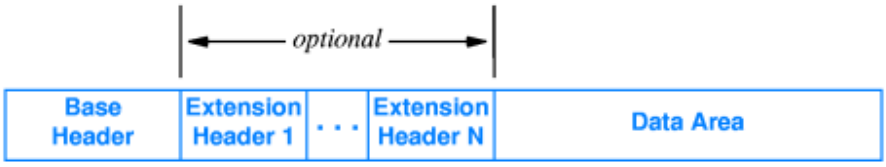


**Figura 35 - Datagrama IPv6 com extensões**

O campo FLOW LABEL pode ser usado para especificar uma classe de tráfego. O NEXT HEADER define a existência ou não de um cabeçalho de extensão, como na figura 36. Quando não houver mais cabeçalhos de extensão, NEXT HEADER especificará o tipo de dados contidos na carga útil do datagrama.

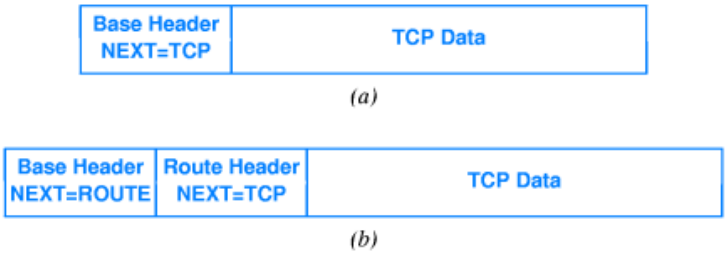


**Figura 36 - Campo NEXT HEADER**

Se, por um lado, o IPv6 aumenta o espaço de endereçamento para um tamanho virtualmente inalcançável ($2^{128}$), por outro lado aumenta a dificuldade de memorização de endereços. Ex.:

**105.220.136.100.255.255.255.0.0.18.128.140.10.255.255**

Para reduzir essa dificuldade, o endereço pode ser expresso em notação (base) hexadecimal e aos pares:

**69DC:8864:FFFF:FFFF:0:1280:8C0A:FFFF**

Como, pelo menos no início, espera-se que os endereços possuam muitos zeros, convencionou-se comprimir os zeros: Ex.:

**FF0C:0:0:0:0:0:0:B1 → FF0C::B1**

Para migração, os 32 bits dos endereços IPv4 ocupam os 32 bits menos significativos no especo de endereços IPv6, iniciando, portanto, com 96 bits em zero.

# 8 ICMP – Internet Control Message Protocol

O ICMP é um protocolo para relato de erros da camada de rede TCP/IP. A figura 37 mostra as possíveis mensagens de erro ICMP.

| Type | Name |
|---|---|
| 0 | Echo Reply |
| 1 | Unassigned |
| 2 | Unassigned |
| 3 | Destination Unreachable |
| 4 | Source Quench |
| 5 | Redirect |
| 6 | Alternate Host Address |
| 7 | Unassigned |
| 8 | Echo |
| 9 | Router Advertisement |
| 10 | Router Solicitation |
| 11 | Time Exceeded |
| 12 | Parameter Problem |
| 13 | Timestamp |
| 14 | Timestamp Reply |
| 15 | Information Request |
| 16 | Information Reply |
| 17 | Address Mask Request |
| 18 | Address Mask Reply |
| 19 | Reserved (for Security) |
| 20-29 | Reserved (for Robustness Experiment) |
| 30 | Traceroute |
| 31 | Datagram Conversion Error |
| 32 | Mobile Host Redirect |
| 33 | IPv6 Where-Are-You |
| 34 | IPv6 I-Am-Here |
| 35 | Mobile Registration Request |
| 36 | Mobile Registration Reply |
| 37 | Domain Name Request |
| 38 | Domain Name Reply |
| 39 | SKIP |
| 40 | Photuris |
| 41-255 | Reserved |

**Figura 37 - Lista de Mensagens ICMP**

Alguns exemplos;

- ✓ SOURCE QUENCH – Se o espaço de armazenamento (buffer) do roteador acaba, haverá o descarte dos datagramas recebidos e que não puderem ser processados, por indisponibilidade de espaço na fila FIFO. Ao descartar um datagrama, o roteador envia uma *source quench* para que o emissor reduza sua taxa de transmissão, para evitar novos descartes;
- ✓ TIME EXCEEDED – Se o campo TTL chega a zero, o roteador descarta o datagrama e envia uma mensagem *time exceeded*. O mesmo ocorre se o *time-out* de remontagem de fragmentos expira;
- ✓ DESTINATION UNREACHABLE – Caso o roteador verifique que o *host* de destino, ou a rede a qual o destino está conectado, está inalcançável, isso é informado através da mensagem do tipo 3;
- ✓ ECHO REQUEST/ECHO REPLY – Usada para *ping*. Ao ser enviada para um *host* (*request*), o software ICMP de destino responde com um *echo reply* usando os mesmos dados da requisição;
- ✓ ADDRESS MASK REQUEST/REPLY – Ao fazer *boot*, um *host* envia um *broadcast* requerendo a mascara de endereço usada na rede. Essa resposta é usada pelo (s) roteador (es) da rede. O software ICMP está disponível na camada IP, e

suas mensagens são encapsuladas conforme mostrado na figura 38, e são sempre enviadas para a origem do datagrama que gerou a falha.

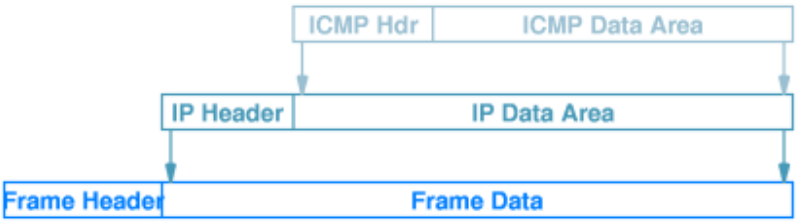


**Figura 38 - Encapsulamento ICMP**

*Traceroute* – O programa *traceroute* (*tracert*, no Windows)usa o ICMP para traçar as rotas. O processo é o seguinte:

a) O *traceroute* gera uma série de datagramas para o destino, com o TTL setado em 1;
b) O primeiro roteador no caminho decrementa o TTL, que alcança zero. O descarte ocorre, e o roteador envia uma mensagem ICMP tipo 11 para o emissor;
c) O origem recebe a mensagem e verifica qual foi o endereço do roteador que o enviou, apresentando-o no *display*;
d) O origem envia a seguir uma datagrama com TTL = 2, e assim sucessivamente, até chegar ao destino final, construindo assim toda a rota.

O ICMP também é usado para descoberta do MTU do caminho, evitando assim a fragmentação. Isso é feito através de uma mensagem ICMP e um teste que causará a mensagem de erro, da seguinte forma:

a) Um datagrama IP é enviado com um FLAG indicando que o datagrama não deve ser fragmentado;
b) Verificando que será necessária a fragmentação, e que há um *bit* indicando que isso não pode ser feito, o roteador gera uma mensagem de erro ICMP e descarta o datagrama; e
c) O origem envia sucessivos datagramas com tamanhos sucessivamente menores, até descobrir o MTU.

# 9 TCP: Transporte Confiável

Este Protocolo de Transporte executa uma função muito importante e desafiadora; fornecer um serviço de entrega de dados confiável usando um serviço de entrega de datagramas não-confiável. Oferece os seguintes serviços para a camada de aplicação:

- ✓ Orientação à Conexão – antes do envio de dados, é feita uma conexão virtual[5] com o destino, identificada por portas, para permitir o controle;
- ✓ Confiabilidade – suporte para reordenação e retransmissão de dados;
- ✓ Comunicação *full-duplex* – através de *buffers* separados para envio e para recepção.

## 9.1 Serviço fim-a-fim

O TCP é um protocolo fim-a-fim porque fornece a abstração de estar criando uma conexão direta com o host de destino, abstraindo-se do ambiente inter-rede, conforme ilustrado na figura 39.

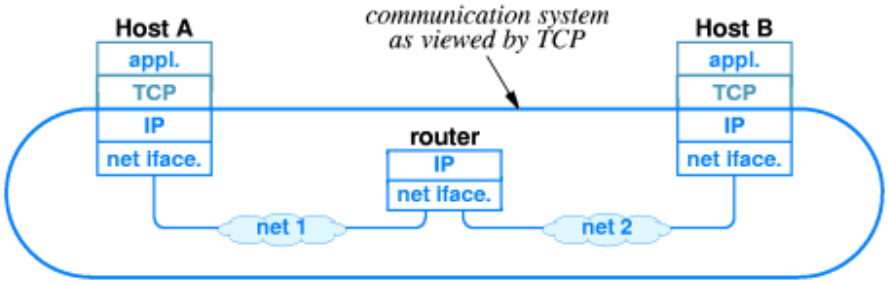


**Figura 39 - Comunicação fim-a-fim**

## 9.2 Perda de Pacotes e Retransmissão

Como a rede não é confiável, um robusto esquema de retransmissão pelo TCP é necessário. A figura 40 ilustra um exemplo desse esquema.

[5] Obtida e mantida exclusivamente por software. Logo, não há alocação de caminhos físicos ou recursos de rede.

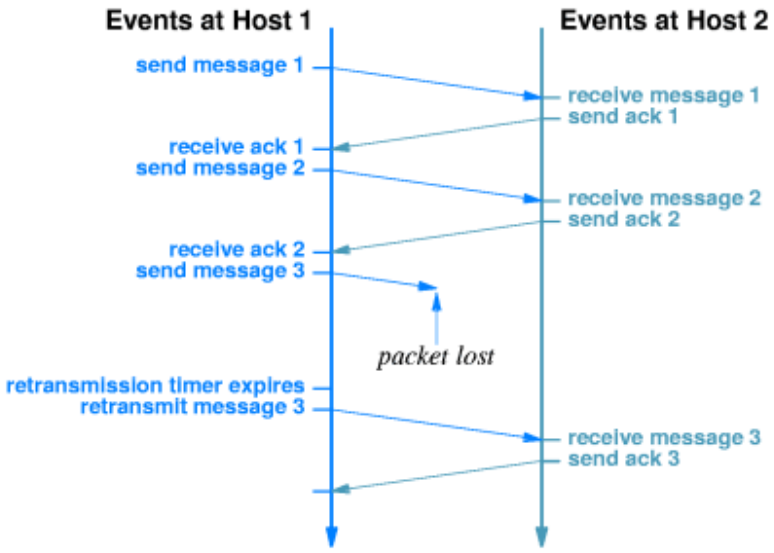


**Figura 40 - Exemplo de Retransmissão**

## 9.3 Controle de Fluxo

O TCP controla fluxo através de janelas conforme apresentado no capítulo 1. Esta janela, no entanto, pode variar de tamanho dinamicamente, de acordo com a capacidade dos pares da comunicação, conforme ilustrado na figura 41.

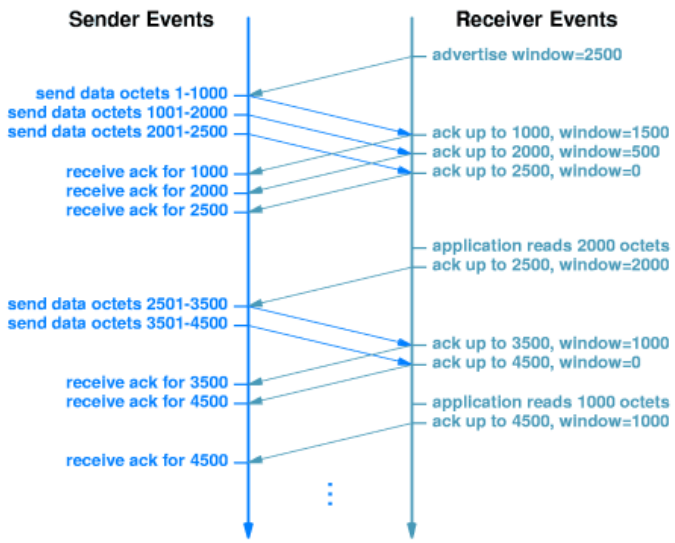


**Figura 41 - Adaptação dinâmica do tamanho da janela**

## *9.4 Three-way Handshake*

Usado para garantir que as conexões sejam estabelecidas e encerradas de forma confiável. É chamado de three-way porque são três as mensagens a serem trocadas, como apresentado na figura 42, que simula o encerramento de uma conexão. Na abertura, o FIN (de FINISH) é substituído pelo SYN (de SYNCHRONIZATION).

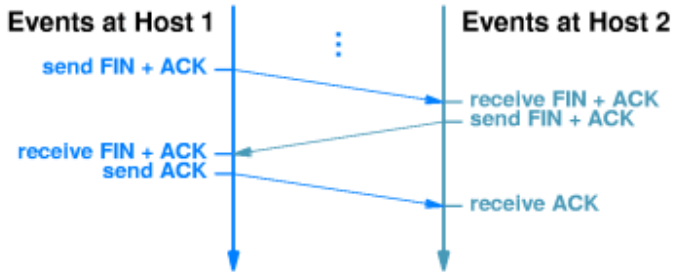


**Figura 42 - Encerramento de Conexão TCP**

## 9.5 Formato do TCP

Os dados da camada de transporte são chamados de “segmento”. O formato deste segmento é único para todas as mensagens (inclusive FIN, SYN e ACK), e é ilustrado na figura 43.

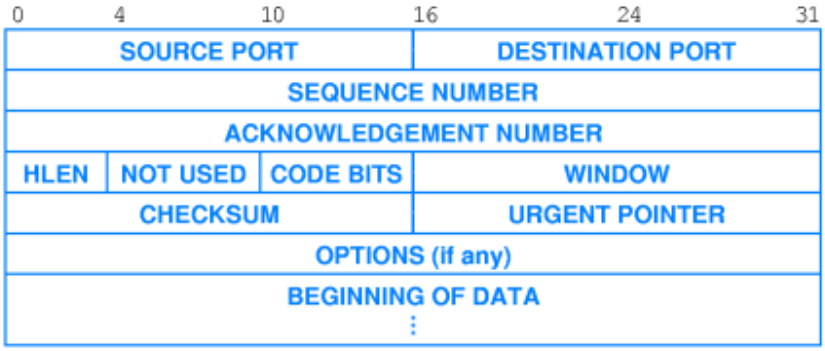


**Figura 43 - Formato do cabeçalho do segmento TCP**

## 9.6 UDP – User Datagram Protocol

O UDP é a opção de transporte do TCP/IP. Quando não há necessidade intrínseca de confiabilidade. Na verdade, o mesmo é usado quando não há segmentação dos dados do usuário, até porque não possui campos para este controle. São típicas situações de protocolos de aplicação, como o DNS, por exemplo, onde as requisições, por serem curtas, são enviadas via UDP, e, na maioria das vezes, em função do tamanho das respostas, elas são transmitidas do servidor via TCP. A figura 44 mostra o cabeçalho UDP.

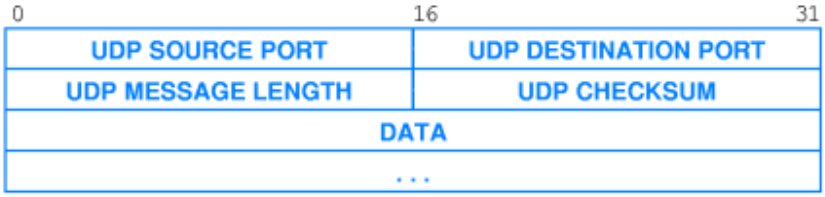


**Figura 44 - Cabeçalho UDP**

## 9.7 Encapsulamento de transporte

Como mostrado no capítulo 1, os dados da aplicação são encapsulados (aninhados) pela camada de transporte, que por sua vez, juntamente com o cabeçalho respectivo, são encapsulados pela camada de rede e assim por diante. A figura 45 ilustra essa situação, supondo-se estar usando o UDP como protocolo de transporte.

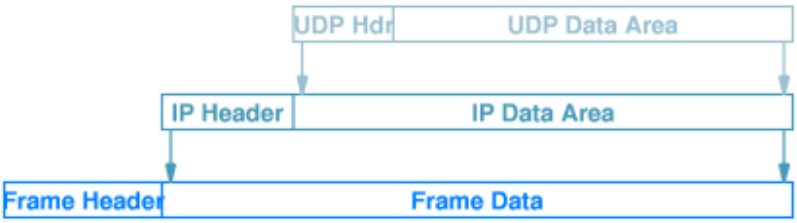


**Figura 45 - Aninhamento de Unidades de dados**

# 10 Camada de Aplicação – Alguns exemplos importantes

Neste capítulo vamos descrever apenas alguns aspectos das aplicações mais importantes para a pilha TCP/IP.

## 10.1 DNS – Domain Name System

O DNS (*Domain Name System*) foi uma das primeiras soluções criadas para popularizar a Internet, já nos seus primórdios. Inicialmente, cada *host* deveria ter uma tabela que associasse um endereço IP a um nome. É claro que esse mecanismo não escala, pois as máquinas podem trocar de endereço e novos nomes surgiam aos borbotões. Para solucionar isso, criou-se uma estrutura hierárquica que realiza esse serviço para as máquinas clientes. Pode-se, no entanto, adquirir um domínio e criar subdomínios internos. A figura 46 apresenta esta situação.

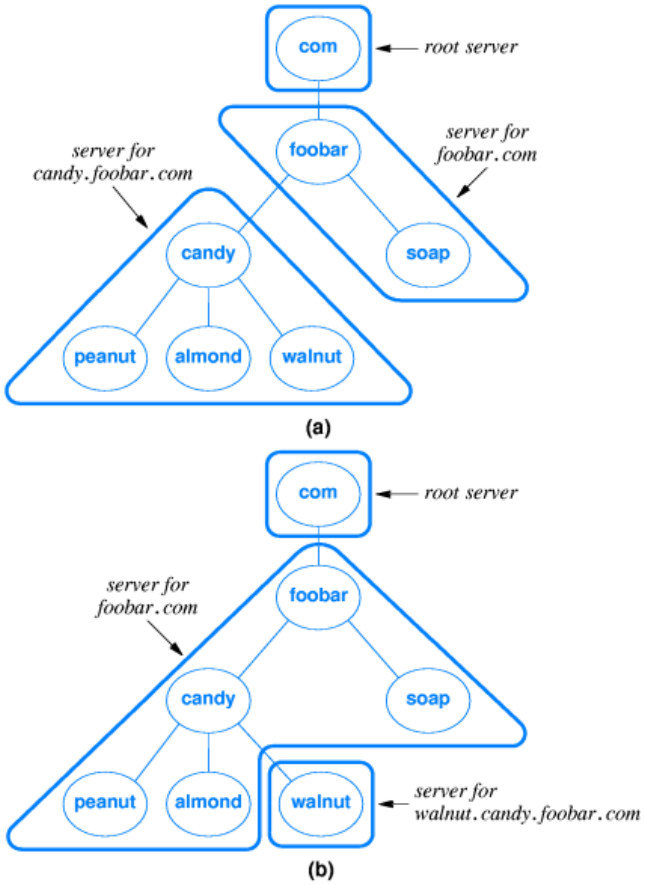


**Figura 46 - Organização Interna de um domínio DNS**

A empresa *Foobar* adquiriu o domínio *foobar.com*, ficando a seu critério a alocação de quantos domínios internos ela quiser. Na mesma figura, letra b, de acordo com a localidade física das divisões da empresa ou a quantidade de clientes em cada domínio interno, ela pode adotar uma arquitetura onde, internamente, o serviço DNS está hierarquicamente distribuído, de acordo com a facilidade de manutenção ou a necessidade da divisão do serviço.

Como a resolução de nomes é recursiva e hierárquica, o tráfego nos servidores raiz tornaria o mecanismo descrito não escalável. Para otimizar o sistema, são usados dois mecanismos: a replicação e o *caching*. A replicação nada mais é do que a cópia dos servidores raiz mundo afora. O que estiver mais disponível de acordo com a localidade do solicitante, emite a resposta. O *Caching* é ainda mais importante do que a replicação. Cada servidor mantém uma *cache* de nomes. Sempre que pesquisa um novo nome, uma cópia da resposta é armazenada em *cache*. Ao receber uma solicitação, o servidor DNS imediatamente consultado verifica sua *cache*, antes de consultar recursivamente o servidor superior.

Outra versatilidade particularmente útil do DNS é a entrada CNAME. Ela permite, a utilização de nomes alternativos para servidores. Com isso, uma solicitação para www.foobar.com pode ser direcionada para qualquer máquina que esteja designada na entrada CNAME do meu DNS *server*, permitindo assim substituições de máquinas rapidamente, sem a necessidade da troca de nomes de domínios ou endereços IP.

## 10.2 Correio eletrônico

Outra aplicação sem a qual não imaginamos mais viver é o correio eletrônico. Empresas e governos vêm substituindo suas correspondências físicas pelo e-mail, e com a evolução dos sistemas de autenticação, em curto prazo os cartórios digitais se tornarão uma realidade, permitindo a conferência de um caráter legal para um simples e-mail. Existe até uma "etiqueta" especial para o ambiente de e-mail, chamada de "*nettiquete*", usada mundialmente. Nunca escreva um e-mail em caixa alta, pois isso é encarado como se o emissor estivesse gritando com o receptor da mensagem.

Estes sistemas são construídos em duas partes, o agente do Usuário e o Agente de Transferência de mensagens. O Agente do usuário é um programa, que usamos para compor (ou ler) o e-mail. Estes Agentes podem ser mais simples ou mais sofisticados, mas a finalidade básica de todos eles é tratar com uma correspondência que possui um envelope e uma mensagem (que inclui um cabeçalho).

O grande problema atual é a diversidade de idiomas, acentuados ou não, alfabetos não-latinos, linguagens sem alfabeto (como o chinês e o japonês) e, cada vez mais, mensagens sem texto, com a transmissão de áudio ou imagens. Para solucionar isso foi criado o padrão MIME (*Multipurpose Internet Mail Extensions*). Este padrão inclui um cabeçalho que indica a versão de MIME usada. Se este cabeçalho não estiver presente, significa que a mensagem é de texto simples ASCII. Os demais cabeçalhos permitem especificar detalhadamente o que está se transmitindo, e basta que seu cliente de correio eletrônico suporte as mídias transmitidas para que ele possa ser lido. Um dos maiores problemas atuais, que é o envio de textos em HTML ou outras linguagens de *tags*, pode ser contornado com o uso do MIME, com o subtipo enriched.

No modelo original da ARPANET, todas as máquinas estavam on-line o tempo todo, permitindo o estabelecimento de conexões TCP para o envio e recepção de mensagens, como mostrado na figura 47 (a). Ocorre que, com a evolução da ARPANET para a Internet, os usuários passaram a não usar mais conexões dedicadas o tempo todo, mas sim conexões discadas sob demanda para um ISP. Neste cenário, foi necessária a criação de um outro protocolo para possibilitar o acesso ao Agente de Transferência de mensagens, agora não mais localizado na própria máquina do usuário, mas sim no servidor de Correio eletrônico do Provedor. Esse protocolo foi (e ainda é) o POP (Post Office Protocol), hoje na versão 3. seu funcionamento é ilustrado na figura 47 (b). O POP3, no entanto, faz a recuperação (*download*) dos e-mail para o HD do usuário, podendo ou não ser deletado do servidor. O problema é que, com isso, o usuário que usa várias máquinas diferentes acaba tendo suas mensagens espalhadas por vários lugares diferentes, o que não é interessante. Para resolver isso foi criado o IMAP (*Internet Message Access Protocol*), que ao invés de baixar tudo para trabalhar *off-line*, fornece facilidades para manipulação dos *e-mail* no próprio servidor.

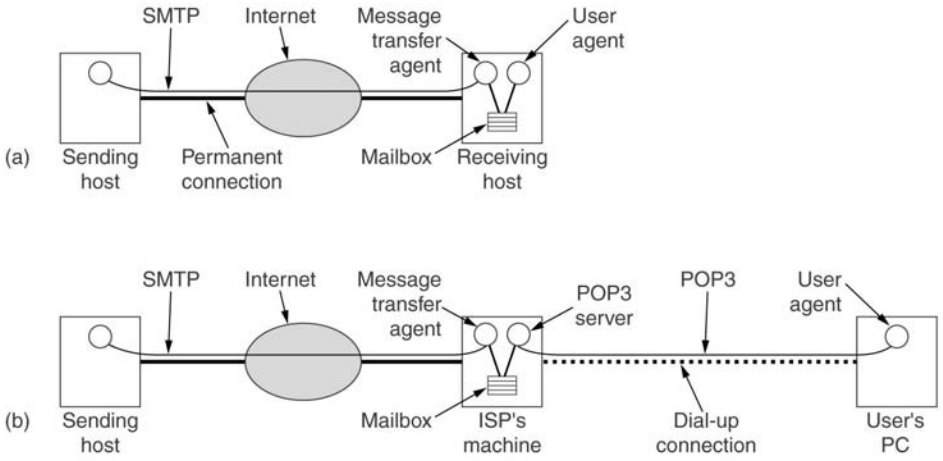


**Figura 47 - SMTP e POP**