Helcio Wagner da Silva

REDES DE COMPUTADORES

EDUFERSA

Helcio Wagner da Silva

REDES DE COMPUTADORES

EDUFERSA

©2017. Direitos Morais reservados ao autor: Helcio Wagner da Silva. Direitos Patrimoniais cedidos à Editora da Universidade Federal Rural do Semi-Árido (EdUFERSA). Não é permitida a reprodução desta obra, podendo incorrer em crime contra a propriedade intelectual, previsto no Art. 184 do Código Penal Brasileiro. Fica facultada a utilização da obra para fins educacionais, podendo a mesma ser lida, citada e referenciada. Editora signatária da Lei n. 10.994, de 14 de dezembro de 2004, que disciplina o Depósito Legal.

Reitor

José de Arimatea de Matos

Vice-Reitor

José Domingues Fontenele Neto

Pró-Reitor de Extensão e Cultura

Rodrigo Sérgio Ferreira de Moura

Coordenador Editorial

Pacelli Costa

CONSELHO EDITORIAL

Pacelli Costa, Walter Martins Rodrigues, Francisco Franciné Maia Júnior, Rafael Castelo Guedes Martins, Keina Cristina S. Sousa, Antonio Ronaldo Gomes Garcia, Auristela Crisanto da Cunha, Janilson Pinheiro de Assis, Luís Cesar de Aquino Lemos Filho, Rodrigo Silva da Costa e Valquíria Melo Souza Correia.

EQUIPE TÉCNICA

Francisca Nataligeuza Maia de Fontes (Secretária), José Arimateia da Silva (Designer Gráfico), Pacelli Costa (Bibliotecário), Nichollas Rennah (Analista de Sistemas).

Dados Internacionais da Catalogação na Publicação (CIP)
Editora Universitária (EdUFERSA)

S861r Silva, Helcio Wagner da.

Redes de computadores/ Helcio Wagner da Silva — Mossoró: EdUFERSA, 2017.

154p.: il.

ISBN: 978-55-5777-075-7

1. Redes de computadores. 2. DNS - Computação. 3. Nível de rede. 4. Roteamento dinâmico. I. Título.

EdUFERSA CDD – 004.6

Bibliotecário-Documentalista
Pacelli Costa (CRB15-658)
Editora filiada:

Av. Francisco Mota, 572 (Campus Leste, Centro de Convivência) Costa e Silva | Mossoró-RN | 59.625-900 | +55 (84) 3317-8267 | http://edufersa.ufersa.edu.br | edufersa@ufersa.edu.br

Governo Federal
Ministro de Educação
José Mendonça Bezerra Filho

Universidade Aberta do Brasil
Responsável pela Diretoria da Educação à Distância
Carlos Cezar Modernel Lenuzza

Universidade Federal Rural do Semi-Árido
Reitor
José de Arimatea de Matos

Pró-Reitor de Graduação
Rodrigo Nogueira de Codes

Núcleo de Educação a Distância
Coordenadora UAB
Valdenize Lopes do Nascimento

• EQUIPE TÉCNICA DO NEAD •

Equipe multidisciplinar
Antônio Charleskson Lopes Pinheiro – Diretor de Produção de Material Didático
Ulisses de Melo Furtado – Gerente de Redes e Ambientes Virtuais
Ângelo Gustavo Mendes Costa - Técnico em Assuntos Educacionais
Adriana Mara Guimarães de Farias – Programadora
Jéssica de Oliveira Fernandes - Social Media
Francisca Monteiro da Silva Perez - Gerente Acadêmico
Ramon Ribeiro Vitorino Rodrigues - Gerente de Criação e Design
José Antonio da Silva - Design de Artes Visuais
Mikael Oliveira de Meneses – Design de Artes Visuais
Ronald Soares Shyu - Intérprete de Libras

Arte da capa
Felipe de Araújo Alves

Edição
EDUFERSA

Equipe administrativa
Rafaela Cristina Alves de Freitas – Assistente em Administração
Iriane Teresa de Araújo – Gerente Financeiro
Antonio Romário Bezerra Nogueira - Assistente Acadêmico
Danielle França Ribeiro - Assistente Financeiro

Equipe de apoio
Nayra Maria da Costa Lima – Design Instrucional
Jarleide Deyse de Souza - Estágiária
Reginaldo Castro da Silva Júnior - Estagiário
Manoel Alcino de Lima Neto - Estagiário

http://nead.ufersa.edu.br/

APRESENTAÇÃO DA DISCIPLINA

Caros alunos,

Sejam bem-vindos à disciplina Redes de Computadores. Esta trata-se de uma disciplina clássica dos cursos de graduação na área de computação. No entanto, a abordagem de redes de computadores por parte de alguns livros (muitos dos quais campeões de vendas) nem sempre é objetiva, havendo ainda textos repletos de generalizações.

Este caderno, por sua vez, segue uma linha mais objetiva ao abordar o assunto naquilo que é seu foco: os protocolos de comunicação e a relação entre eles, de forma a prover os serviços que a maioria das pessoas no planeta utiliza para acesso a conteúdo e interações remotas.

Redes de computadores é uma disciplina muito importante para o profissional de computação porque lhe permite enxergar aquilo que está ao seu redor. Com base na consciência situacional adquirida ao longo da disciplina, ele será capaz de analisar determinados problemas relacionados a redes e adotar soluções adequadas.

SOBRE O AUTOR

HELCIO WAGNER DA SILVA

Graduado em Engenharia Elétrica, pela Universidade Federal do Rio Grande do Norte (UFRN), em 1998. É também Mestre em Engenharia Elétrica pela UFRN, na área de Engenharia da Computação, tendo obtido seu título em 2000. Cursou doutorado em Engenharia Elétrica nas áreas de Telecomunicações e Telemática, concluído em 2006, na Universidade Estadual de Campinas (UNICAMP). Desde 2008, é professor do curso de Ciência da Computação da Universidade Federal Rural do Semi-Árido (UFERSA), onde ministra as disciplinas Redes de Computadores I e Redes de Computadores II, ambas na graduação.

SUMÁRIO

UNIDADE 1: Os Níveis de Aplicação e Transporte
Introdução
HTTP
FTP
Correio Eletrônico
DNS
TCP & UDP

UNIDADE 2: O Nível de Rede
Introdução
Endereços e Máscaras
Tabela de Rotas e Roteamento
A Escassez de Endereços IPv4
CIDR
Agregação de Rotas
Regra de Compatibilidade com o Prefixo mais Longo
NAT
Diferenciação de Datagramas
Fragmentação de Datagramas
Aliviando os Efeitos em Laços de Roteamento
Especificando a Quem Entregar o Conteúdo Encapsulado
IPv6 – Um Novo Começo
Introdução ao Roteamento Dinâmico
RIP
OSPF
BGP

UNIDADE 3: Os Níveis de Enlace e Físico
ICMP
ARP & RARP
DHCP
Redes Ethernet
VLANs
STP
WiFi
Noções de Cabeamento Estruturado

I OS NÍVEIS DE APLICAÇÃO E TRANSPORTE

Esta unidade fornece uma introdução às redes de computadores, descrevendo em linhas gerais a arquitetura de protocolos em vigência na internet – o famoso modelo TCP/IP. Após esta introdução, a unidade aborda protocolos de grande notoriedade pertencentes ao nível de aplicação, a saber: HTTP, FTP, SMTP, POP3/IMAP e DNS. Por fim, serão estudados os protocolos do nível de transporte (TCP e UDP). Além do estudo dos protocolos do nível de aplicação, esta unidade descreve como aqueles protocolos utilizam os serviços prestados pelos protocolos do nível de transporte.

Objetivos:

- Introduzir a disciplina ao aluno;
- Apresentar os principais protocolos do nível de aplicação;
- Abordar os protocolos do nível de transporte;
- Demonstrar como os protocolos do nível de aplicação utilizam os serviços prestados pelos protocolos do nível de transporte.

Introdução

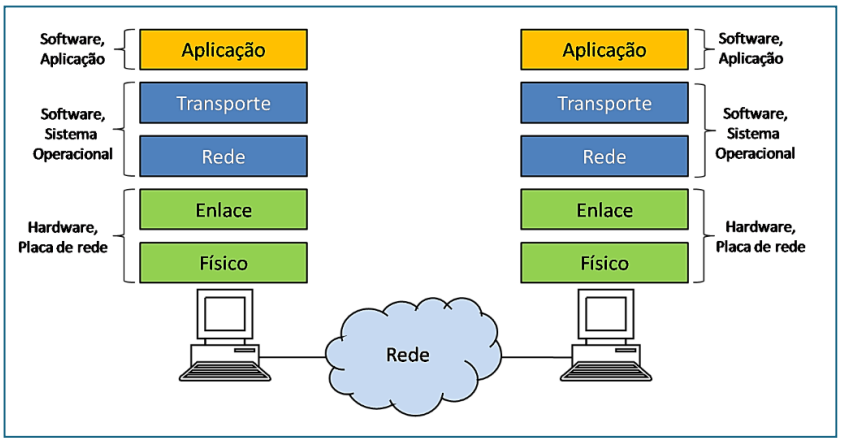
Atualmente, a utilização de redes de computadores é comum na vida de qualquer ser humano. Qualquer pessoa, por exemplo, utiliza a internet para acessar informações e interagir com outras pessoas por meio de aplicações. Isto, naturalmente, faz com que haja uma maior necessidade de profissionais aptos a entender, projetar e administrar redes de computadores.

Para que dois computadores possam se comunicar através de uma rede, ambos devem utilizar um mesmo protocolo de comunicação, definido em termos de um conjunto de mensagens denominadas PDUs (*Protocol Data Units*) – e de uma sequência na qual estas mensagens devem ser trocadas entre as aplicações.

Antigamente, cada fabricante de computadores definia seu protocolo de comunicação, que não era divulgado aos seus concorrentes. O resultado era que um computador de um fabricante só podia se comunicar com outro do mesmo fabricante – o que forçava a empresa ou pessoa a comprar seus computadores daquele fabricante. Para solucionar esse problema, a ISO (*International Organization for Standardization*) definiu o RM-OSI (*Reference Model for Open Systems Interconnection*). Como o próprio nome indica, o RM-OSI era um modelo que seria utilizado como referência pelos fabricantes para que os computadores produzidos por eles pudessem se comunicar com computadores produzidos por outros fabricantes.

Diferentemente dos outros modelos, que eram proprietários e fechados, o RM-OSI era um modelo aberto – isto é, estava disponível a todos que se interessassem por ele. No entanto, sua relativa complexidade atrapalhou a adoção pelo mercado. Na prática, o modelo utilizado é outro, inspirado no RM-OSI, porém mais simplificado. Este modelo é conhecido como TCP/IP – uma homenagem a dois de seus protocolos mais famosos. Como ilustra a figura 1, trata-se da especificação de um conjunto de níveis ou camadas empilhados. A maioria dos níveis é implementada em *software*, seja pela aplicação em si (no nível de aplicação) quanto pelo sistema operacional (é o caso dos níveis de transporte e rede). Os níveis de enlace e físico são implementados em *hardware*, na placa de rede do computador.

Figura 1: Níveis do modelo TCP/IP



A tabela 1 resume a função de cada um dos níveis do modelo TCP/IP. Cada um deles é responsável por um aspecto diferente da comunicação. A função do nível de aplicação, por exemplo, é dependente da natureza da aplicação: pode ser a transferência de arquivos, o envio de uma mensagem de correio eletrônico, etc. Os dados do nível de aplicação podem se perder no meio do caminho, e cabe ao nível de transporte retransmiti-los, caso esta perda ocorra. Para chegar ao seu destino, estes dados devem percorrer o melhor caminho possível, dentre vários. É função do nível de rede guiá-los através do melhor caminho possível.

O primeiro passo, naturalmente, é fazer com que os dados saiam do computador. A função do nível de enlace é obter o acesso ao meio físico para transmiti-los. Esta transmissão, na prática, consiste na conversão dos dados em sinais eletromagnéticos – e esta é a função do nível físico.

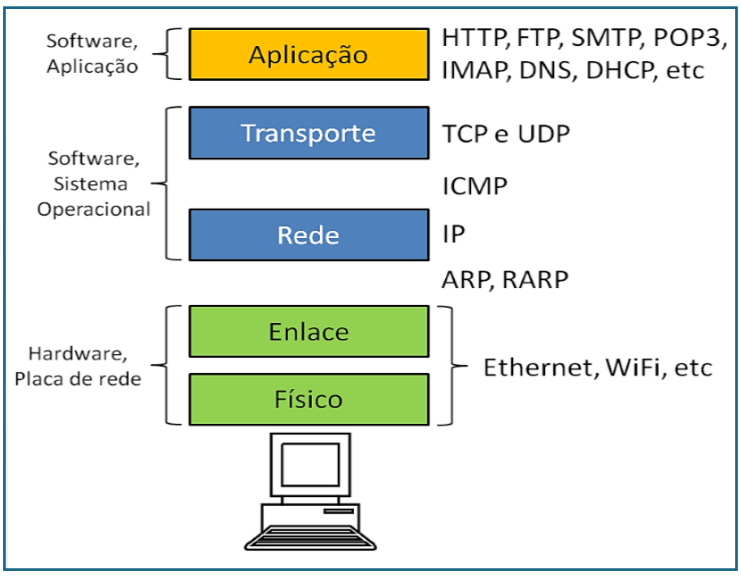
Tabela 1: Função de cada nível do modelo TCP/IP

NÍVEL	FUNÇÃO
Aplicação	Depende da aplicação – enviar uma mensagem de correio eletrônico, por exemplo.
Transporte	Fornecer ao nível de aplicação um serviço confiável de entrega desta mensagem.
Rede	Fazer com que um pacote encapsulando a mensagem chegue ao destino através do melhor caminho possível.
Enlace	Obter o acesso ao meio de transmissão para transmissão do pacote.
Físico	Transmitir o pacote na forma de sinais eletromagnéticos no meio de transmissão.

Cada um dos níveis do modelo TCP/IP é representado por um ou mais protocolos, e o estudo das redes de computadores é o estudo daqueles protocolos. A figura 2 ilustra alguns deles. O nível de aplicação, cuja funcionalidade depende da aplicação, possui virtualmente infinitos protocolos. Os protocolos listados na figura (HTTP, FTP, SMTP, POP3, IMAP, DNS e DHCP) serão objetos de estudo de capítulos posteriores. O nível de transporte, por outro lado, possui apenas dois protocolos – o TCP e o UDP.

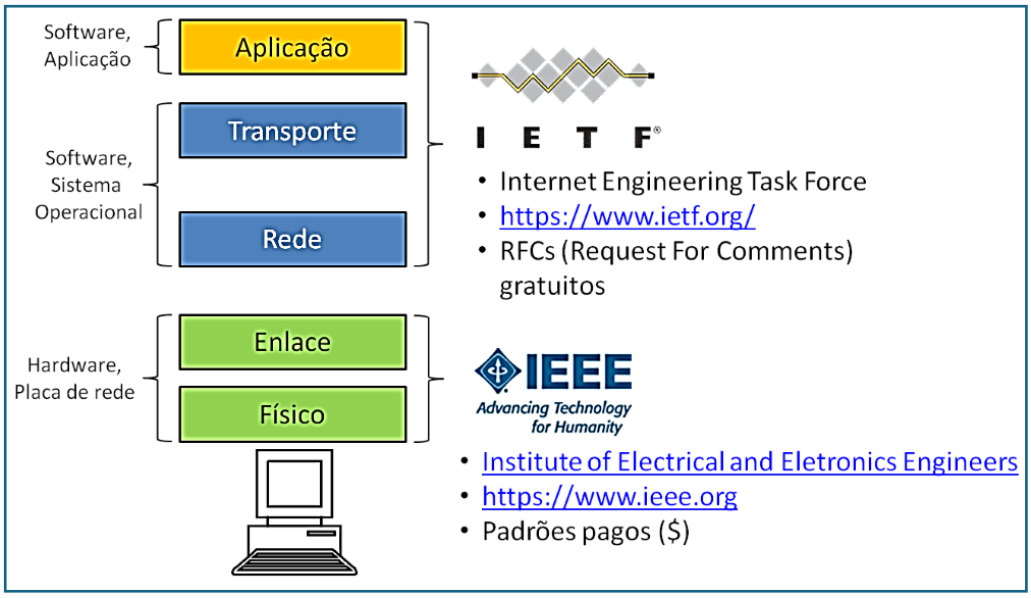
O nível de rede possui apenas o protocolo IP como seu representante. Os níveis de enlace e físico são aqueles nos quais são definidas tecnologias como Ethernet (e suas variantes, Fast Ethernet e Gigabit Ethernet) e WiFi. A figura 2 ilustra ainda alguns protocolos cuja localização é polêmica, não havendo consenso sobre eles pertencerem ou não a determinado nível. É o caso dos protocolos ICMP, ARP e RARP, havendo muitos outros naquela condição.

Figura 2: Alguns protocolos definidos no modelo TCP/IP



Todos os protocolos mencionados na figura 2 são especificados em documentos próprios. Conforme ilustra a figura 3, os protocolos dos níveis de aplicação, transporte e rede são especificados no âmbito da IETF (*Internet Engineering Task Force*) na forma de documentos acessíveis, gratuitamente, denominados RFCs (*Request For Comments*). Por outro lado, os protocolos dos níveis de enlace e físico são especificados no âmbito do IEEE (*Institute of Eletrical and Eletronics Engineers*) na forma de documentos cuja aquisição demanda certa quantia de dinheiro.

Figura 3: Organizações de padronização no modelo TCP/IP



Estudar determinado protocolo é destacar seus principais pontos e detalhar seu relacionamento com os demais. Neste contexto, pode-se abordar o modelo TCP/IP de duas formas. Uma abordagem consiste em iniciar o estudo pelo nível de aplicação e descer até o nível físico, algo que é conhecido como abordagem *Top-Down*. É a abordagem adotada por Kurose; Ross (2010). A outra abordagem é caracterizada pelo início do estudo pelo nível físico e o seu término pelo nível de aplicação, numa abordagem conhecida como *Bottom-Up*. É o caso de autores como Tanenbaum; Wetherall (2011) e Comer (2000). Este caderno utiliza a abordagem *Top-Down,* tendo em vista que os protocolos do nível de aplicação são mais facilmente perceptíveis pelo aluno do que as formas segundo as quais os sinais são transmitidos no nível físico.

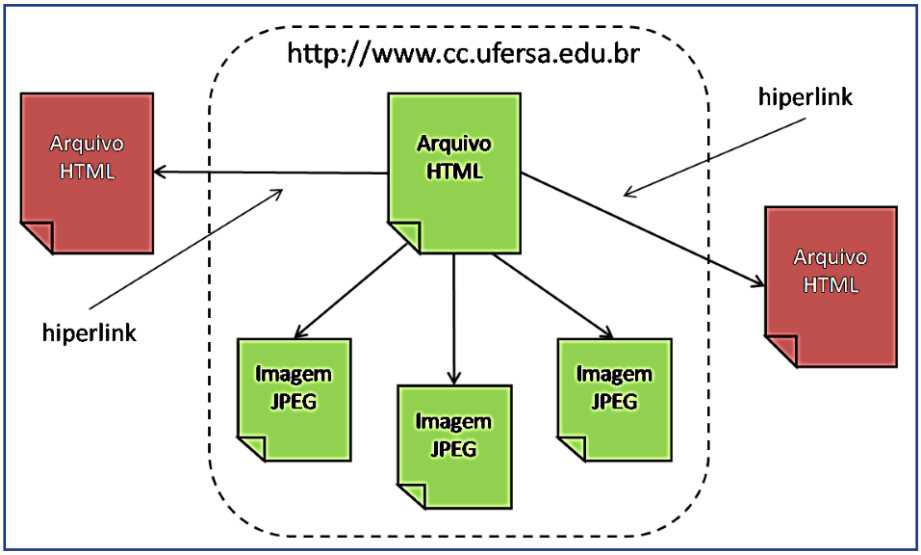
Neste contexto, inclusive, o caderno provê informações minimalistas acerca daquele nível, resumindo-se a noções de cabeamento estruturado. Explica-se esta decisão pelo fato de que a compreensão de estratégias de transmissão de sinais no nível físico requer um domínio de ferramentas matemáticas que não estão (e nem devem estar) ao alcance do estudante do curso de Ciência da Computação.

HTTP

Definitivamente, o intrincado de páginas web que serve como pano de fundo para a busca e troca de informações confunde-se com a própria internet. Em outras palavras, muitas pessoas dizem que aquelas páginas são a própria internet, o que não é verdade. A internet surgiu no final da década de 1960 e funcionou muito tempo no "modo texto". A ideia da chamada WWW (*World Wide Web*) foi proposta no final da década de 1980 pelo físico Tim Bernes-Lee, no CERN (*European Organization for Nuclear Research*). Em termos de importância para a humanidade, a web é por vezes comparada ao rádio e a TV.

A figura 4 ilustra um exemplo de como o conteúdo da web é estruturado arquitetonicamente. Os documentos delimitados pela superfície pontilhada correspondem ao conteúdo exibido quando um usuário digita o endereço http://www.cc.ufersa.edu.br em um navegador (*browser*). O que este usuário vê é a exibição do texto codificado em HTML (*HyperText Markup Language*) e três imagens JPEG (*Joint Photographic Experts Group*). Ele vê também *hiperlinks* para os outros dois documentos HTML – se clicar em algum daqueles hiperlinks, o documento para o qual ele aponta passará a ser exibido.

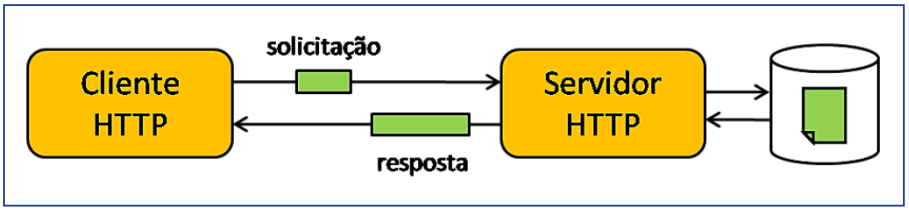
Figura 4: Um arquivo HTML com referências diretas a quatro imagens e com *hiperlinks*



A estrutura da web é justamente essa: objetos que fazem referência a outros objetos que são exibidos junto com eles, como o arquivo HTML as três imagens JPEG, ou a outros objetos que estão à distância de um clique, como os dois outros arquivos HTML. O HTTP (*HyperText Transfer Protocol*) é o protocolo utilizado para a transferência (de cópias) desses objetos a partir do fornecimento, por parte do usuário, de um endereço denominado URL (*Uniform Resource Locator*).

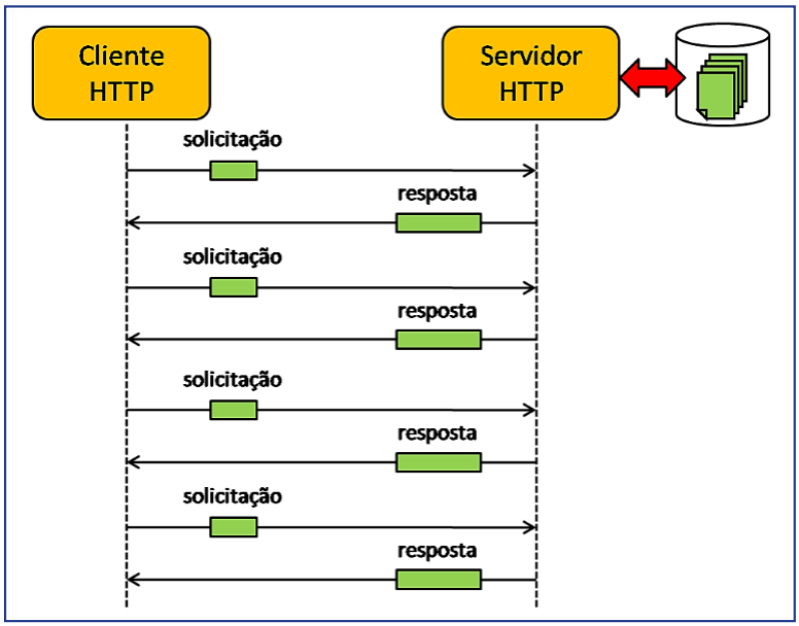
O HTTP é baseado no paradigma cliente/servidor e funciona de forma simples, conforme ilustrado pela Figura 5: um cliente HTTP (browser) envia uma solicitação para um servidor web, buscando obter um objeto a partir de um endereço. O servidor então retorna uma cópia daquele objeto em uma resposta HTTP.

Figura 5: Envio de solicitação HTTP e recepção de resposta



Na obtenção de uma página HTML que referencia diretamente outros objetos, como é o caso da página da Figura 4 (que referencia três imagens JPEG), o cliente envia automaticamente outras solicitações para obtenção destes outros objetos – conforme ilustrado pela figura 6. Nela, o *browser* envia uma solicitação HTTP para obtenção da página HTML a partir de http://www.cc.ufersa.edu.br e, ao receber o arquivo HTML, descobre que ele referencia diretamente três imagens JPEG. Ele então envia novas solicitações HTTP, cada uma delas visando à obtenção de uma das imagens.

Figura 6: Envio automático de várias solicitações HTTP

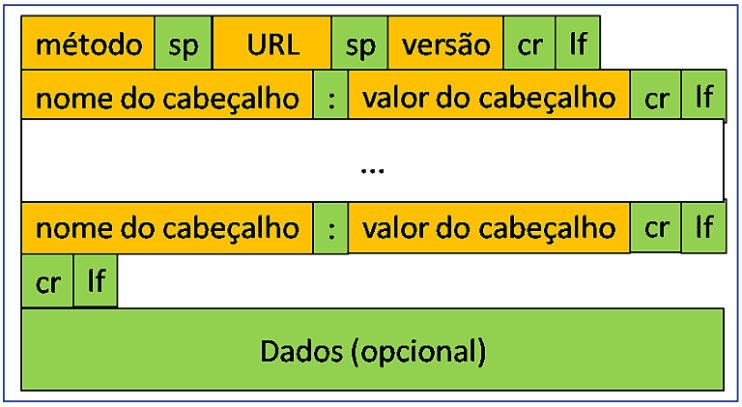


Uma solicitação HTTP tem o formato ilustrado pela figura 7. A sua primeira linha inicia com a descrição do método utilizado, o URL e a versão do protocolo. Estes campos são separados pelo caractere de espaço, sp. A linha (bem como as demais) se encerra com os caracteres cr (*carriage return*) e lf (*line feed*). Há vários métodos disponíveis no HTTP, porém os mais populares são GET e POST. O GET é o método mais utilizado para obter objetos e solicitações que usam este método não levam informações ao servidor – ou levam muito pouco, como é o caso de uma solicitação de busca.

Por outro lado, solicitações que usam o método POST são utilizadas, justamente, para levar informações ao servidor – o preenchimento de formulário ou *upload* de um arquivo, por exemplo. O URL representa o caminho, dentro do domínio, para o objeto desejado. O campo versão identifica qual a versão do HTTP utilizada – será visto mais adiante que o HTTP possui duas versões. Seguindo-se esta primeira linha, há uma série de linhas de cabeçalho.

Cada um deles possui um propósito específico: há cabeçalhos para definir o domínio da solicitação, fornecer informações sobre o *browser*, descrever os tipos de conteúdo suportados por ele, etc. Cada cabeçalho é composto por um nome e um valor, separados pelo caractere ":" (dois-pontos).

Figura 7: Formato da solicitação HTTP



A figura 8 ilustra o formato de uma resposta HTTP. A linha inicial possui os campos versão, código de status e frase. A esta linha seguem linhas de cabeçalho, informando a data em que a resposta foi enviada, o tamanho do campo de dados, o tipo do conteúdo, etc. A tabela 2 ilustra o formato definido para os diferentes códigos de status, bem como alguns exemplos e suas frases correspondentes.

Figura 8: Formato da resposta HTTP

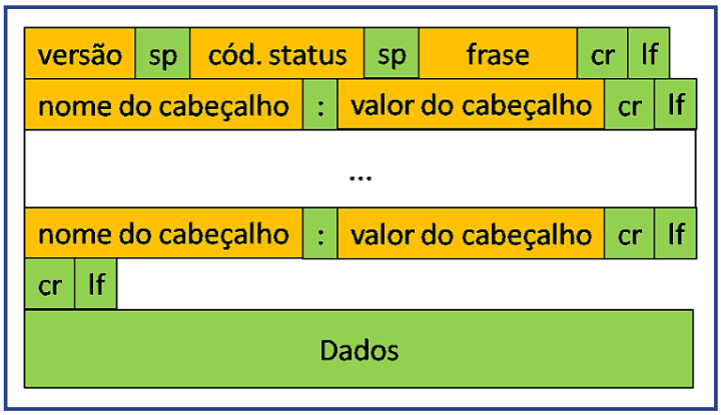


Tabela 2: Exemplos de códigos de status e frases correspondentes

CÓDIGO DE *STATUS* → SIGNIFICADO	EXEMPLO DE FRASE → SIGNIFICADO
1xx → Informação	100 Continue → o servidor concorda em atender a requisição do cliente
2xx → Sucesso	200 OK → requisição bem-sucedida
3xx → Redirecionamento	304 Not Modified → a página na cache ainda é válida
4xx → Erro do cliente	404 Not, Found → página não encontrada
5xx → Erro no servidor	500 Internal Server Error → erro interno no servidor

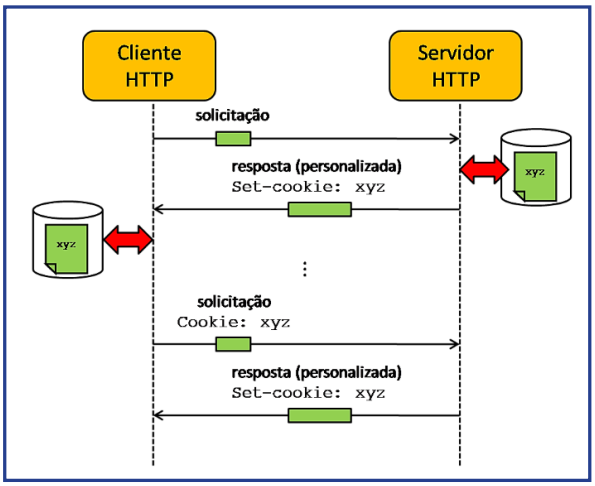
O HTTP é um protocolo considerado *stateless*. Isto significa que um servidor web, em princípio, não armazena informações sobre quem está enviando uma solicitação, apenas respondendo a ela. Porém, há uma forma de um servidor web saber quem enviou uma solicitação recém-recebida. Trata-se da utilização de *cookies*.

Criados pela Netscape, os *cookies* são informações enviadas pelos servidores web junto com uma resposta a uma solicitação. Quando novas solicitações são enviadas àqueles servidores, os *cookies* vão junto com elas, permitindo ao servidor saber quem está enviando as solicitações. A figura 9 ilustra um exemplo de obtenção e utilização de *cookies*. Nela, um servidor HTTP recebe uma solicitação e constrói uma resposta personalizada.

Esta resposta pode ser construída com base em alguma informação fornecida pelo usuário – como uma palavra de busca, por exemplo. O fato é que o servidor HTTP cria um *cookie* para aquele usuário e o armazena em uma base de dados. Junto com o *cookie*, ele armazena algumas outras informações do usuário, mas é apenas o cookie que ele retorna na resposta personalizada, na forma de um cabeçalho *set-cookie*.

Quando o cliente HTTP recebe a resposta, ele também armazena o cookie em uma pequena base de dados, associando-o ao servidor HTTP que o enviou. Assim, quando uma nova solicitação é enviada àquele servidor, o *cookie* é enviado em um cabeçalho *cookie*.

Figura 9: Exemplo de utilização de cookies



Grande parte do tráfego na internet vem de (cópias de) objetos web transmitidos via HTTP – e o fenômeno das redes sociais contribui muito para isto. Assim, de forma a contribuir para que este tráfego não sobrecarregue os enlaces, estratégias de cache foram criadas. A primeira delas atende pelo nome de GET condicional. Ela se baseia no armazenamento, por parte do cliente, de objetos recebidos.

Na Figura 10, por exemplo, o cliente HTTP envia uma solicitação cuja resposta contém, além do objeto propriamente dito, o cabeçalho *Last-modified-since,* que diz qual foi a última data em que este objeto foi modificado no servidor – no caso, em uma terça-feira, 16 de Junho de 2016.

Ao receber este objeto, o cliente salva uma cópia dele em uma base de dados local. Esta base de dados é acessada quando uma nova solicitação HTTP for enviada e, ao saber que o objeto consta nela, o cliente adiciona o cabeçalho *If-modified-since* à solicitação com o valor fornecido pelo cabeçalho *Last-modified-since*. Uma vez recebida a solicitação, o servidor HTTP tem duas opções:

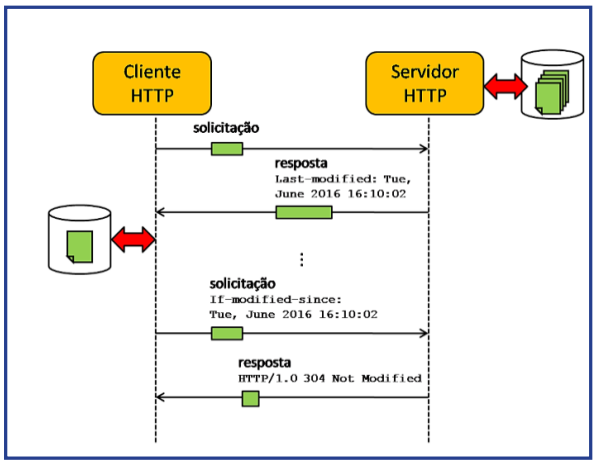
• Se o objeto sofreu alguma mudança desde a data informada no cabeçalho *If-modified-since*, uma nova cópia do objeto é enviada em uma resposta HTTP cujos código de status e frase são

200 e OK, respectivamente;

• Porém, se o objeto não sofreu nenhuma mudança desde a data informada no cabeçalho *If-modified-since*, o servidor envia resposta HTTP com código de status e frase iguais a 304 e *Not Modified*, respectivamente, e o campo de dados vazio.

Na figura 10, que ilustra a segunda opção, o cliente recebe a resposta HTTP e verifica que a cópia do objeto armazenada localmente ainda é válida e a exibe ao usuário. Esta estratégia é eficiente porque respostas vazias produzem menor tráfego na rede e a exibição dos objetos, por estarem localizados localmente, se torna mais rápida.

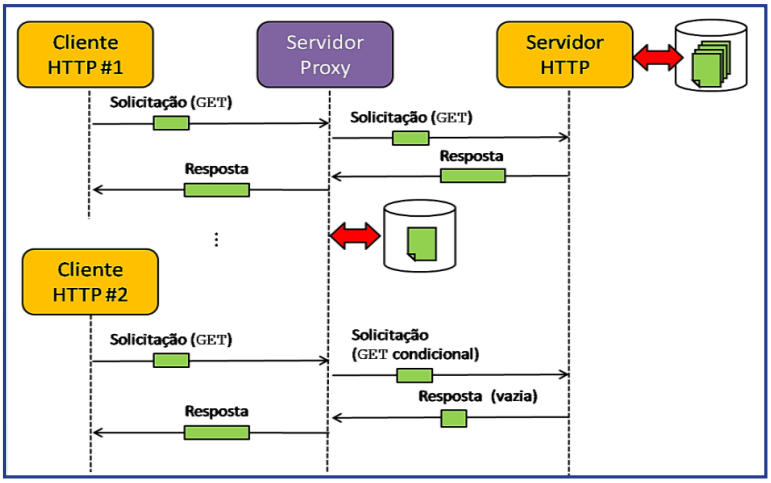
Figura 10: Exemplo de GET condicional



Outra estratégia para diminuição do tráfego devido à web é a adoção de um servidor *proxy*. Este tipo de servidor atua intermediando as solicitações HTTP enviadas por vários clientes em um ambiente corporativo. A figura 11 ilustra um exemplo de utilização de um servidor *proxy.* Nela, um cliente HTTP obtém um objeto mediante intermediação do servidor *proxy*.

Este servidor armazena localmente uma cópia do objeto de forma que, quando outro cliente HTTP solicita aquele mesmo objeto, o servidor envia uma nova solicitação na forma de um GET condicional. Supondo que o objeto não tenha sido modificado no servidor HTTP, uma resposta vazia é retornada para o servidor *proxy* – que retorna a cópia do objeto armazenada localmente.

Figura 11: Exemplo de servidor *proxy*

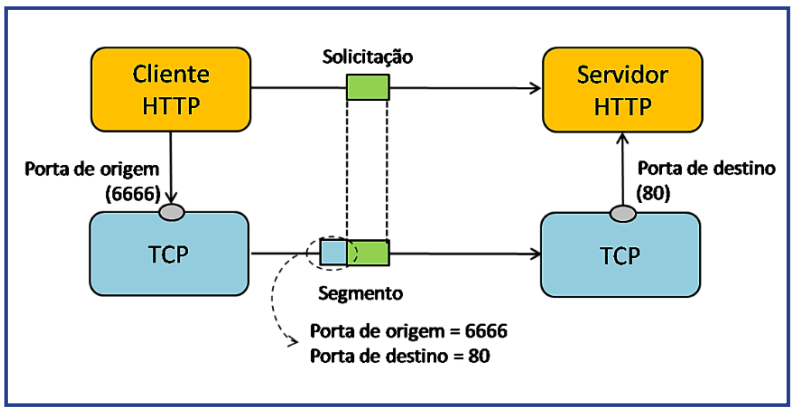


O HTTP é um usuário dos serviços prestados pelo TCP, um dos protocolos definidos para o nível de transporte. O HTTP utiliza o TCP para que este último garanta que solicitações e respostas HTTP sejam transmitidas e recebidas integralmente.

A Figura 12 ilustra uma solicitação HTTP encapsulada por um segmento TCP – este é o nome atribuído às PDUs do nível de transporte. Para que isto ocorra, o cliente HTTP tem que enviar a solicitação por meio de uma porta TCP – este é o nome atribuído aos pontos de acesso aos serviços do nível de transporte. A porta é identificada por um número de 16 bits, o que provoca a existência de 65.536 portas diferentes para o TCP e o mesmo número para o UDP. As portas de 0 a 1023 são denominadas portas reservadas, no sentido em que somente serviços padronizados pela IETF podem utilizá-las. No contexto do HTTP, o servidor permanece associado à porta reservada de número 80 desde o momento em que é inicializado até o momento em que é encerrado. O cliente, por sua vez, associa-se a uma porta aleatória para o envio de uma solicitação e a recepção de uma resposta. Após isto, ele é desassociado daquela porta.

Na Figura 12, o cliente envia a solicitação pela porta 6666 e o TCP a encapsula em um segmento que possui em seu cabeçalho a identificação daquela porta e da porta de destino (80), à qual está associado o servidor. A informação da porta de destino é extremamente útil para o servidor, pois ele precisa saber para onde enviar a PDU encapsulada pelo segmento – por definição, o TCP não sabe qual é o tipo da PDU encapsulada em um segmento.

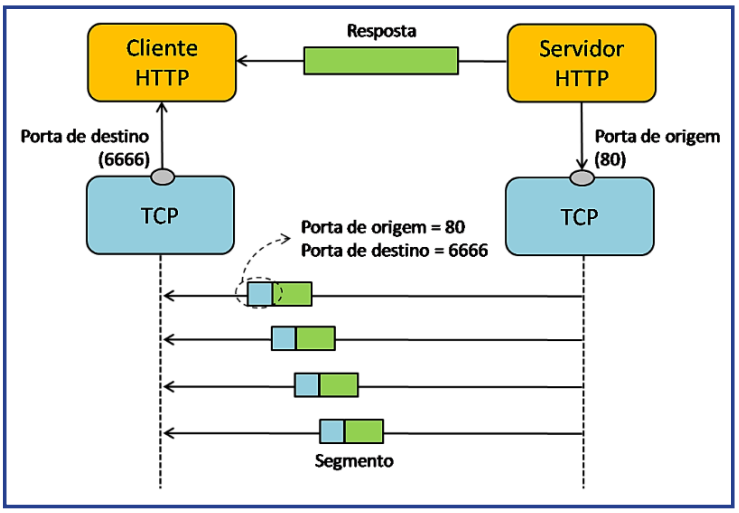
Figura 12: HTTP utilizando o TCP para envio de uma solicitação



A Figura 13 ilustra o caminho inverso, isto é, o envio da resposta HTTP. Embora o servidor HTTP tenha repassado integralmente aquela resposta, o TCP a segmenta em várias partes – vem daí o fato de a PDU de transporte ser denominada segmento. Os motivos para esta segmentação serão compreendidos mais adiante. Quando se estuda unicamente o HTTP, abstraindo os demais níveis, é comum considerar-se que a resposta é transmitida integralmente para o cliente.

Na prática, porém, a resposta HTTP chega ao cliente transportada em vários segmentos TCP. Cada um deles possui um cabeçalho que contém, dentre outras informações, a porta de origem (80) e a porta de destino – no caso, à porta a qual o cliente associou-se temporariamente, 6666.

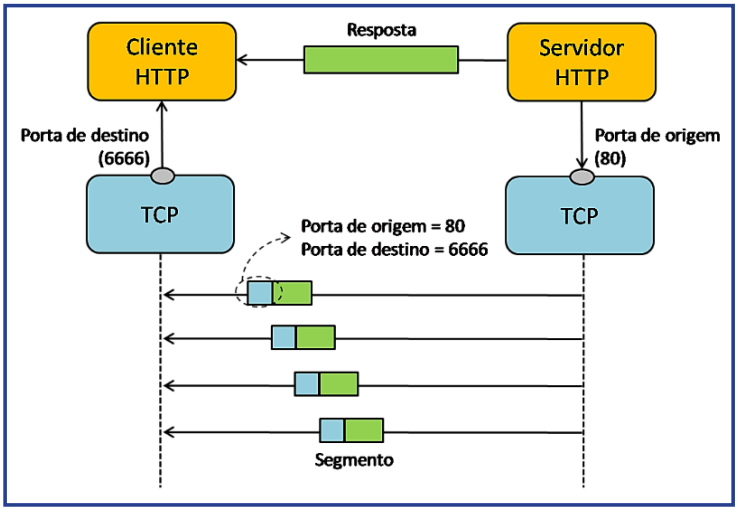
Figura 13: Envio de uma resposta HTTP em vários segmentos TCP



A troca de solicitações e resposta HTTP acontece durante o que é conhecido como conexão TCP. Mais adiante, será visto como uma conexão TCP proporciona um serviço de entrega confiável para os protocolos do nível de aplicação. Por ora, há de se considerar apenas que uma conexão TCP deve ser aberta antes que haja a transmissão de PDUs de aplicação e fechada após todas as PDUs terem sido transmitidas.

Os procedimentos de abertura e fechamento de conexão envolvem trocas de segmentos TCP específicos, o que demanda certo tempo. Dito isto, a Figura 14 ilustra a diferença entre as duas versões do HTTP, denominadas HTTP 1.0 e HTTP 1.1. Considera-se a obtenção do objeto ilustrado pela Figura 4 – isto é, um arquivo HTML que referencia diretamente três imagens JPEG.

Figura 14: Diferença entre as versões do HTTP



As versões do HTTP possuem o mesmo formato de solicitação e resposta, mudando apenas a forma como utilizam o TCP. A versão original do HTTP, denominada HTTP 1.0 e especificada pela RFC 1945, define que cada cópia de objeto deve ser transferida em uma conexão TCP transiente. No caso ilustrado pela Figura 14, é necessário abrir quatro conexões TCP: a primeira delas para transferir uma cópia do arquivo HTML e outras três para transferir cópias das imagens JPEG referenciadas pelo arquivo.

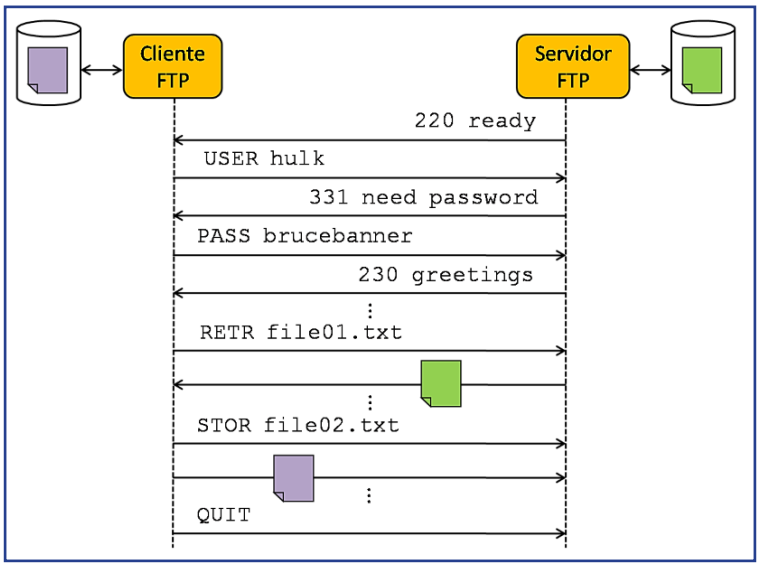
Por outro lado, o HTTP 1.1, especificado pela RFC 2616, define que todas as cópias de objetos devem ser transmitidas dentro de uma única conexão TCP persistente. Observando a Figura 14, conclui-se que uma única conexão TCP persistente, com apenas uma abertura e fechamento, leva a uma transmissão mais rápida.

FTP

Conforme dito anteriormente, o conceito da web surgiu bem depois da internet. Dentre os serviços suportados inicialmente pela internet, estava a transferência de (cópias de) arquivos de um computador para outro. Para tal, foi especificado na RFC 959 o FTP (File Transfer Protocol). Semelhantemente ao HTTP, o FTP é um protocolo cliente/servidor que opera em modo-texto e utiliza os serviços prestados pelo TCP, mas as semelhanças param por aí. Conforme ilustra a Figura 15, uma sessão FTP é caracterizada pela presença de várias interações entre cliente e servidor por meio do envio de comandos e respostas.

As interações iniciais servem para identificação do cliente por meio dos comandos USER e PASS, de forma que o FTP é considerado um protocolo *stateful* – em oposição ao HTTP, que é um protocolo *stateless* por definição. Em seguida, o cliente faz o *download* do arquivo file01.txt utilizando o comando RETR e, logo mais, faz o *upload* de file02.txt utilizando o comando STOR. Por fim, ele encerra a sessão FTP enviando o comando QUIT.

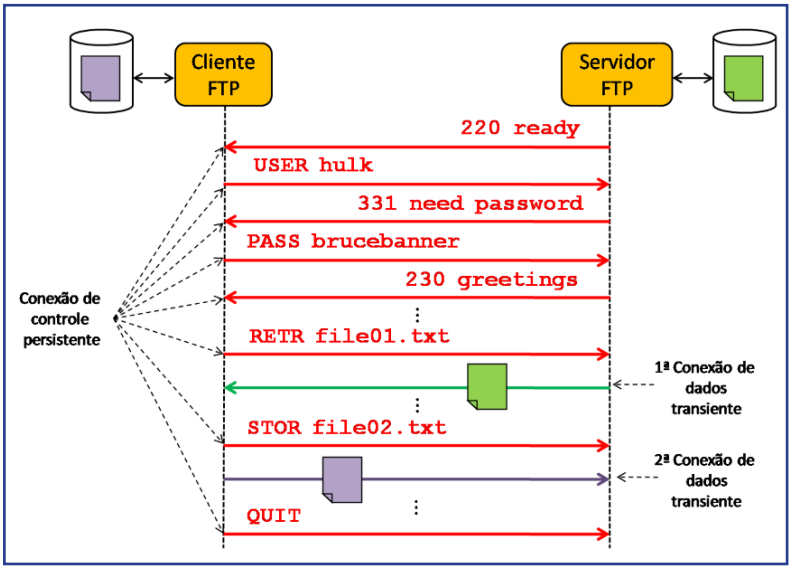
Figura 15: Comandos e respostas em uma sessão *FTP*



A forma como o FTP utiliza os serviços prestados pelo TCP é muito diferente daquela descrita para o HTTP. Uma sessão FTP possui uma única conexão TCP persistente, denominada conexão de controle, por meio da qual todos os comandos são enviados – servidores FTP são permanentemente associados à porta TCP de número 21 para a abertura dessa conexão. Por outro lado, os arquivos são transferidos através de conexões TCP transientes, denominadas conexões de dados.

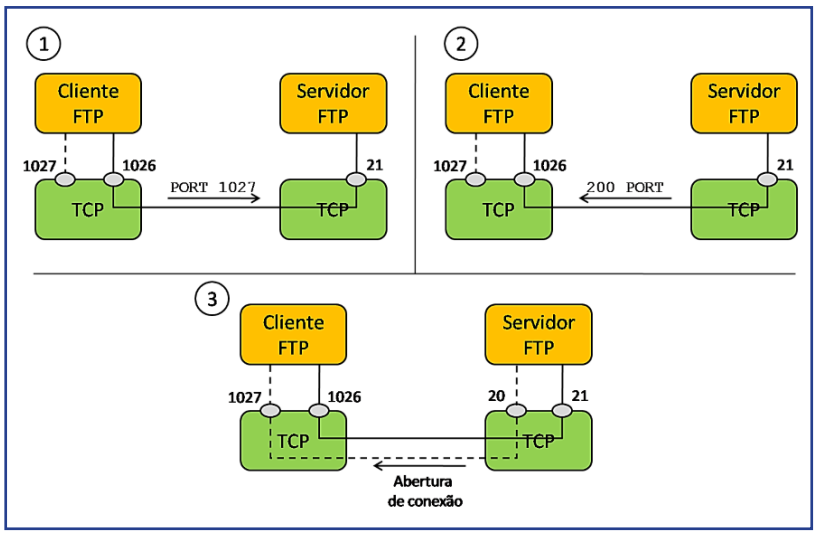
A Figura 16 ilustra a mesma sessão ilustrada pela Figura 15, neste caso destacando a conexão TCP de controle persistente na cor vermelha e as (duas) conexões de dados transientes na cor verde.

Figura 16: A mesma sessão FTP, com as conexões TCP *diferenciadas*



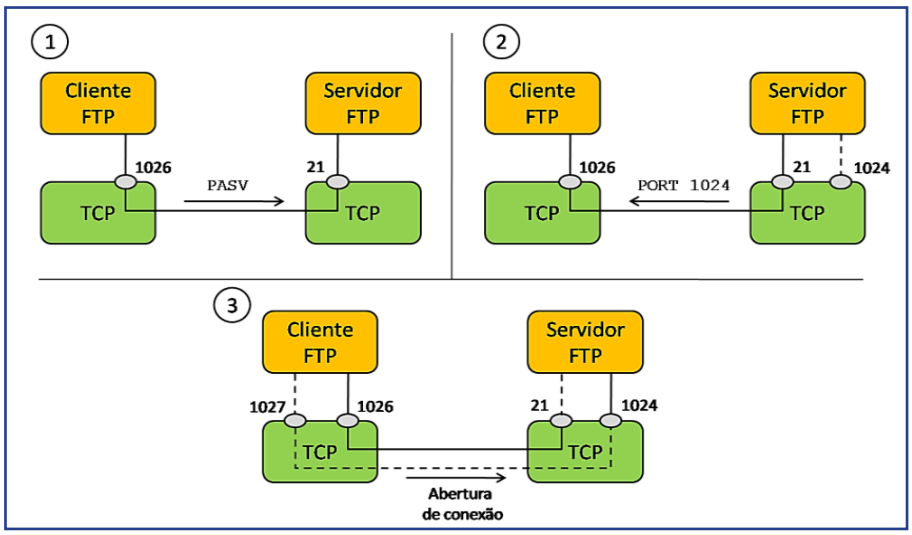
A forma como o servidor FTP se comporta com relação à abertura de uma conexão de dados permite classificar a operação do FTP em um de dois modos distintos, denominados ativo e passivo. No modo ativo, o servidor FTP abre a conexão de dados. A Figura 17 ilustra um exemplo em que o FTP opera no modo ativo. Nela, o cliente envia um comando PORT informando ao servidor que ele está associado à porta 1027, aguardando uma solicitação de abertura de conexão de dados. O servidor responde àquele comando e associa-se à porta 21 para abrir a conexão de dados, que servirá para transferência de um arquivo entre as partes.

Figura 17: Exemplo de operação do FTP no modo *ativo*



A Figura 18 ilustra um exemplo no qual o FTP opera no modo passivo, solicitado pelo cliente por meio do comando PASV. O servidor responde com um comando PORT que informa qual é à porta a qual o servidor está associado para receber uma solicitação de abertura de conexão de dados – no caso, a porta 1024. Em seguida, o cliente associa-se à porta 1027 e solicita a abertura da conexão de dados com o servidor para a transferência de um arquivo entre eles.

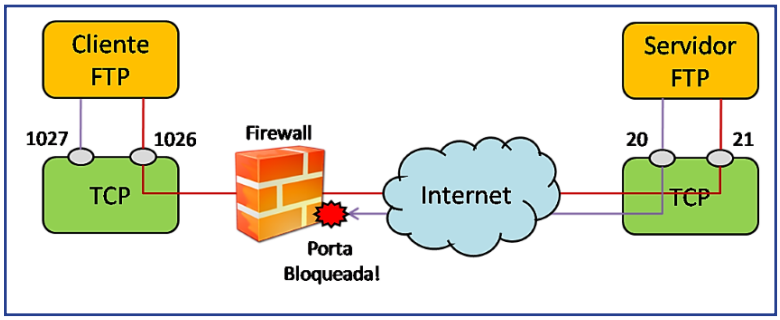
Figura 18: Exemplo de operação do FTP no modo passivo



O FTP foi projetado numa época na qual não era comum a presença de *firewalls.* Estes equipamentos bloqueiam pacotes com base em alguns parâmetros – dentre eles, portas em segmentos TCP. A Figura 19 ilustra o que ocorre quando há um *firewall* na rede do cliente. Com o FTP operando em modo ativo, apesar de o cliente ter informado ao servidor que está associado à porta 1027 para a conexão de dados, o servidor não consegue abrir aquela conexão porque o *firewall* bloqueia a tentativa.

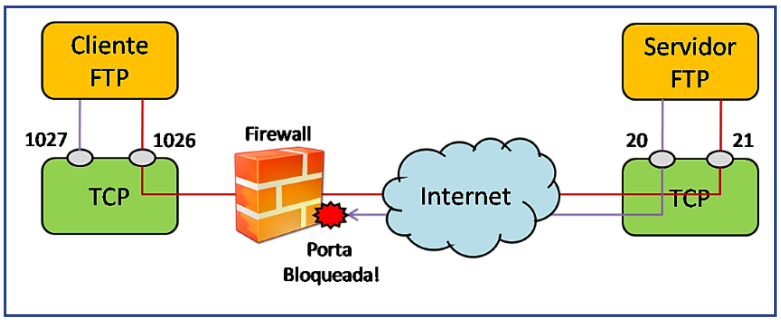
O *firewall* é tipicamente configurado para permitir a abertura de conexão com apenas determinadas portas – tipicamente, aquelas associadas a serviços bem conhecidos. A porta 1027 é apenas uma dentre as várias que estão bloqueadas pelo *firewall.* É possível alterar as configurações do firewall para desbloqueá-la, mas é difícil prever quais e quando potenciais clientes FTP utilizarão o serviço e a que portas eles irão se associar.

Figura 19: *Firewall* na rede do cliente FTP operando em modo ativo



A Figura 20 ilustra o que ocorre quando há um *firewall* na rede do servidor. Com o FTP operando em modo passivo, mesmo que o servidor tenha informado ao cliente que está associado à porta 1024 para abertura da conexão de dados, o cliente não consegue abrir aquela conexão porque o *firewall* bloqueia a tentativa. Neste caso, porém, é plausível configurar o *firewall* para permitir que uma conexão seja aberta com uma faixa de portas na máquina que executa o servidor FTP. O tamanho desta faixa vai depender da quantidade de conexões de dados simultâneas que se deseja permitir.

Figura 20: *Firewall* na rede do servidor FTP operando em modo passivo



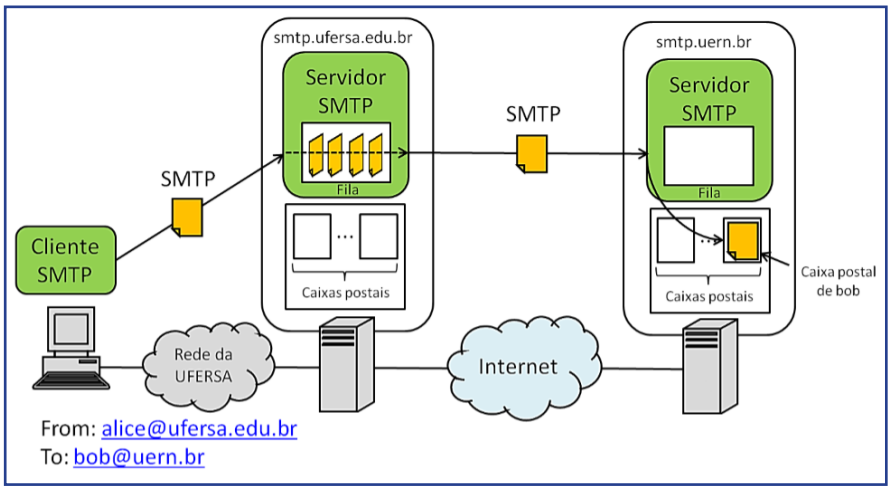
Outro problema relacionado ao FTP é que as credenciais de segurança do usuário limitam-se a um *login* e uma senha que são transmitidos sem nenhuma criptografia. Portanto, qualquer aplicação do tipo *sniffer* de pacotes pode obter aquelas informações e agir como um usuário legítimo do FTP. Por esta razão, o FTP não é mais utilizado na transferência sigilosa de arquivos. Naturalmente, há vários servidores FTP ao longo da internet que não se preocupam com isso por armazenarem arquivos que devam ser de fato disponibilizados a qualquer um que tenha interesse neles. Eles são chamados de servidores de FTP anônimos, por não exigirem credenciais dos usuários.

Correio Eletrônico

O correio eletrônico é um dos serviços iniciais da internet – bem antes do surgimento do conceito da Web, os usuários trocavam *emails* que continham somente texto codificado em ASCII (*American Standard Code for Information Interchange*). Arquitetonicamente, o correio eletrônico é também baseado no paradigma cliente/servidor. Clientes recebem genericamente a denominação de UAs (*User Agents*) e servidores recebem a denominação de MTAs (*Message Transfer Agents*). Os UAs são os componentes nos quais os usuários confeccionam e leem as mensagens. Os primeiros UAs eram aplicações de modo-texto, sem interface gráfica. Um exemplo de UA deste tipo era o *pine*, desenvolvido pela Universidade de Washington. Mais tarde, surgiram UAs com interface gráfica, como o Outlook da Microsoft e o Thunderbird, da Mozilla. Atualmente, o *browser* é o UA mais comum, haja vista que a maioria das pessoas utiliza o serviço de *WebMail*.

Na Figura 21, suponha que Alice (alice@ufersa.edu.br) deseja enviar uma mensagem para Bob (bob@uern.br). Admita que ela esteja usando como UA a versão clássica do Outlook, por exemplo. Quando ela clica no botão enviar, uma cópia da mensagem construída é transferida para o servidor local através do SMTP (*Simple Mail Transfer Protocol*). O SMTP é especificado pela RFC 821, e utiliza os serviços prestados pelo TCP – mais especificamente, servidores SMTP são associados à porta número 25 do TCP. O servidor SMTP local de Alice é o servidor SMTP da UFERSA, identificado por smtp.ufersa.edu.br – deve-se ter em mente que esta informação deve ser inserida manualmente na configuração do Outlook. Naquele servidor, a mensagem é posta em uma fila para ser enviada ao servidor SMTP do destinatário. Haja vista que a mensagem é endereçada a Bob, que possui uma conta de *email* na UERN, a mensagem é enviada ao servidor SMTP daquele domínio – que é identificado por smtp.uern.br. Ao chegar no servidor SMTP da UERN, a mensagem é posta na caixa postal de Bob.

Figura 21: Utilização do SMTP para envio da mensagem



A Figura 22 ilustra a troca de mensagens entre os servidores SMTP de Alice (A) e de Bob (B). Semelhantemente ao HTTP, o SMTP é um protocolo modo-texto no qual as respostas a comandos contêm um código de status e uma frase correspondente. Na figura, percebe-se facilmente que a mensagem é transmitida por partes – o formato da mensagem será detalhado logo adiante.

Figura 22: Troca de mensagens entre os servidores SMTP de Alice (A) e de Bob (B)

B: 220 SMTP server ready

A: HELO ufersa.edu.br

B: 250 Hello ufersa.edu.br

A: MAIL FROM:<alice@ufersa.edu.br>

B: 250 OK

A: RCPT TO:<bob@uern.br>

B: 250 OK

A: DATA

B: 354 Start mail input; end with "."

A: Subject: Informacao

A: Olah, Bob!

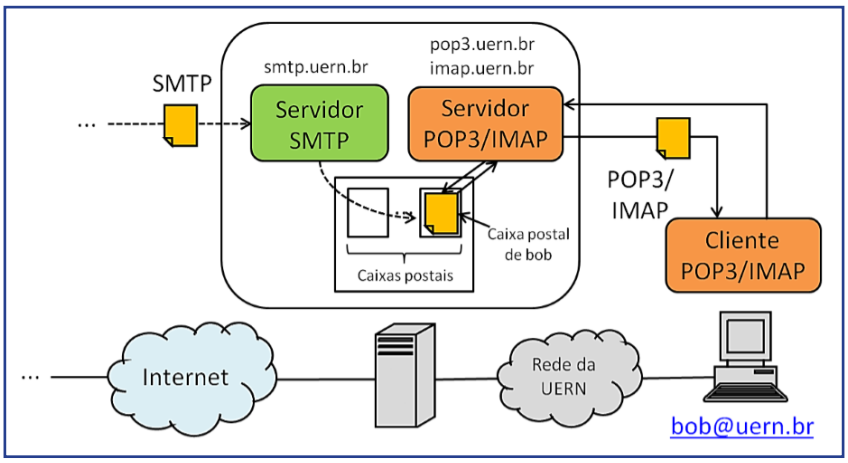
A: Tudo bem contigo?

A: .

B: 250 OK

É comum afirmar que o SMTP é um protocolo utilizado para envio de mensagens. Isto porque há outros protocolos utilizados por um UA para obter mensagens armazenadas em uma caixa postal. Estes protocolos são ditos operarem na entrega final de mensagens. Assim, supondo que Bob também esteja usando como UA a versão clássica do Outlook, ele pode ler suas mensagens utilizando o POP3 ou o IMAP. A Figura 23 ilustra este procedimento.

Figura 23: POP3 ou IMAP como protocolos para entrega final da mensagem



O POP3 (Post Office Protocol – Version 3), especificado pela RFC 1939, é um usuário dos serviços prestados pelo TCP – mais especificamente, servidores POP3 são associados à porta 110 do TCP. Uma sessão POP3 possui três fases, denominadas autorização, transações e atualizações. A Figura 24 ilustra uma troca de mensagens entre o cliente (C) e o servidor (S) POP3 de Bob – o servidor POP3 de Bob é identificado na Figura 23 como pop3.uern.br. A fase de autorização corresponde à passagem do login e da senha de Bob para o servidor, através dos comandos USER e PASS. A fase de transações corresponde às operações realizadas por Bob – a listagem das mensagens (STAT e LIST), a obtenção de uma delas (RETR) e a sua remoção (DELE). A fase de atualizações corresponde à efetivação das operações realizadas na fase anterior – a remoção da mensagem de fato só ocorre após o comando QUIT. O comportamento padrão do POP3 é transferir todas as mensagens para o computador do usuário no início da sessão, esvaziando sua caixa postal. Há uma opção para se deixar armazenada uma cópia de cada mensagem no servidor POP3, mas ela raramente está habilitada.

Figura 24: Troca de mensagens entre cliente e servidor POP3 de Bob

```
S:  +OK POP3 server ready
C:  USER bob
S:  +OK user name accepted, password please
C:  PASS mysecretpassword
S:  +OK bob's inbox has 1 message (120 octets)
C:  STAT
S:  +OK 1 120
C:  LIST
S:  +OK 1 message (120 octets)
S:  1 120
S:  .
C:  RETR 1
S:  +OK 120 octets
```

```
S:   Received: from smtp.ufersa.edu.br
     by smtp.uern.br
     To: bob@uern.br
     From: alice@ufersa.edu.br
     Subject: Informacao
     Olah, Bob!
     Tudo bem contigo?
     .
C:   DELE 1
S:   +OK message 1 deleted
C:   QUIT
S:   +OK POP3 server signing off
```

Um protocolo alternativo ao POP3 é o IMAP (Internet Message Access Protocol), especificado pela RFC 2060. Ele também é um usuário dos serviços prestados pelo TCP – servidores IMAP são associados à porta 143 do TCP. A Figura 25 ilustra uma troca de mensagens entre o cliente (C) e o servidor (S) IMAP de Bob – o servidor IMAP de Bob é identificado na figura 23 como imap.uern.br. Diferentemente do POP3, o IMAP descarrega apenas o cabeçalho das mensagens no início da sessão – utilizando o comando FETCH ... BODY[HEADER}. O usuário pode então solicitar posteriormente o descarregamento do corpo de uma delas com comandos como FETCH ... BODY[TEXT]. Outra diferença que o IMAP guarda com o POP3 é a manutenção de cópias de cada mensagem no servidor. O IMAP é considerado um protocolo muito mais flexível – e complexo – do que o POP3.

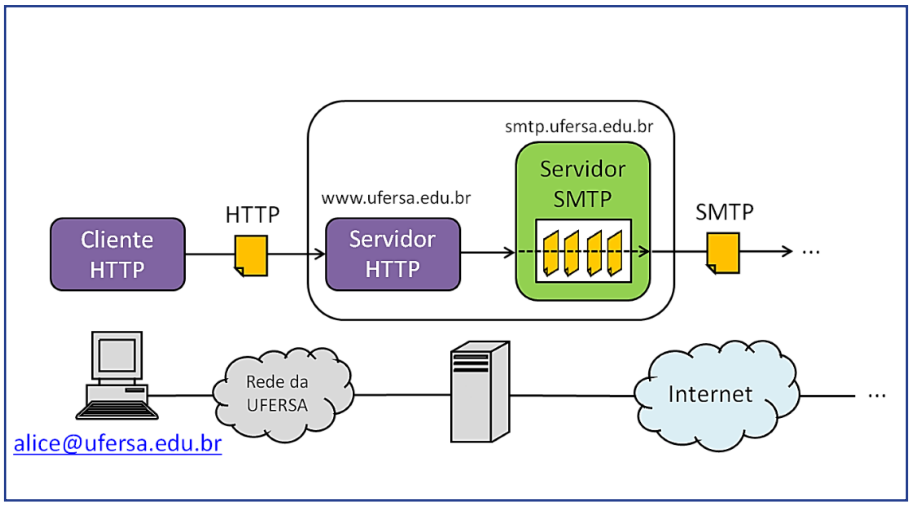
Figura 25: Troca de mensagens entre cliente e servidor IMAP de Bob

```
S:   OK IMAP server ready
C:   LOGIN bob mysecretpassword
S:   OK User bob authenticated
C:   SELECT Inbox
S:   1 EXISTS
     0 RECENT
     SELECT completed
C:   FETCH 1 BODY[HEADER]
S:   1 FETCH (BODY[HEADER]
     Received: from smtp.ufersa.edu.br
     by smtp.uern.br
     To: bob@uern.br
     From: alice@ufersa.edu.br
     Subject: Informacao
     OK FETCH completed
```

```
C:  FETCH 1 BODY[TEXT]
S:  1 FETCH (BODY[TEXT]
    Olah, Bob!
    Tudo bem contigo?
    OK FETCH completed
C:  LOGOUT
S:  BYE IMAP server terminating connection
    OK LOGOUT completed
```

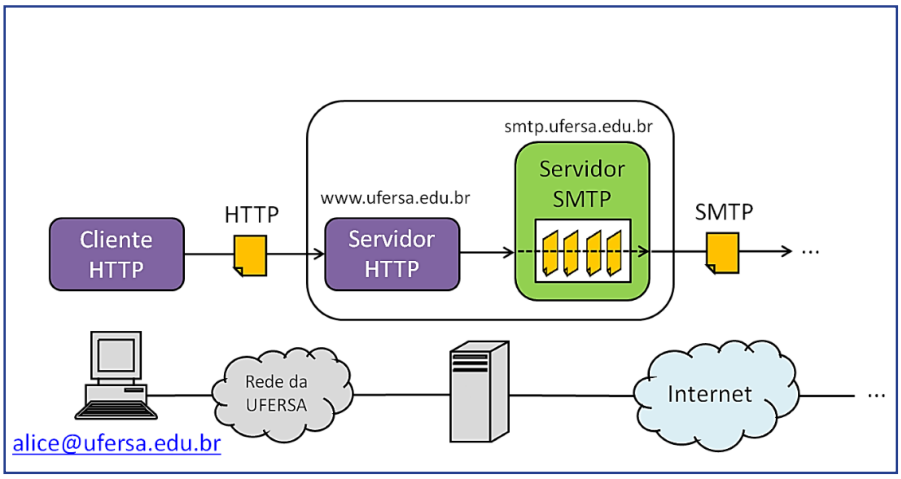
A Figura 26 ilustra o que ocorre quando Alice utiliza um *browser* como UA para enviar a mensagem de correio eletrônico para Bob – após tê-la confeccionado utilizando aquela aplicação, naturalmente. Conforme pode ser observado, a mensagem é transmitida inicialmente para o servidor HTTP que, por sua vez, repassa a mensagem para o servidor SMTP da UFERSA. Assim, embora a mensagem seja transmitida inicialmente utilizando o HTTP, ela continua sendo transmitida (em última instância) utilizando o SMTP.

Figura 26: Remetente utilizando *browser* como UA



A Figura 27 ilustra o que ocorre quando Bob utiliza um *browser* como UA para obter a mensagem de correio eletrônico enviada por Alice – e lê-la utilizando aquela aplicação, naturalmente. Conforme pode ser observado, o servidor HTTP da UERN tem acesso à caixa postal de Bob, onde a mensagem foi previamente inserida pelo servidor SMTP. Neste contexto, a entrega final da mensagem é integralmente feita utilizando o HTTP, não mais havendo a presença dos protocolos POP3 ou IMAP.

Figura 27: Destinatário utilizando o *browser* como UA



O formato básico de uma mensagem de correio eletrônico é definido pela RFC 822. Como pode ser visto na Figura 28, ela requer no mínimo três campos: From, To e Subject. Eles identificam, respectivamente, o remetente, o destinatário e o assunto da mensagem.

Figura 28: Formato da mensagem de correio eletrônico

From: alice@ufersa.edu.br

To: bob@uern.br

Subject: Quadro de Monet.

A RFC 822 previa que as mensagens seriam compostas apenas de texto codificado em ASCII. Isto proíbe a inserção de anexos ou mensagens cujo texto contivesse caracteres acentuados. Para contornar esta situação, foram definidos nas RFCs 2045 e 2046 os cabeçalhos MIME (*Multi purpose Internet Mail Extensions*). A Figura 29 ilustra um exemplo de utilização de cabeçalhos MIME para inserção de uma imagem JPEG em uma mensagem de correio eletrônico. O cabeçalho *Content-Type* define se o conteúdo inserido é uma imagem JPEG (como é o caso ilustrado), uma imagem GIF, um texto codificado em outra forma (que não o ASCII), um vídeo em MPEG, etc.

Figura 29: Exemplo de utilização de cabeçalho MIME

From: alice@ufersa.edu.br

To: bob@uern.br

Subject: Quadro de Monet.

MIME-Version: 1.0

Content-Transfer-Encoding: base64

Content-Type: image/jpeg

(dados codificados em base64....

.....

...(dados codificados em base64

A Figura 30 ilustra agora uma mensagem composta por três partes: um texto codificado em ASCII, uma imagem JPEG e outro texto codificado em ASCII. Neste caso, o valor do cabeçalho *Content-Type* é *multipart/mixed*, e cada uma das partes é delimitada pelo valor do campo *Boundary.* Na figura, o valor deste campo é *StartOfNextPart*.

Figura 30: Uma mensagem formada por várias partes

From: alice@ufersa.edu.br
To: bob@uern.br
Subject: Quadro de Monet.
MIME-Version: 1.0
Content-Type: multipart/mixed;
Boundary=StartOfNextPart
--StartOfNextPart
Bob, eis aqui o quadro de Monet.
--StartOfNextPart
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
(dados codificados em base64....
.....
...(dados codificados em base64)
--StartOfNextPart
Espero que ajude em seu trabalho.
.

DNS

A comunicação de dados por meio de uma rede requer conhecimento de endereços IP associados aos computadores remotos. Senão, vejamos:

• Como um *browser* sabe qual é a máquina que hospeda o servidor web da UFERSA (www.ufersa.edu.br)?

• Como um UA sabe quais são as máquinas que hospedam os servidores SMTP e POP3 da UFERSA (smtp.ufersa.edu.br e pop.ufersa.edu.br, respectivamente)?

• Como o servidor SMTP da UFERSA sabe qual é a máquina que hospeda o servidor SMTP do domínio uern.br?

O papel do DNS (*Domain Name System*), especificado pelas RFCs 1034 e 1035, é resolver esses nomes – isto é, descobrir quais são os endereços IP associados a eles. O DNS é um protocolo cliente/servidor que consiste no envio de uma solicitação (também denominada *query*) e na recepção de uma resposta. Diferentemente de todos os protocolos estudados até agora, o DNS faz uso do UDP – servidores DNS são associados à porta UDP de número 53.

O motivo pelo qual o DNS utiliza o UDP é que a resolução de nomes deve ser um processo extremamente rápido. Ela deve preceder o envio de solicitações HTTP, comandos FTP e mensagens de correio eletrônico. E o fato é que o estabelecimento, manutenção e encerramento de uma conexão TCP demandam tempo.

A figura 31 ilustra como domínios são estruturados hierarquicamente. Partindo do domínio raiz (.), seguem-se os TLDs (*Top Level Domain*) – normalmente, associados a países (au – Austrália, br – Brasil, fr – França, jp – Japão, etc). A partir deles, seguem os demais domínios em hierarquia de vários níveis.

Há vários servidores responsáveis pelo domínio raiz. Os denominados servidores raiz contêm o mapeamento para os servidores dos TLD – isto é, eles sabem quais são os endereços IP daqueles servidores. Treze organizações são responsáveis por estes servidores. A figura 32 ilustra onde estão localizadas estas organizações. Cada uma delas mantém um conjunto de servidores raiz, e cada conjunto é representado por um único endereço IP, usando técnicas de espelhamento.

Os componentes de um conjunto estão espalhados ao redor do globo, e a maioria dos servidores raiz está localizada fora dos EUA. Em http://www.root-servers.org/, é possível saber quais são os servidores raiz mais próximos de uma determinada localidade.

Figura 31: Estrutura hierárquica de domínios

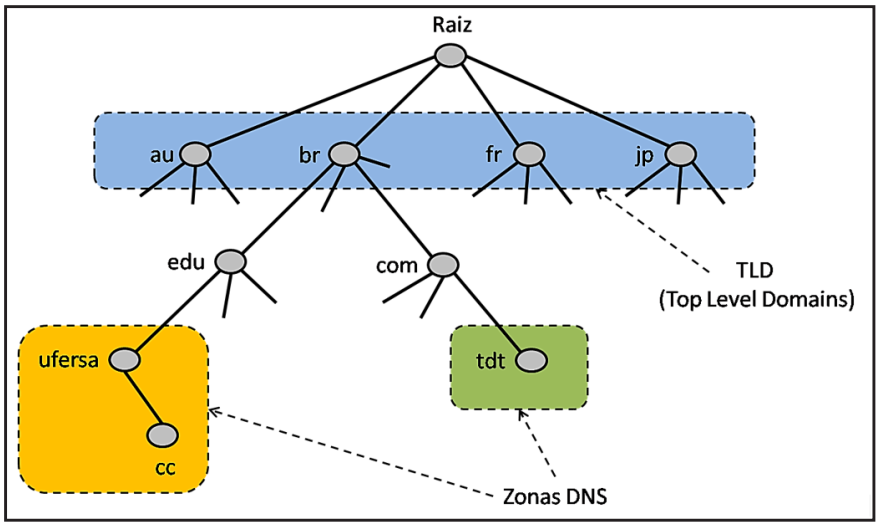
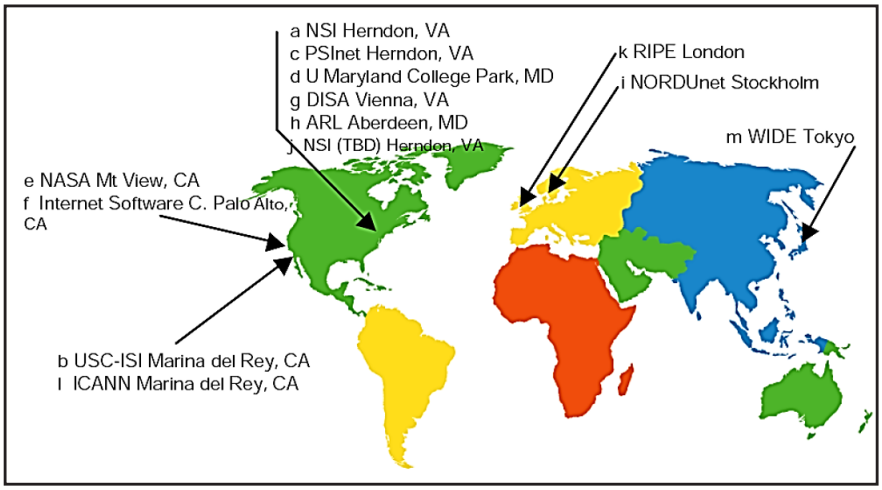


Figura 32: Organizações responsáveis pelos servidores raiz



Uma resolução DNS pode ser feita de várias formas. A Figura 33 ilustra uma destas formas. Supõe-se que um cliente HTTP presente na rede da UFERSA queira acessar a página web da loja virtual fictícia "Tubarões de Tibau" – www.tdt.com.br. Como o cliente HTTP não sabe qual é o computador que executa o servidor HTTP correspondente, ele pergunta seu cliente DNS instalado no mesmo computador (1).

Este cliente envia a solicitação para o servidor DNS da UFERSA (2), também denominado de servidor DNS local – cada computador é configurado com o endereço de seu servidor DNS local. Cada servidor DNS local é equipado com uma lista de servidores raiz, cujos endereços são listados na Tabela 3.

Figura 33: Resolução DNS com um mix de procedimentos recursivos e iterativos

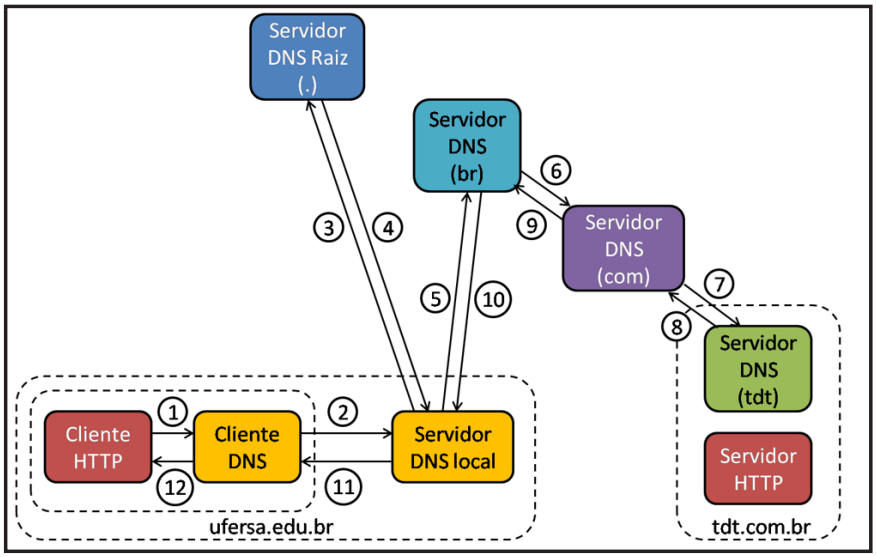


Tabela 3: Lista de endereços IP de servidores raiz

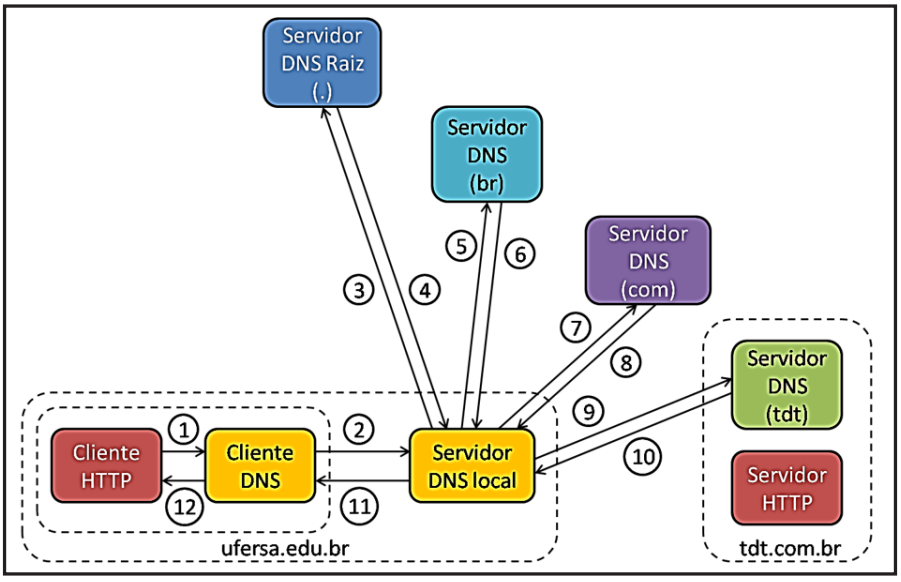
Nome Canônico	Endereço IP	Organização responsável
a.root-servers.net	198.41.0.4	VeriSign, Inc.
b.root-servers.net	192.228.79.201	University of Southern California (ISI)
c.root-servers.net	192.33.4.12	Cogent Communications
d.root-servers.net	199.7.91.13	University of Maryland
e.root-servers.net	192.203.230.10	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4	US Department of Defence (NIC)
h.root-servers.net	128.63.2.53	US Army (Research Lab)
i.root-servers.net	192.36.148.17	Netnod
j.root-servers.net	192.58.128.30	VeriSign, Inc.
k.root-servers.net	193.0.14.129	RIPE NCC
l.root-servers.net	199.7.83.42	ICANN
m.root-servers.net	202.12.27.33	WIDE Project

O servidor DNS local consulta um dos servidores raiz e recebe como resposta o endereço IP do servidor do TLD br (3 e 4) – neste tipo de procedimento, denominado iterativo, o servidor consultado fornece uma resposta imediata. De posse do endereço IP, o servidor DNS local consulta o servidor do domínio br (5), que verifica no nome a ser resolvido (www.tdt.com.br) que o próximo domínio subordinado a ele é o domínio com.br. Ele então repassa a consulta àquele servidor (6), em um procedimento denominado recursivo.

O servidor DNS do domínio com.br executa a mesma lógica do servidor DNS do domínio br e repassa recursivamente a solicitação ao servidor do domínio tdt.com.br (7). O servidor DNS do domínio tdt.com.br é considerado o servidor DNS oficial daquele domínio, porque conhece o endereço IP da máquina que hospeda o servidor HTTP do *site* da loja. O servidor DNS oficial retorna a resposta para o servidor do domínio com.br (8), que a retorna para o servidor DNS do domínio br (9), que a retorna para o servidor DNS local da UFERSA (10), que a retorna para o cliente DNS local (11), que informa o endereço IP para o cliente HTTP (12).

Um servidor raiz tem sempre o comportamento interativo. Ele sempre fornece o endereço IP de um TLD. Os demais servidores ao longo do caminho até o servidor oficial podem comportar-se de forma interativa ou recursiva. Muitas vezes, eles são configurados de forma interativa para que não percam tempo esperando que respostas recursivas guardem informações de estado acerca das resoluções em curso. Assim, a Figura 34 ilustra outra forma de resolver o mesmo endereço, desta vez utilizando procedimentos puramente interativos.

Figura 34: Resolução DNS com procedimentos puramente iterativos



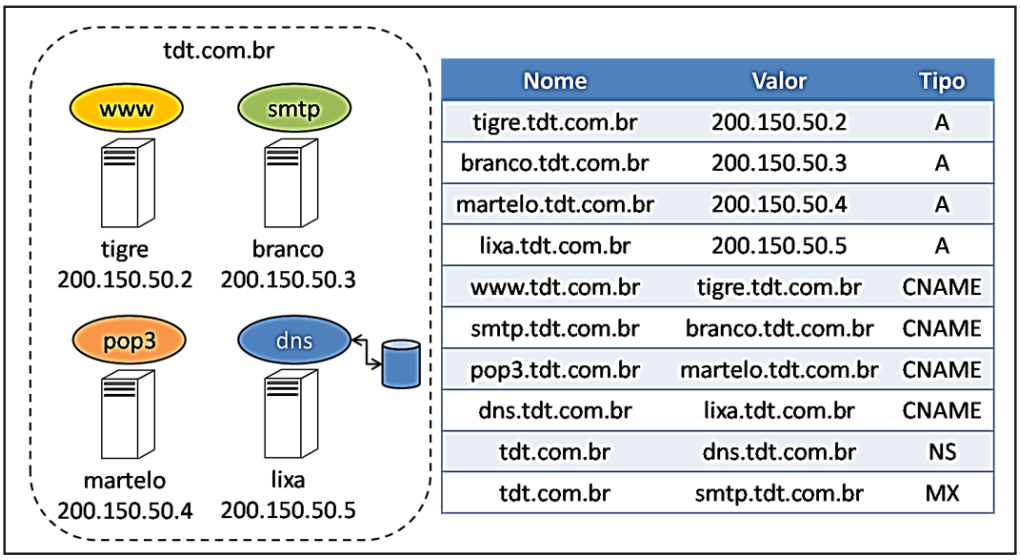
A base de dados mantida por um servidor DNS oficial é composta por Registros de Recurso (RRs). Um RR é uma estrutura de dados cujos principais campos são nome, valor e tipo. O que vem a estar nos campos nome e valor depende do tipo do RR. A Tabela 4 ilustra os principais tipos de RR e os significados dos demais campos em cada um deles.

Tabela 4: Tipos principais de RR e seus significados

Tipo	Nome	Valor
A (Address)	Nome canônico	Endereço IP
CNAME (Canonical Name)	Apelido	Nome canônico
NS (Name Server)	Domínio	Apelido do servidor DNS
MX (Mail eXchange)	Domínio	Apelido do servidor SMTP

Para entender a Tabela 4, a figura 35 ilustra a base de dados mantida pelo servidor DNS oficial do domínio fictício tdt.com.br. Destacam-se quatro computadores, cujos *hostnames* são tigre, branco, martelo e lixa. Seus nomes canônicos são nada mais do que a junção dos seus hostnames com o domínio – por exemplo, o nome canônico de tigre é tigre.tdt.com.br. Cada computador possui um apelido que reflete o serviço associado a ele. Tomando mais uma vez o computador cujo *hostname* é tigre, seu apelido é www – que evidencia que ele executa um servidor HTTP. Similarmente, os apelidos de branco, martelo e tigre são smtp.tdt.com.br, pop3.tdt.com.br e dns.tdt.com.br, respectivamente. Isto indica que aqueles computadores executam um servidor SMTP, um servidor POP3 e o servidor oficial do domínio, respectivamente.

Figura 35: Exemplo de banco de dados mantido por um servidor DNS oficial



Nome	Valor	Tipo
tigre.tdt.com.br	200.150.50.2	A
branco.tdt.com.br	200.150.50.3	A
martelo.tdt.com.br	200.150.50.4	A
lixa.tdt.com.br	200.150.50.5	A
www.tdt.com.br	tigre.tdt.com.br	CNAME
smtp.tdt.com.br	branco.tdt.com.br	CNAME
pop3.tdt.com.br	martelo.tdt.com.br	CNAME
dns.tdt.com.br	lixa.tdt.com.br	CNAME
tdt.com.br	dns.tdt.com.br	NS
tdt.com.br	smtp.tdt.com.br	MX

Na Figura 35, os quatro primeiros RRs são do tipo A, e associam os nomes canônicos dos computadores aos seus respectivos endereços IP. Em seguida, há quatro RRs do tipo CNAME, que associam apelidos aos nomes canônicos. O penúltimo RR é do tipo NS, que associa o domínio tdt.com.br ao apelido do servidor DNS. O último RR é do tipo MX, e associa aquele domínio ao seu servidor SMTP. Portanto, quando o cliente DNS realiza uma consulta para saber qual é o endereço IP correspondente a www.tdt.com.br, ele envia o RR incompleto (www.tdt.com.br, CNAME). Ao final do processo, os seguintes RRs são recebidos por ele:

• O RR do tipo CNAME que informa que www.tdt.com.br é, na verdade, o apelido de tigre.tdt.com.br;

• O RR do tipo A que informa que tigre.tdt.com.br corresponde ao endereço IP 200.150.50.2;

Uma resposta a uma consulta DNS também contém dados acerca do servidor oficial. Assim, também são retornados os seguintes RRs:

• O RR do tipo NS que informa que o servidor oficial do domínio tdt.com.br é dns.tdt.com.br;

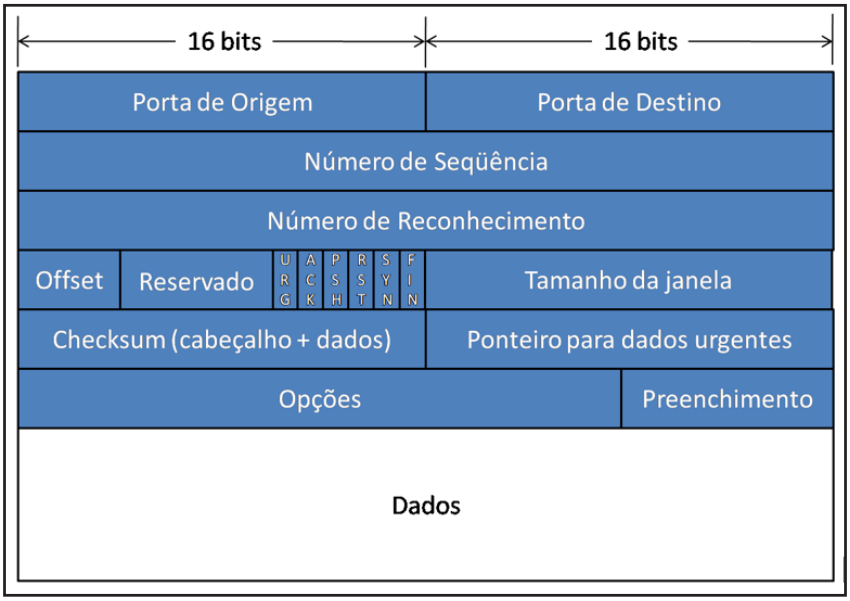
• O RR do tipo CNAME que informa que dns.tdt.com.br é, na verdade, o apelido de lixa.tdt.com.br;

• O RR do tipo A que informa que lixa.tdt.com.br corresponde ao endereço IP 200.150.50.5.

TCP & UDP

O TCP (*Transmission Control Protocol*) é especificado originalmente pela RFC 793, sendo um dos protocolos definidos para o nível de transporte. Ele é responsável por três importantes procedimentos: o controle de erros, o controle de fluxo e o controle de congestionamento. Ao realizá-los, o TCP provê um serviço confiável de entrega de PDUs do nível de aplicação – que é exatamente a função do nível de transporte. Para entender como ele realiza aqueles procedimentos, é necessário examinar o formato do segmento TCP, ilustrado pela Figura 36.

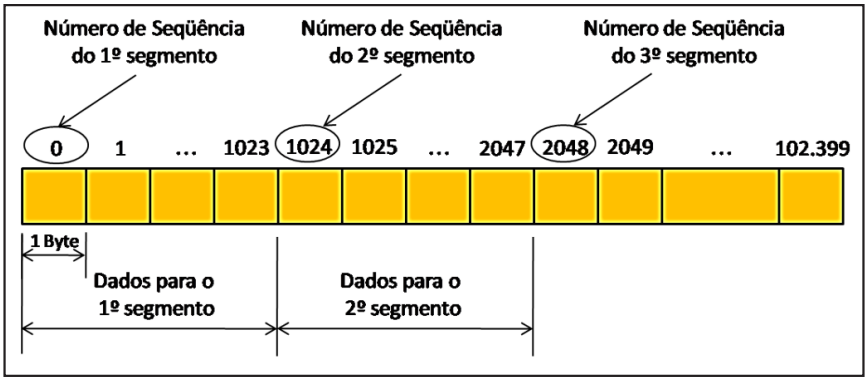
Figura 36: Formato do segmento TCP



As portas de origem e destino são campos que permitem identificar as aplicações usuárias do TCP – dentre as quais estão processos HTTP, FTP, SMTP, POP3, IMAP. Os 16 bits permitem endereçar 65.536 portas diferentes, sendo que as portas de número 0 a 1023 são denominadas portas reservadas, estando associadas a protocolos padronizados pela IETF e, por isto, bem conhecidos.

Os números de sequência e de reconhecimento são cruciais para que o TCP realize o controle de erros. Para entender este procedimento, há de se considerar o fato de que todos os dados enviados pelo nível de aplicação para o TCP são numerados por ele. A Figura 37 ilustra como uma cadeia de aproximadamente 100 KB de dados repassados pelo nível de aplicação é preparada pelo TCP para envio. Conforme se percebe, cada byte da cadeia recebe um número. Além disso, a transmissão é feita em partes, cada uma destas partes sendo transportada no campo de dados em um segmento. Por fim, o número de sequência de cada segmento é o número do primeiro byte de dados transmitido por ele.

Figura 37: Numeração dos dados enviados pelo nível de aplicação



A Figura 38 ilustra como os números de sequência e de reconhecimento são utilizados. A entidade TCP da esquerda transmite sua cadeia de bytes por partes, cada uma delas encapsulada em um segmento. O primeiro segmento contém os primeiros 1024 bytes, numerados de 0 a 1023. Como o número de sequência identifica o número do primeiro daqueles bytes, o número de sequência daquele segmento é igual a 0. O próximo segmento transmitido por aquela entidade TCP encapsula outros 1024 bytes, numerados de 1024 a 2047. Assim, seu número de sequência é igual a 1024. Utilizando a mesma lógica, é fácil concluir que um terceiro segmento, não ilustrado pela figura, teria 2048 como seu número de sequência.

O controle de erros provido pelo TCP baseia-se no fato de que, quando uma entidade TCP transmite um segmento, ele deve receber outro segmento que permite a ela saber que o segmento transmitido chegou ao seu destino, sem erros. Este outro segmento possui um número de reconhecimento que pode ser interpretado das seguintes formas:

• O número de reconhecimento informa qual é o número do próximo byte que uma entidade TCP espera que seu par envie;

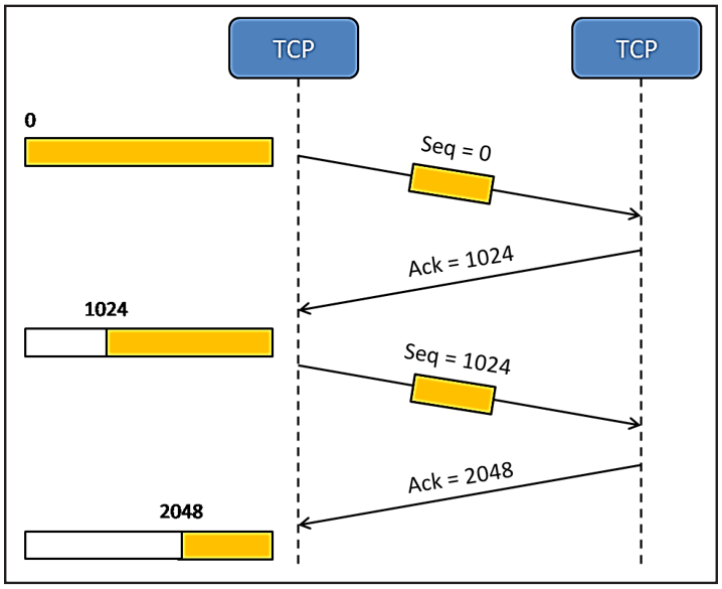
• O número de reconhecimento informa que todos os bytes na sequência que vai do início da cadeia até o byte anterior ao seu valor foram, corretamente, recebidos.

Portanto, o fato de que o primeiro segmento transmitido pela entidade TCP da direita possui número de reconhecimento igual a 1024 informa que este é o número do próximo byte esperado por aquela entidade, que acabou de receber os bytes de números 0 a 1023 da entidade TCP da esquerda. Também informa que a sequência definida pelos bytes de número 0 a 1023 foi corretamente transmitida. Neste sentido, o primeiro segmento transmitido pela entidade TCP da direita reconhece a recepção do primeiro segmento transmitido pela entidade TCP da esquerda.

Sempre que uma entidade TCP transmite um segmento encapsulando dados, ela mantém aqueles dados em seu *buffer* de transmissão até que receba o reconhecimento associado àquele segmento. Após receber o reconhecimento, a entidade pode descartar aqueles dados. É por isto que a entidade TCP da esquerda descarta os primeiros 1024 bytes da cadeia quando recebe o reconhecimento associado a eles.

O segundo segmento enviado pela entidade TCP da direita reconhece a recepção do segundo segmento enviado pela entidade TCP da esquerda. Seu número de reconhecimento possui valor igual a 2048, que informa que o número do próximo byte esperado pela entidade TCP da direita é 2048. Também informa que todos os bytes cujos números vão de 0 a 2047 foram recebidos sem erros. Ao receber este reconhecimento, a entidade TCP da esquerda descarta mais 1024 bytes na cadeia.

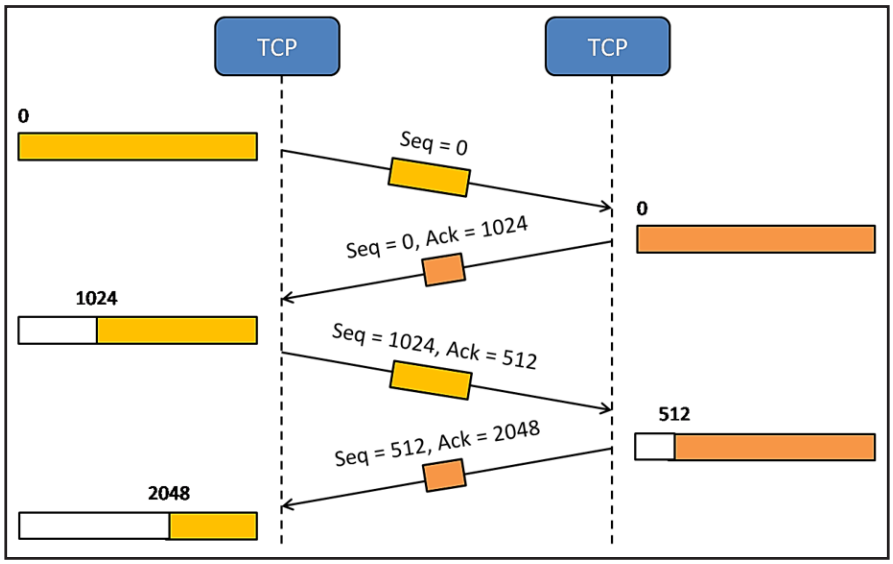
Figura 38: Utilização de números de sequência e de reconhecimento



O cenário ilustrado corresponde àquele no qual o fluxo de dados ocorre de maneira assimétrica, no qual um dos lados envia muito mais dados do que o outro. O descarregamento de objetos web através do HTTP e o *download ou upload* de um arquivo através do FTP são exemplos de tais cenários. Há outros, porém, nos quais o fluxo de dados ocorre de forma mais ou menos simétrica. Aplicações interativas, de troca de mensagens, são exemplos deste tipo de cenário alternativo. Neles, é comum a ocorrência do reconhecimento por carona (*piggyback acknowledgement*), a estratégia utilizada por uma entidade TCP tanto para transmitir dados de sua própria cadeia para uma entidade TCP remota quanto para reconhecer os dados transmitidos previamente por aquela entidade remota.

A figura 39 ilustra um exemplo de utilização desta estratégia. Ambas as entidades possuem agora cadeias de bytes a serem transmitidas para a outra. O primeiro segmento transmitido pela entidade TCP da direita tanto transmite os primeiros 512 bytes da sua cadeia quanto reconhece os 1024 bytes enviados previamente pela entidade TCP da esquerda. O segundo segmento transmitido pela entidade TCP da esquerda tanto transmite os próximos 1024 bytes da sua cadeia quanto reconhece aqueles 512 bytes. O segundo segmento enviado pela entidade TCP da direita tanto transmite os próximos 512 bytes de sua cadeia quanto reconhece o novo conjunto de 1024 bytes enviado pela entidade TCP da esquerda.

Figura 39: Exemplo de reconhecimento por carona



Uma vez entendido o significado dos números de sequência e reconhecimento, será descrito agora como eles são utilizados pelo TCP para prover o controle de erros. Em particular, cada segmento enviado tem um temporizador associado a ele, denominado temporizador de retransmissão. A regra geral é que, quando uma entidade transmite um segmento, ela aguarda um reconhecimento para aquele segmento até que o temporizador de retransmissão é expirado. Quando isto acontece, o segmento é retransmitido – vem daí a denominação do temporizador. A Figura 40 ilustra a perda do segmento original.

Naturalmente, se o segmento original é perdido, não existirá um reconhecimento associado a ele. Outro exemplo é ilustrado pela Figura 41, onde o segmento original é recebido, mas seu reconhecimento de fato é perdido. Em ambos os casos, pelo fato de o reconhecimento para o segmento original não chegar à entidade que o transmitiu, o temporizador de retransmissão expira, e o segmento é retransmitido.

Atualmente, a utilização de redes de computadores é comum na vida de qualquer ser humano. Qualquer pessoa, por exemplo, utiliza a internet para acessar informações e interagir com outras pessoas por meio de aplicações. Isto, naturalmente, faz com que haja uma maior necessidade de profissionais aptos a entender, projetar e administrar redes de computadores.

Para que dois computadores possam se comunicar através de uma rede, ambos devem utilizar um mesmo protocolo de comunicação. Um protocolo de comunicação é definido em termos de um conjunto de mensagens – denominadas PDUs (*Protocol Data Units*) – e de uma sequência na qual estas mensagens devem ser trocadas entre as aplicações.

Figura 40: Perda do segmento e expiração do temporizador de retransmissão

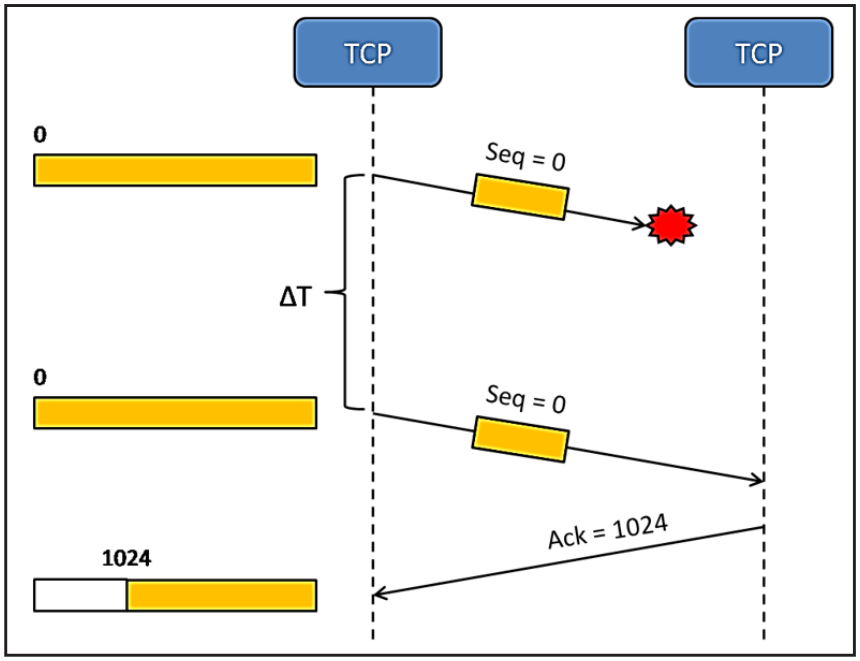
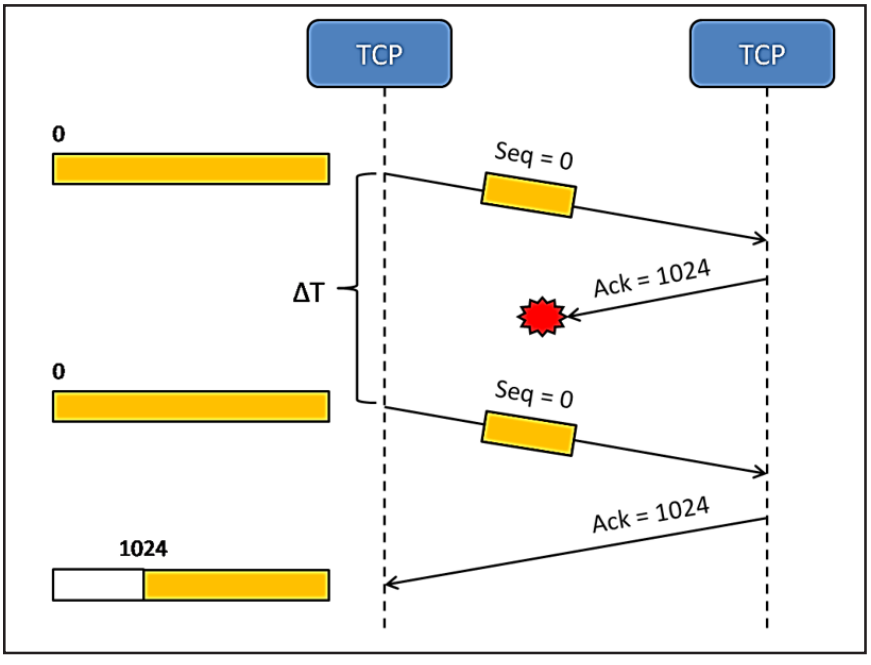


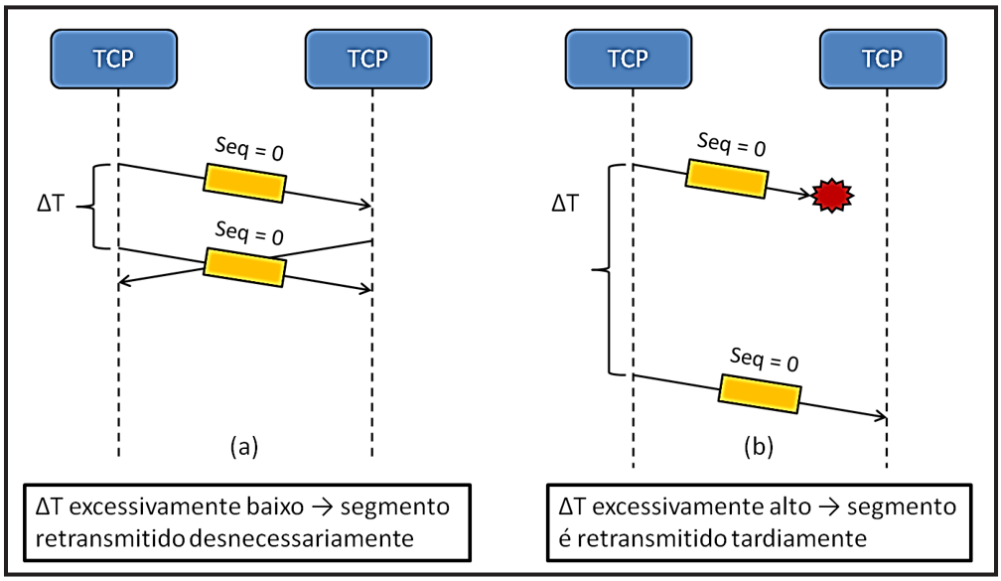
Figura 41: Perda do reconhecimento, e expiração do temporizador de retransmissão



Para entender a perda de segmentos que carregam ou não dados, há de considerar-se que, seguindo a teoria de redes, estes segmentos são encapsulados por PDUs do nível de rede – denominadas datagramas, as quais atravessam vários equipamentos, denominados roteadores, ao longo do caminho até seu destino. Quando um datagrama chega a um roteador, ele é armazenado temporariamente em uma fila. O problema é que, numa eventual condição de congestionamento naquele roteador, não há espaço na fila e, consequentemente, o datagrama que chega é descartado – isto é, ele é perdido. É função do nível de transporte (mais especificamente, do TCP) retransmiti-lo, e esta é a explicação para a existência do temporizador de retransmissão.

A discussão anterior leva à seguinte questão: qual deve ser o valor do intervalo de tempo ΔT no qual expira o temporizador de retransmissão? Conforme ilustrado pela Figura 42, definir estaticamente um valor excessivamente baixo faria com que eventualmente segmentos que tivessem chegado ao seu destino fossem retransmitidos desnecessariamente; por outro lado, um valor estático excessivamente alto faria com que demorasse muito para que um eventual segmento perdido fosse retransmitido.

Figura 42: Efeitos da definição estática do ΔT



O valor a ser definido para *ΔT*, denominado **RTO** (*Retransmission TimeOut*), varia proporcionalmente à variação do tempo de ida e volta de um segmento, este último denominado **RTT** (*Round-Trip Time*). Para o cálculo do RTO, uma entidade TCP mantém duas variáveis de estado, denominadas **SRTT** (*Smoothed Round-Trip Time*) e **RTTVAR** (*Round-Trip Time VARiation*).

Quando a primeira medida do RTT (*R*) é obtida, a entidade TCP realiza as seguintes atribuições:

$SRTT = R$

$RTTVAR = \frac{R}{2}$

$RTO = SRTT + 4 \cdot RTTVAR$

Quando uma medida subsequente do RTT (*R'*) é obtida, a entidade TCP realiza as seguintes atribuições:

$RTTVAR = 0{,}75 \cdot RTTVAR + 0{,}25 \cdot |SRTT - R'|$

$SRTT = 0{,}875 \cdot SRTT + 0{,}125 \cdot R'$

$RTO = SRTT + 4 \cdot RTTVAR$

A Tabela 5 ilustra um exemplo de cálculo do RTO com base em seis medidas iniciais de RTT. Percebe-se que todos os segmentos subsequentes foram recebidos antes do RTO calculado para eles. Quando isto não ocorre, o segmento é retransmitido e o RTO dobra de valor. Se o mesmo segmento for perdido novamente, mas uma vez o RTO dobra de valor. Este processo pode se repetir até que um teto definido para RTO seja atingido, a partir do qual o TCP sinaliza para o nível de Aplicação sobre a ocorrência de um erro irrecuperável.

Tabela 5: Exemplo de cálculo do RTO

	R'	SRTT	RTTVAR	RTO
1	98,9	98,9	49,45	296,7
2	106,0	99,79	38,86	255,23
3	99,0	99,69	29,34	217,05
4	99,5	99,67	22,05	187,87
5	105,0	100,34	17,87	171,82
6	104,0	100,80	14,32	158,08

O *Offset* indica onde a parte do segmento referente aos dados começa. Isso é necessário porque o cabeçalho do segmento TCP pode variar em tamanho, dependendo do uso do campo de tamanho variável Opções.

Na Figura 36, nota-se a presença de algumas *flags*. Elas são utilizadas por uma entidade TCP para sinalizar ao seu par a ocorrência de determinados fatos ou eventos. A *flag* URG, por exemplo, sinaliza que o campo Ponteiro para Dados Urgentes é significante. Este ponteiro é utilizado para sinalizar que uma porção dos dados em um segmento deve ter prioridade de leitura em relação aos demais dados daquele segmento. A *flag* ACK sinaliza que o número de reconhecimento, cuja função foi definida anteriormente, é significante em um segmento. A *flag* PSH sinaliza que os dados encapsulados devem ser entregues imediatamente ao nível de aplicação em vez de armazenados temporariamente em um *buffer* – esta *flag* é muito utilizada em segmentos que encapsulam dados provenientes de aplicações interativas. A *flag* RST é utilizada por uma entidade TCP remota para recusar a abertura de uma conexão – pela falta de uma aplicação associada a uma porta, por exemplo. As *flags* SYN e FIN são utilizadas nos processos de abertura e encerramento de uma conexão TCP.

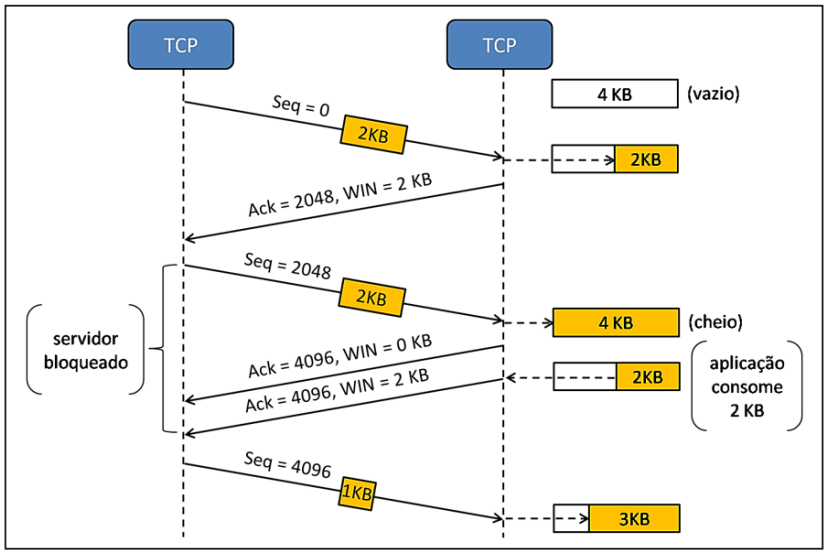
Toda entidade TCP possui um *buffer* onde são armazenados os dados recebidos de seus pares. Se a velocidade com a qual novos dados chegam a este buffer é maior que a velocidade com a qual os dados que já estão neles são consumidos pelo nível de Aplicação, o *buffer* tende a encher. Uma vez cheio, qualquer segmento que chegar à entidade será descartado, independentemente de ter chegado na ordem correta e de não possuir nenhum bit corrompido – ele será descartado por simplesmente não haver espaço em *buffer* para armazenar os dados encapsulados. O controle de fluxo é um procedimento particularmente útil neste cenário. Ele é realizado a partir do campo tamanho da janela, no cabeçalho do segmento TCP.

A Figura 43 ilustra um exemplo de aplicação do controle de fluxo. Nela, a entidade TCP receptora possui um *buffer* de 4 KB, inicialmente vazio. A entidade TCP transmissora envia um segmento contendo 2 KB de dados, que são colocados naquele *buffer.* A entidade receptora reconhece aquele segmento, informando que o tamanho da janela (WIN) agora é de 2 KB. Sabendo disso, a entidade transmissora envia um segmento contendo 2 KB de dados. Considerando que aqueles 2 KB completariam o *buffer* da entidade receptora, a entidade transmissora bloqueia a transmissão de dados posteriores àqueles. Ao reconhecer aqueles 2 KB, a entidade receptora também informa que o *buffer* está cheio. Somente quando o nível de aplicação consome 2 KB é a entidade TCP receptora envia um novo reconhecimento avisando que há espaço para mais 2 KB.

A entidade TCP transmissora sai do seu estado de bloqueio e transmite mais dados – neste caso, um segmento contendo 1 KB de dados. Estes dados serão armazenados no *buffer* e no reconhecimento àquele segmento, que não é ilustrado pela Figura 43, o tamanho da janela será

de 1 KB. Este é o valor que corresponde à diferença entre o tamanho total do *buffer* (4 KB) e o espaço ocupado nele (3 KB).

Figura 43: Exemplo de controle de fluxo



O controle de congestionamento é um procedimento realizado pelo TCP de forma que a taxa de transmissão se adapte às condições da rede. Para entendê-lo, o primeiro passo é compreender o conceito de janela de transmissão, que identifica os segmentos transmitidos, mas ainda não reconhecidos. A Figura 44 ilustra a aplicação da estratégia conhecida como *stop-and-go.* Nela, o tamanho da janela é sempre igual a 1 (um) segmento. No início, o segmento 1 é transmitido e uma cópia sua ocupa a janela. Quando o reconhecimento àquele segmento chega, a cópia é descartada e o segmento 2 é transmitido, com sua cópia passando a ocupar a janela.

Quando o reconhecimento associado ao segmento 2 chega, a cópia daquele segmento é descartada e o segmento 3 é transmitido com sua cópia passando a ocupar a janela. A continuidade deste processo, ilustrada na Figura 44, permite compreender como esta estratégia, bem como suas variantes, são baseadas no conceito de janela deslizante – de fato, graficamente a janela tem o efeito de "deslizar" ao longo do tempo.

Figura 44: Estratégia *stop-and-go*

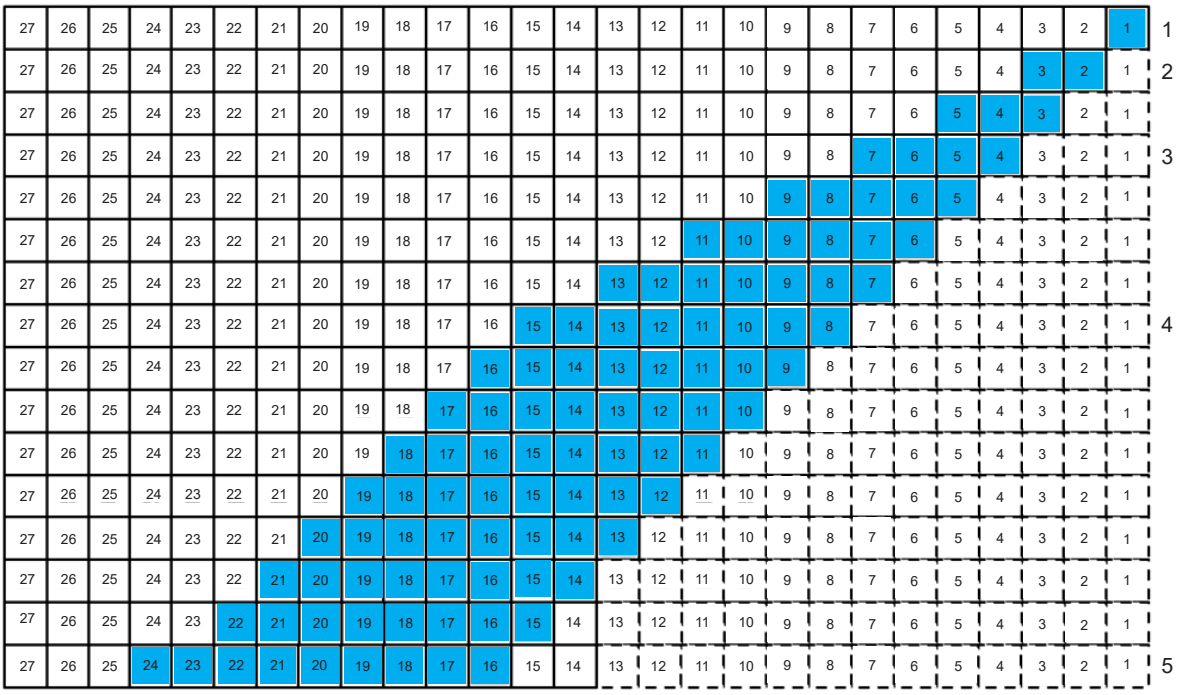
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	**1**
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	**2**	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	**3**	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	**4**	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	**5**	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	**6**	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	**7**	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	**8**	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	**9**	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	**10**	9	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	**11**	10	9	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	**12**	11	10	9	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	14	**13**	12	11	10	9	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	15	**14**	13	12	11	10	9	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	16	**15**	14	13	12	11	10	9	8	7	6	5	4	3	2	1
27	26	25	24	23	22	21	20	19	18	17	**16**	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

O problema com a estratégia *stop-and-go* é que ela não permite que uma taxa de transmissão elevada seja atingida, mesmo que as condições de tráfego da rede sejam favoráveis. Taxas de transmissão elevadas são conseguidas através de janelas maiores. Porém, a janela de transmissão não pode crescer indefinidamente – e é este o motivo pelo qual o TCP adapta o seu tamanho à situação da rede com relação às condições de tráfego. A Figura 45 ilustra como a janela de transmissão cresce em uma versão inicial do TCP, conhecida como TCP Tahoe. Esta versão possui uma fase conhecida como partida lenta, na qual a janela de transmissão inicia em 1 (um) e vai aumentando em 1 (um) a cada segmento reconhecido. Inicialmente, a janela é ocupada unicamente pela cópia do segmento 1 – que é transmitido. Quando o reconhecimento para aquele segmento chega, a cópia é descartada e a janela agora possui valor igual a 2 (dois), sendo definida pela cópia dos segmentos 2 e 3 – que são transmitidos.

Uma vez que o reconhecimento para o segmento 2 chega, a cópia daquele segmento é descartada e a janela possui valor igual a 3 (três), sendo definida pelas cópias dos segmentos 3, 4 e 5 – os dois últimos segmentos sendo transmitidos. Uma vez que o reconhecimento para o segmento 3 chega, a cópia daquele segmento é descartada e a janela possui valor igual a 4 (quatro), sendo definida pelas cópias dos segmentos 4, 5, 6 e 7 – os dois últimos segmentos sendo transmitidos. Processo semelhante pode ser visto até a transmissão do segmento 8.

A partir daquele momento, o tamanho da janela é mantido em 8 (oito) até que todos os seus segmentos originais – que vão do número 8 ao 16 – sejam reconhecidos. Quando isto acontece, o tamanho da janela aumenta em 1 (um), passando – neste caso – a 9 (nove). A Figura 45 destaca à direita os momentos nos quais a janela é inteiramente renovada (identificados como 1, 2, 3, 4 e 5).

Figura 45: Visão microscópica do TCP Tahoe

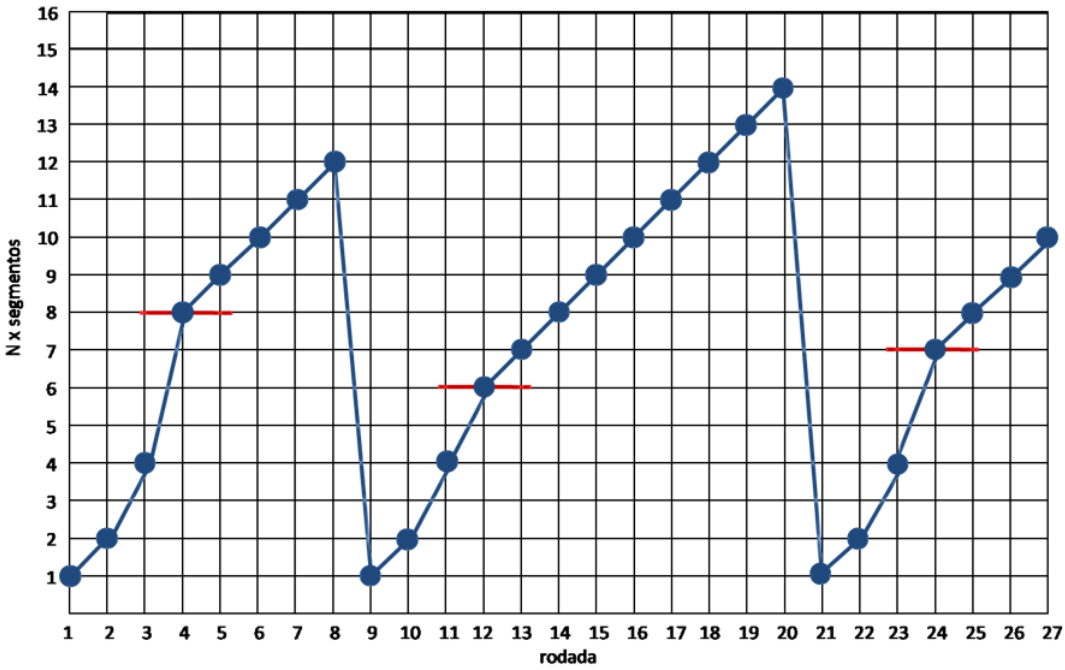


A Figura 46 ilustra os momentos nos quais a janela de transmissão é inteiramente renovada – atribui-se a cada um destes momentos o termo rodada. Na fase de partida lenta, descrita anteriormente, a janela dobra de tamanho a cada rodada, crescendo de forma exponencial. A partir de um determinado valor, denominado limiar, a janela aumenta linearmente. O fato é que o aumento da janela, embora mais lento, não se mantém para sempre. Durante a 8ª rodada, quando a janela de transmissão está em 12 segmentos, ocorre um evento de perda evidenciado pela expiração de um temporizador de retransmissão. O TCP Tahoe reage a este evento reduzindo a janela de transmissão para 1 (um) na 9ª rodada e redefinindo o valor do limiar para a metade do valor da janela atingida no momento da perda – haja vista que o evento de perda ocorreu quando a janela era de 12 (doze) segmentos, o novo valor do limiar passa a ser 6 (seis).

Assim, na 10ª rodada a janela é de 2 (dois) segmentos, na 11ª rodada ela é de 4 (quatro) segmentos e na 12ª rodada ela atinge o máximo de 6 (seis) segmentos. A partir dali, a janela volta a aumentar linearmente. As condições de tráfego da rede melhoram, e a expiração de um temporizador ocorre agora quando a janela de transmissão atinge o valor de 14 segmentos, na 20ª rodada. A entidade TCP volta à fase de partida lenta na 21ª rodada e a janela cresce exponencialmente até o limiar de 7 (sete) segmentos, a partir daí crescendo linearmente.

A natureza adaptativa do controle de congestionamento provido pelo TCP é tal que a janela de transmissão aumenta e diminui ao sabor das condições de tráfego da rede. Quando tais condições são favoráveis, verifica-se que eventos de perda ocorrem em janelas maiores. Por outro lado, quando elas pioram os eventos de perdas ocorrem em janelas menores.

Figura 46: Visão macroscópica do TCP Tahoe

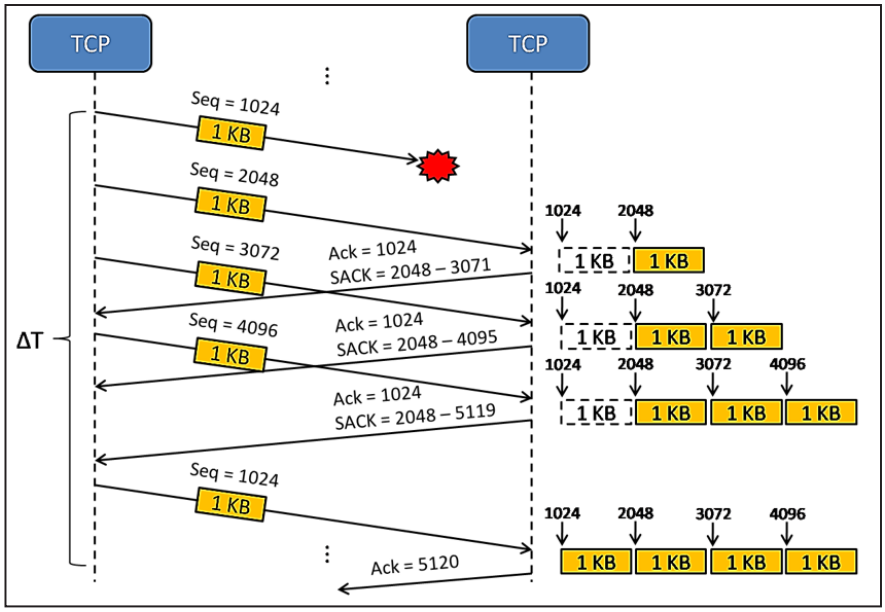


A versão do TCP comumente implementada em núcleos de sistema operacional é a TCP Reno. Esta versão caracteriza-se pela presença de um mecanismo conhecido como retransmissão rápida, levando em consideração a recepção de três reconhecimentos duplicados para retransmitir um segmento perdido antes da expiração de seu temporizador de retransmissão. A Figura 47 ilustra a aplicação deste mecanismo. Nela, quatro segmentos de 1 KB são transmitidos consecutivamente – supõe-se uma janela de quatro segmentos. O primeiro dos quatro é perdido.

Na recepção do segundo, a entidade TCP receptora percebe que há uma lacuna e envia um reconhecimento que informa que ela está esperando um segmento cujo número de sequência seja 1024 – e não 2048, como foi o caso. Esta entidade receptora informa também, por meio do atributo SACK (*Selective Acknowledgement*), presente no campo de opções do segmento, que ela recebeu dados fora de ordem na faixa de numeração que vai de 2048 até 3071. Após a recepção do terceiro segmento, a entidade TCP receptora envia outro reconhecimento que continua informando que ela espera receber um segmento cujo número de sequência seja 1024 e que atualiza a faixa de numeração dos dados recebida fora de ordem para de 2048 até 4095.

Após a recepção do quarto segmento, a entidade TCP receptora envia outro reconhecimento que continua informando que ela espera receber um segmento cujo número de sequência seja 1024 e que atualiza a faixa de numeração dos dados recebida fora de ordem para de 2048 até 5119. Quando a entidade TCP transmissora recebe este terceiro reconhecimento duplicado (em termos do número de reconhecimento), ela retransmite o primeiro segmento antes que o temporizador de retransmissão associado a ele expire, em ΔT.

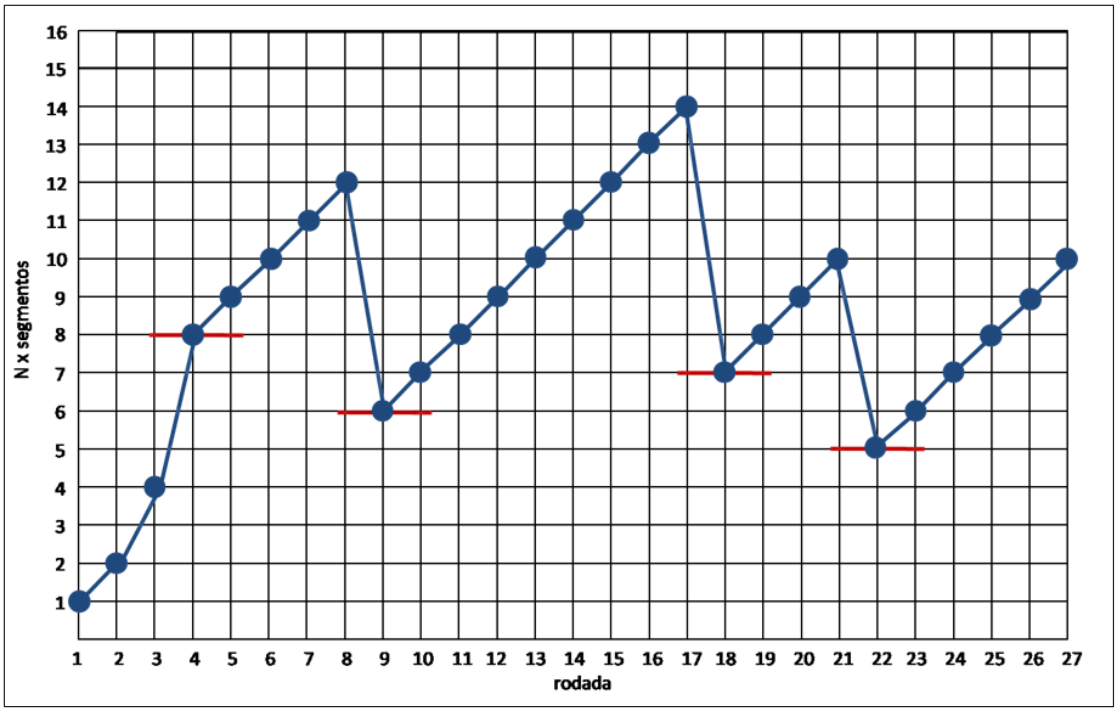
Figura 47: Exemplo de aplicação do mecanismo de retransmissão rápida



A Figura 48 ilustra graficamente como o TCP Reno leva em consideração a recepção de três reconhecimentos duplicados para ajustar a janela de transmissão. Nas 8ª, 17ª e 21ª rodadas, ocorrem eventos de perda evidenciados pela recepção de três reconhecimentos duplicados. Nas rodadas consecutivas àquelas, a janela de transmissão foi reduzida não para 1 (um), mas diretamente para o novo limiar – cujo valor continua correspondendo à metade do valor da janela na rodada em que houve o evento de perda.

A despeito do comportamento descrito no parágrafo anterior, o TCP Reno continua reduzindo o valor da janela de transmissão para 1 (um) quando o evento de perda é caracterizado pela expiração do temporizador de retransmissão – que ocorre quando a rede é congestionada em tal ponto que nem mesmo os reconhecimentos duplicados chegam à entidade TCP transmissora. Nestes casos, a adoção de uma postura mais conservadora, levando a entidade de volta à fase de partida lenta, parece ser mais aconselhável.

Figura 48: Visão macroscópica do TCP Reno



A análise da janela de transmissão realizada há pouco pressupõe que não havia limitação de espaço no *buffer* do receptor. Naturalmente, além das condições impostas pela rede, há de ser levado em conta aquele espaço – normalmente, contudo, o maior limitante à taxa de transmissão são as condições de tráfego da rede.

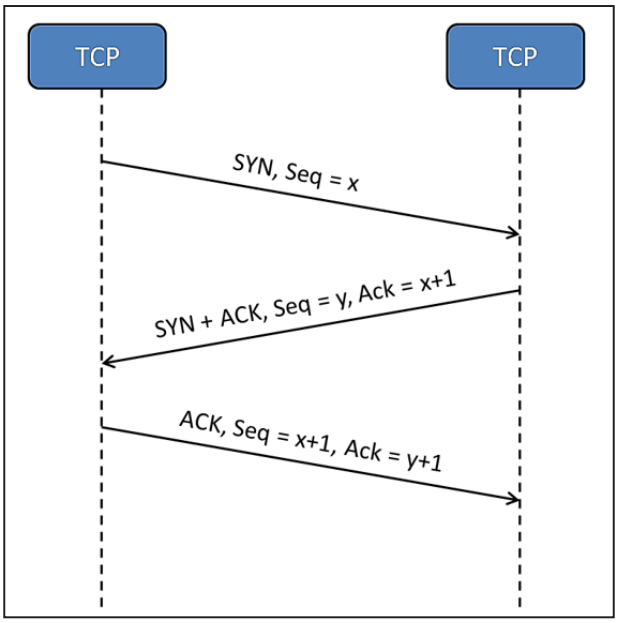
Os controles de erros, de fluxo e de congestionamento são realizados no transcorrer de uma conexão TCP. Esta conexão é aberta utilizando um procedimento conhecido como *three-way handshake*, ilustrado pela Figura 49. Trata-se de uma troca de três segmentos específicos que não transportam dados de Aplicação. Uma conexão TCP é iniciada por uma das partes, que envia um segmento com a *flag* SYN ativada e com o seu número de sequência inicial, x.

A outra parte responde com outro segmento contendo as *flags* SYN e ACK ativadas e com o seu número de sequência inicial, y. Este segmento reconhece o segmento anterior, ao possuir x+1 como número de reconhecimento. A parte iniciante envia um último segmento com a *flag* ACK ativada e com o número de sequência x+1. Este segmento também reconhece o segmento anterior, ao possuir y+1 como número de reconhecimento. Duas particularidades devem ser salientadas aqui:

1. Segmentos reconhecem segmentos anteriores que não transportam dados – o segmento com as *flags* SYN e ACK ativadas reconhece o segmento anterior com a *flag* SYN ativada, e o último segmento com a *flag* ACK ativada reconhece o segmento anterior com as *flags* SYN e ACK ativadas;

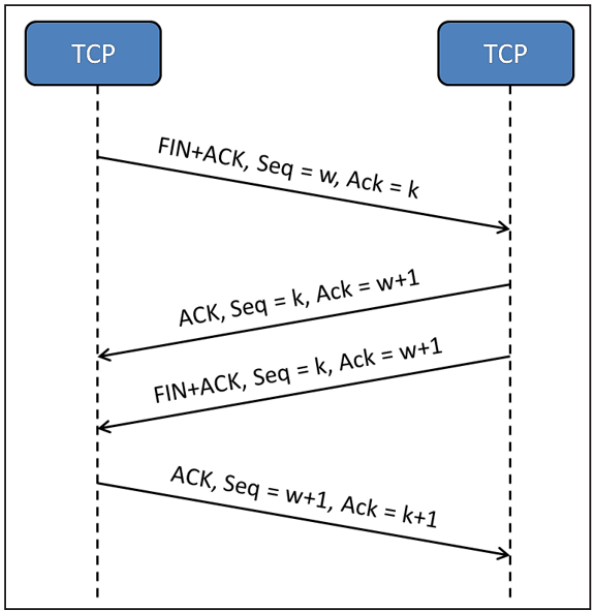
2. Os primeiros segmentos enviados pelas partes contendo dados terão números de sequência que repetem os últimos números – a parte que inicia a conexão enviará dados numerados a partir de x+1 e a outra parte enviará dados numerados a partir de y+1.

Figura 49: Abertura de conexão



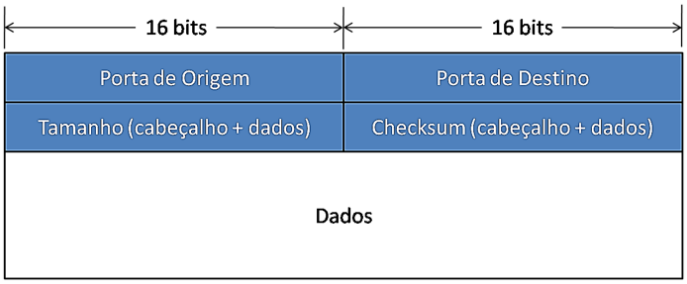
O encerramento de uma conexão TCP se dá bilateralmente, conforme ilustrado pela Figura 50. Uma das partes envia um segmento com as *flags* FIN e ACK ativadas. Este segmento pode ou não conter dados. A outra parte responde a este segmento com outro com a *flag* ACK ativada. Esta troca significa que a parte iniciante não deseja mais enviar dados para a outra. Mas isso não significa que a outra parte não queira enviar dados. Esta é a razão pela qual ela tem que enviar seu próprio segmento com as *flags* FIN e ACK ativadas, que será reconhecido pela outra parte. Nesta fase, à semelhança do que ocorre na fase de estabelecimento de conexão, segmentos que não contêm dados são reconhecidos – é o caso dos segmentos com as *flags* FIN e ACK ativadas.

Figura 50: Encerramento de conexão



O UDP (*User Datagram Protocol*) é especificado pela RFC 768. Ele é o protocolo de transporte utilizado por aplicações que não podem, ou não querem, pagar o preço dos serviços prestados pelo TCP. Este preço é o tempo gasto nos procedimentos que caracterizam o TCP. A Figura 51 ilustra o formato do segmento UDP. Comparado ao formato do segmento TCP, ilustrado pela Figura 36, o cabeçalho do UDP é minimalista.

Figura 51: Formato do segmento UDP



O UDP não realiza o controle de erros – isto explica a ausência de números de sequência e de reconhecimento no cabeçalho do segmento. Ele também não realiza o controle de fluxo – razão pela qual não existe o campo tamanho da janela – nem o controle de congestionamento. De fato, no UDP a taxa de segmentos recebidos é limitada apenas pela largura de banda. Neste protocolo, não existe sequer o conceito de "conexão" – todos os segmentos transmitidos contêm dados do nível de aplicação.

De todos os protocolos de aplicação estudados nesta unidade, apenas o DNS utiliza o UDP como protocolo de transporte – e isto se ampara no fato de que uma consulta DNS deve ser realizada o mais rapidamente possível. Porém, a despeito do que se possa pensar, o UDP é um protocolo de transporte muito utilizado na prática. Serviços de *streaming*, como aquele prestado pelo *YouTube*, fazem uso do UDP para transmissão de áudio e vídeo. De fato, se o TCP fosse utilizado por aqueles serviços, eventuais segmentos perdidos ao longo da transmissão seriam inevitavelmente retransmitidos; o efeito percebido pelo usuário final seria a paralisação de um vídeo a que ele estivesse assistindo. Psicologicamente, é melhor que haja uma pequena degradação no *streaming* recebido (que às vezes sequer é percebida) do que uma paralisação deste.

II O NÍVEL DE REDE

A versão atual do protocolo IP, IPv4, é executada por todos os computadores, bem como por equipamentos denominados roteadores, ao longo da internet. Esta unidade é dedicada inteiramente a este que é considerado o principal protocolo do modelo TCP/IP. Todos os campos que compõem um datagrama IPv4 são cobertos, e atenção especial é dedicada à exposição da problemática da escassez de endereços. Neste contexto, esta unidade aborda em detalhes as soluções de curto prazo (CIDR e NAT) e fornece noções acerca da solução de longo prazo para este problema: a adoção da próxima versão do protocolo IP, denominada IPv6.

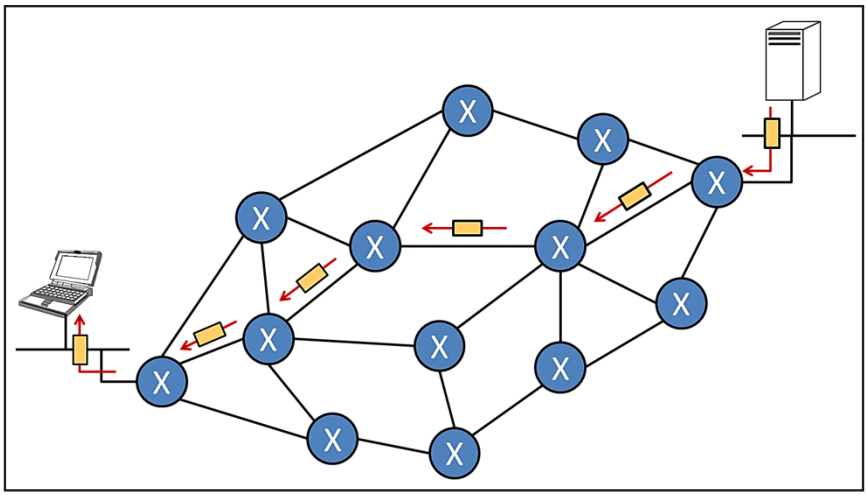
Objetivos:

- Descrever os campos que compõem o cabeçalho de um datagrama IPv4;
- Abordar as estratégias utilizadas atualmente com relação à escassez de endereços IPv4;
- Fornecer noções acerca da próxima versão do protocolo IP, o IPv6.

Introdução

A Figura 52 ilustra o caminho percorrido por um datagrama – denominação dada à PDU do nível de rede – até alcançar o seu destino, por meio de uma malha de roteadores. Este datagrama pode estar encapsulando um segmento TCP, que, por sua vez, pode estar encapsulando uma parte de um arquivo muito importante descarregado a partir de um servidor FTP. Cada roteador ao longo do caminho tem que repassar o datagrama por uma de suas interfaces, e saber exatamente qual é aquela interface é justamente a função do nível de rede.

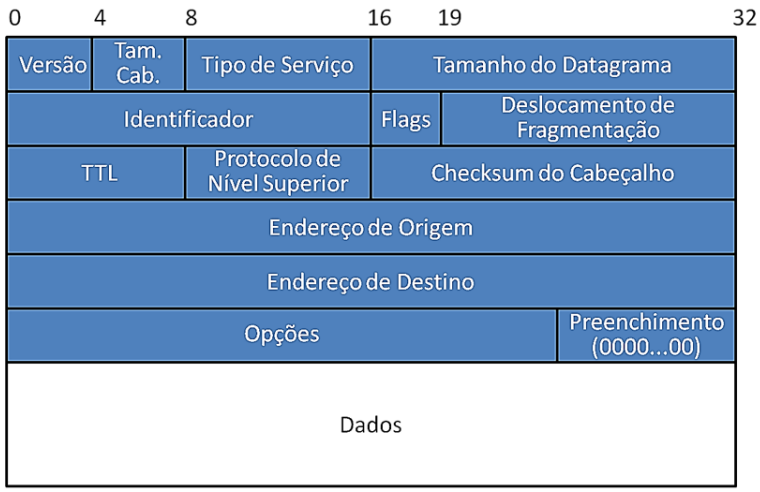
Figura 52: Um datagrama sendo roteado ao longo de uma malha de *roteadores*



Endereços e Máscaras

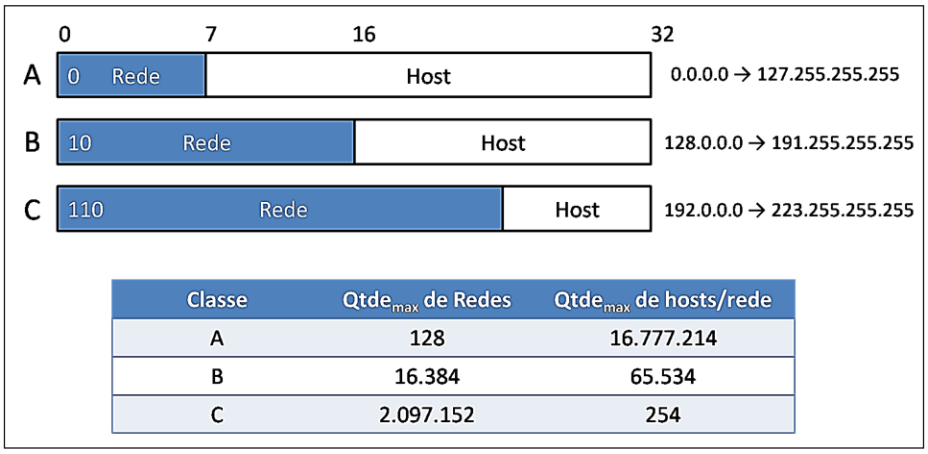
O protocolo do nível de rede é o IP (*Internet Protocol*), cuja versão atual (IPv4) é especificada pela RFC 791. A Figura 53 ilustra o formato do datagrama IPv4. Alguns de seus campos têm nomes que elucidam facilmente seus significados, tais como versão, tamanho do cabeçalho, tamanho do datagrama e *checksum*. Outros merecem uma análise mais minuciosa. Dentre eles estão os endereços de origem e destino. Um endereço IP é um número formado por quatro octetos – ou 32 bits. Como um número de 32 bits é difícil de ser memorizado por um humano, endereços IP são representados utilizando a notação decimal separada por pontos. Com esta notação, o endereço IP 11000000 00000101 00110000 00000011 é representado por 192.5.48.3.

Figura 53: Formato do datagrama IPv4



Um endereço IP está associado a uma interface de rede, e é dividido em uma porção de rede e uma porção de *host*. Dentre os endereços que fazem parte de uma mesma rede, todos eles possuem a mesma porção de rede. O que diferencia um do outro é a porção de *host*. Até 1981, a porção de rede era definida pelos oito primeiros bits de um endereço. Em termos práticos, a internet era limitada a 256 redes (28). A partir daquela data, entraram em vigor as classes de endereços. A Figura 54 ilustra as três classes principais definidas.

Figura 54: Classes de endereços



Classe	$Qtde_{max}$ de Redes	$Qtde_{max}$ de hosts/rede
A	128	16.777.214
B	16.384	65.534
C	2.097.152	254

Os endereços classe A eram caracterizados por possuírem o primeiro octeto (oito primeiros bits) destinado à porção de rede, e os três demais octetos (16 bits) destinados à porção de *host*. Além disso, o primeiro bit de um endereço classe A deve necessariamente começar com 0 (zero). Esta estruturação fez com que houvesse apenas 128 (28) redes classe A diferentes na internet, cada uma delas capaz de acomodar 16.777.214 (224 - 2) endereços úteis.

O desconto de dois endereços se deve à destinação do primeiro deles para compor o endereço da rede e do último para compor o endereço de *broadcast* da rede. Por exemplo, na rede classe A 10.0.0.0, o endereço 10.0.0.0 é o endereço da rede e 10.255.255.255 é o endereço de *broadcast*. O endereço de *broadcast* é utilizado quando se deseja enviar um datagrama que atinja, simultaneamente, todos os computadores de uma rede. Na Figura 54, observa-se que endereços classe A vão de 0.0.0.0 a 127.255.255.255.

Os endereços classe B eram caracterizados por possuírem os dois primeiros octetos (16 primeiros bits) destinados à porção de rede e os dois demais octetos (16 bits) destinados à porção de *host*. Além disso, os dois primeiros bits de um endereço classe B devem necessariamente ser 10. Esta estruturação fez com que houvesse 16.384 (214) redes classe B diferentes na Internet, cada uma delas capaz de acomodar 65.534 (216 - 2) endereços úteis.

Similarmente, o comentado com relação à classe A, o desconto de dois endereços se deve à destinação do primeiro deles para compor o endereço da rede e do último para compor o endereço de broadcast da rede. Por exemplo, na rede classe B 150.50.0.0, o endereço 150.50.0.0 é o endereço da rede e 150.50.255.255 é o endereço de *broadcast*. Na Figura 54, observa-se que endereços classe B vão de 128.0.0.0 a 191.255.255.255.

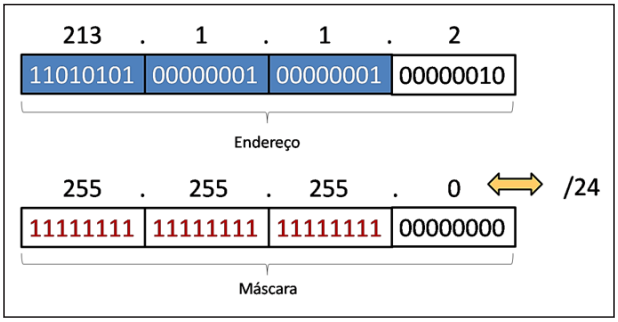
Os endereços classe C eram caracterizados por possuírem os três primeiros octetos (24 primeiros bits) destinados à porção de rede e o octeto restante (8 bits) destinado à porção de *host*. Além disso, os três primeiros bits de um endereço classe C devem, necessariamente, ser 110. Esta estruturação fez com que houvesse 2.097.152 (221) redes classe C diferentes na internet, cada uma delas capaz de acomodar 254 (28 - 2) endereços úteis.

Similarmente, o comentado com relação às classes A e B, o desconto de dois endereços se deve à destinação do primeiro deles para compor o endereço da rede e do último para compor o endereço de *broadcast* da rede. Por exemplo, na rede classe C 200.150.100.0, o endereço 200.150.100.0 é

o endereço da rede e 200.150.100.255 é o endereço de *broadcast.* Na Figura 54, observa-se que endereços classe C vão de 192.0.0.0 a 223.255.255.255.

Todo endereço de rede é acompanhado de uma máscara. A máscara de rede também é um número de 32 bits, cuja definição se dá por marcar com o bit 1 (um) toda a porção de rede do endereço correspondente e marcar os demais bits (a porção de host) com o bit 0 (zero). A Figura 55 ilustra a máscara de rede associada ao endereço 213.1.1.2. Este é um endereço pertencente à rede classe C 213.1.1.0, cuja porção de rede corresponde aos 24 primeiros bits (213.1.1). A máscara de rede é, portanto, definida marcando-se com o bit 1 (um) todos aqueles bits e com o bit 0 (zero) os demais bits. Ela é expressa como 255.255.255.0. Uma forma alternativa e mais compacta de representação de uma máscara de rede é a composição do caractere "/" acrescido da quantidade de bits 1 (um) da máscara. No exemplo da Figura 55, a representação compacta da máscara 255.255.255.0 é /24.

Figura 55: Exemplo de máscara de rede



A Figura 56 ilustra uma interconexão de redes simples. Ela é formada por três redes *Fast Ethernet* formadas a partir de *hubs* e interconectadas por um roteador. Tipicamente, as redes são representadas juntamente com suas máscaras, como é o caso de 213.1.1.0/24, 213.1.2.0/24 e 213.1.3.0/24. Por estar conectado a cada uma das três redes, o roteador possui três endereços distintos, cada um deles pertencendo a uma delas. Estes endereços são 213.1.1.1, 213.1.2.1 e 213.1.3.1. Cada um deles é considerado o endereço de *gateway* para sua rede correspondente. O endereço de *gateway* pode ser qualquer um dos 254 endereços úteis de cada rede. Porém, a convenção utilizada ao longo deste documento atribui o primeiro daqueles endereços ao endereço de *gateway.*

Na Figura 56, percebe-se que os endereços IP individuais são representados somente por meio da porção de *host* – por exemplo, o endereço 213.1.2.2 é representado apenas por .2 (visto que a porção de rede 213.1.2 está implícita). Todos os computadores ilustrados pela figura são configurados com endereço IP, máscara de rede e endereço de *gateway.* As três interfaces do roteador são configuradas com seu endereço IP e sua máscara de rede.

Figura 56: Uma interconexão de redes simples, formada por três redes

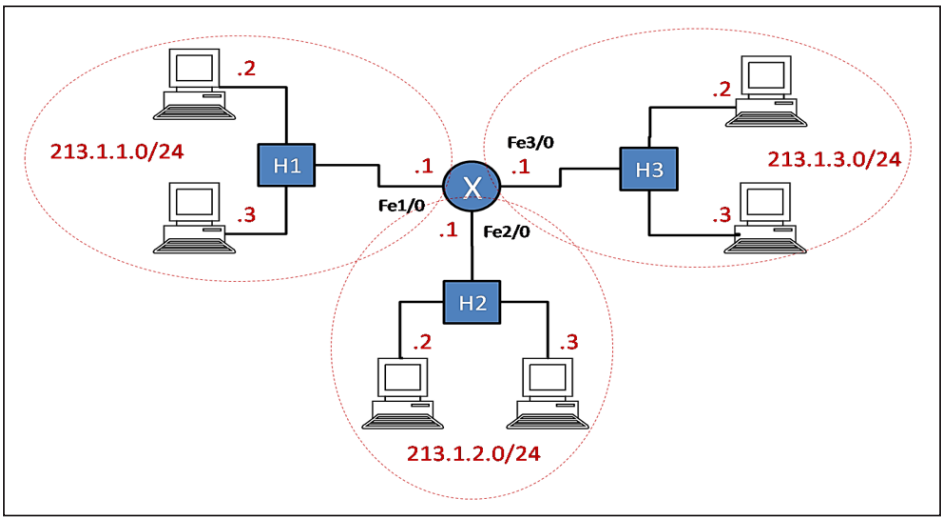


Tabela de Rotas e Roteamento

A Tabela 6 ilustra o conteúdo da tabela de rotas do roteador da Figura 56. A tabela de rotas de um roteador é composta por entradas. Cada entrada possui o endereço de uma rede acessível a partir do roteador e sua respectiva máscara. Além disso, a entrada contém endereço IP do próximo roteador no caminho até aquela rede (denominado próximo salto) e a interface por meio da qual o roteador vai enviar o datagrama endereçado àquela rede (denominada interface de saída). Na tabela, não há próximo salto para as redes 213.1.1.0/24, 213.1.2.0/24 e 213.1.3.0/24.

Tabela 6: Tabela de rotas do roteador da Figura 56

Rede	Máscara	Próximo Salto	Interface de Saída
213.1.1.0	255.255.255.0	—	Fe1/0
213.1.2.0	255.255.255.0	—	Fe2/0
213.1.3.0	255.255.255.0	—	Fe3/0

As entradas que são vistas referem-se a redes às quais o roteador está diretamente conectado, e estão presentes automaticamente na tabela de rotas. Elas são formadas da seguinte forma:

• Na interface Fe1/0, o roteador realiza um AND bit-a-bit entre o endereço IP (213.1.1.1) e a máscara (255.255.255.0) atribuídos àquela interface. O resultado é 213.1.1.0, o endereço da rede à qual a interface Fe1/0 está diretamente conectada;

• Na interface Fe2/0, o roteador realiza um AND bit-a-bit entre o endereço IP (213.1.2.1) e a máscara (255.255.255.0) atribuídos àquela interface. O resultado é 213.1.2.0, o endereço da rede à qual a interface Fe2/0 está diretamente conectada;

• Na interface Fe3/0, o roteador realiza um AND bit-a-bit entre o endereço IP (213.1.3.1) e a máscara (255.255.255.0) atribuídos àquela interface. O resultado é 213.1.3.0, o endereço da rede à qual a interface Fe3/0 está diretamente conectada.

O roteamento é o procedimento realizado por um roteador cujo resultado é o encaminhamento de um datagrama recebido por uma (e somente uma) das suas interfaces. Ele é feito junto à tabela de rotas do roteador com base no endereço IP de destino do datagrama. A Figura 57 ilustra este procedimento. Para cada uma das N entradas da tabela de rotas, é computado um AND bit-a-bit entre o endereço IP de destino e a máscara presente na entrada. Se o resultado corresponder à rede da entrada, a rota é tida como elegível. Pode haver mais de uma rota elegível. Neste caso, uma regra (vista mais adiante) definirá qual delas será a rota escolhida. O procedimento termina com o encaminhamento do datagrama pela interface correspondente à rota escolhida.

Figura 57: Procedimento realizado para o roteamento de datagramas

```
Para (i = 1; i <= N; i++) {
  Se ((IP_dst & Máscara[i]) == Rede[i]) {
    Selecione rota como elegível;
  }
}
Resolva potenciais conflitos;
Encaminhe datagrama pela rota escolhida;
```

Como exemplo, supõe-se que um datagrama é enviado por 213.1.1.2 e é endereçado a 213.1.2.2. Os seguintes passos são executados:

1. Na primeira entrada, é computado um AND bit-a-bit entre o endereço IP de destino (213.1.2.2) e a máscara da entrada (255.255.255.0). O resultado (213.1.2.0) não corresponde à rede da entrada (213.1.1.0). Conclusão: esta rota não é elegível;

2. Na segunda entrada, é computado um AND bit-a-bit entre o endereço IP de destino (213.1.2.2) e a máscara da entrada (255.255.255.0). O resultado (213.1.2.0) corresponde à rede da entrada (213.1.2.0). Conclusão: esta rota é elegível;

3. Na terceira entrada, é computado um AND bit-a-bit entre o endereço IP de destino (213.1.2.2) e a máscara da entrada (255.255.255.0). O resultado (213.1.2.0) não corresponde à rede da entrada (213.1.2.0). Conclusão: esta rota não é elegível;

4. Havendo-se apenas uma rota elegível, correspondente à segunda entrada, o datagrama para 213.1.2.2 é encaminhado pela interface daquela entrada – isto é, Fe2/0.

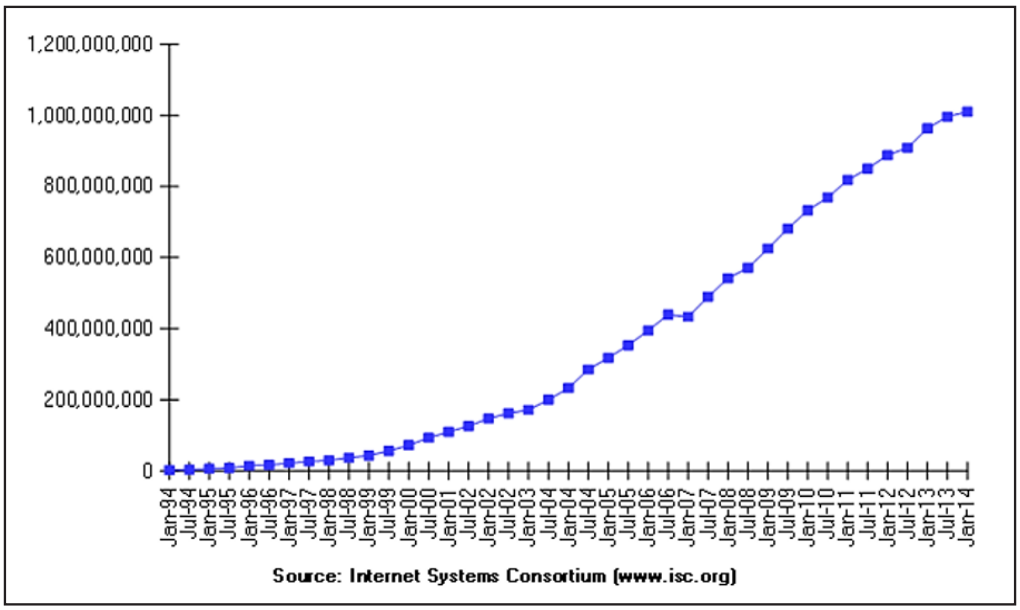
A Escassez de Endereços IPv4

A IANA (*Internet Assigned Numbers Authority*) é uma organização encarregada, dentre outras coisas, da alocação de endereços IP. Como esta é uma tarefa complexa, a IANA conta com as seguintes RIRs (*Regional Internet Registries*):

- AFRINIC (*African Network Information Center*), abrangendo a África;
- ARIN (*American Registry for Internet Numbers)*, abrangendo os Estados Unidos, Canadá, várias partes do Caribe e Antártida;
- APNIC (*Asia-Pacific Network Information Centre*), abrangendo a Ásia, Austrália, Nova Zelândia e países vizinhos;
- LACNIC (*Latin America and Caribbean Network Information Centre*), abrangendo a América Latina e algumas partes do Caribe;
- RIPE NCC (*Réseaux IP Européens Network Coordination Centre*), abrangendo Europa, Rússia, Oriente Médio e Ásia Central.

A Figura 58 ilustra o crescimento exponencial da quantidade de computadores (*hosts*) na internet. Nela, pode-se perceber que, em janeiro de 2014, já havia perto de um bilhão de *hosts*. Esta demanda por endereços IPv4 (inimaginável na época da criação da internet), juntamente com a alocação dos endereços em faixas que correspondem a redes classes A, B e C, tendiam a esgotar os endereços disponíveis muito antes do que era previsto originalmente.

Figura 58: Crescimento da quantidade de computadores na internet



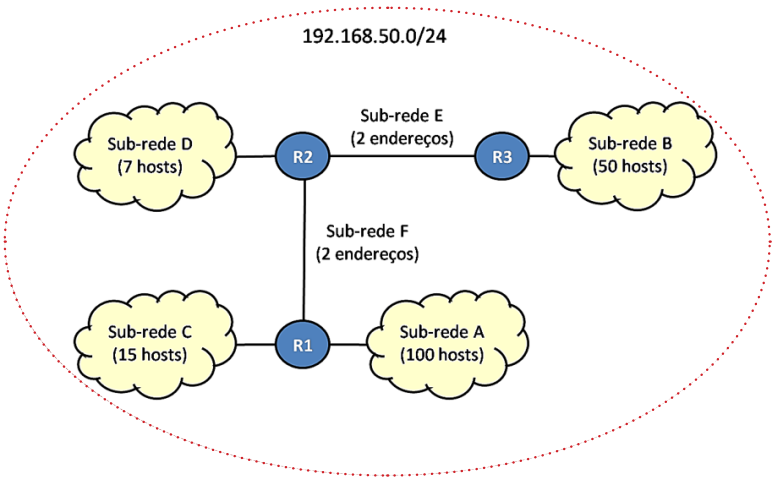
A solução definitiva para o problema da escassez de endereços é a adoção do IPv6, a nova versão do protocolo IP. Ela possui endereços de 128 bits, em vez dos endereços de 32 bits do IPv4. Esta é considerada uma solução de longo prazo, pela complexidade de sua implementação – não apenas todos os computadores da internet terão que possuir endereços IPv6, mas também todos os roteadores terão que possuí-los e ser capazes de manipular datagramas IPv6. Por esta razão, foram criadas soluções temporárias, de curto prazo, que são implementadas atualmente e que trabalham com o IPv4. Estas soluções atendem pelos nomes de CIDR e NAT.

CIDR

Definida em 1993 pela RFC 1517, o CIDR (*Classless InterDomain Routing*) tem como função diminuir o desperdício na alocação de endereços IPv4. A concepção do CIDR surgiu em uma época na qual se constatou que a maioria das redes classe B possuía menos de 50 *hosts* – há de ser lembrado que tais redes possuem suporte para 65.534 hosts. O CIDR utiliza um conceito denominado VLSM (*Variable Length Subnet Masking*), cuja essência é a definição de máscaras de tamanho variável para sub-redes formadas a partir de redes maiores. Assim, não existe mais o conceito de classes de endereços na internet, e as suas redes têm máscaras cujos tamanhos são bem variados – não mais se restringindo a /8, /16 e /24.

A Figura 59 ilustra um exemplo no qual um administrador de rede recebe uma faixa de endereços definida por 192.168.50.0/24 e deve atender à demanda de cada sub-rede de uma interconexão. Em outras palavras, ele deve definir – da forma mais eficiente possível – quais são o endereço de rede e máscara de cada sub-rede de forma a comportar a quantidade de endereços solicitados.

Figura 59: Uma interconexão de redes com suas demandas *individuais*



O processo de segmentação de uma rede maior em sub-redes menores é feito "roubando" bits da porção de *host* original e fazendo-os compor a (nova) porção de rede, dividindo a rede original em duas metades. A Figura 60 ilustra a primeira segmentação realizada, dividindo a rede 192.168.50.0/24 nas sub-redes 192.168.50.0/25 e 192.168.50.128/25. Neste processo, o primeiro bit (da esquerda para a direita) da porção de *host* original é "roubado" e estabelecido em 0 (zero) na primeira sub-rede e em 1 (um) na segunda sub-rede.

Cada uma das sub-redes tem capacidade para suportar 126 endereços úteis (27 – 2), e qualquer uma delas pode ser a sub-rede A, que requer 100 *hosts*. Adotando-se uma convenção interna de escolher sempre a primeira opção, a sub-rede A passou a ser 192.168.50.0/25. Há ainda 16 endereços ociosos, mas esta quantidade é bem menor do que se teria caso uma rede /24 inteira (uma antiga rede classe C) fosse atribuída à sub-rede A. O efeito gráfico da primeira segmentação é ilustrado pela Figura 61.

Figura 60: Primeira segmentação realizada

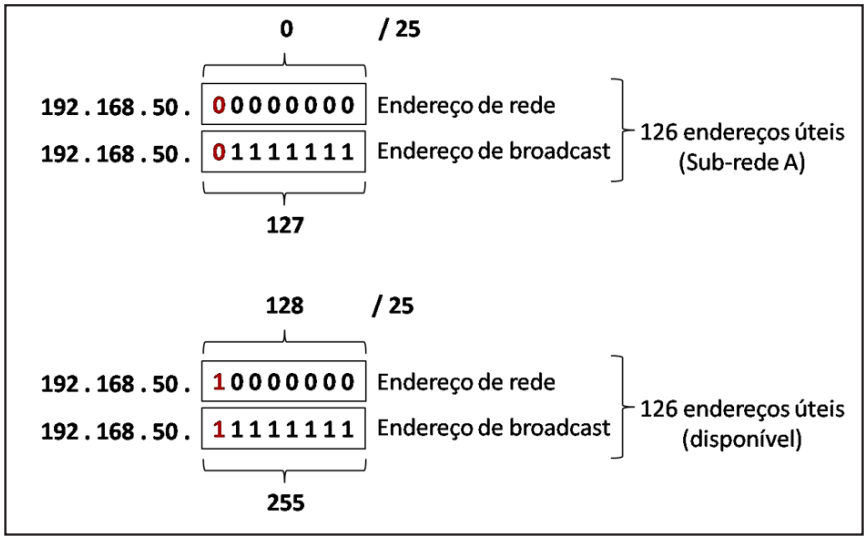
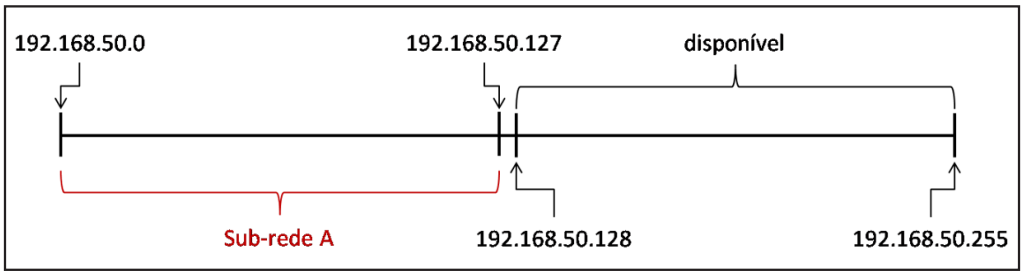


Figura 61: Efeito gráfico da primeira segmentação



A Figura 62 ilustra a segunda segmentação realizada, dividindo a sub-rede 192.168.50.128/25 nas sub-redes 192.168.50.128/26 e 192.168.50.192/26. Neste processo, mais um bit é "roubado" da porção de *host* da sub-rede original e incorporado à porção de rede das sub-redes derivadas. Também mais uma vez, este bit é estabelecido em 0 (zero) na primeira sub-rede e em 1 (um) na segunda sub-rede. Cada uma das sub-redes tem capacidade para suportar 62 endereços úteis (26 – 2), e qualquer uma delas pode ser a sub-rede B, que requer 50 *hosts*. Prosseguindo na convenção interna de escolher sempre a primeira opção, a sub-rede B passou a ser 192.168.50.128/26. A Figura 63 ilustra o efeito gráfico da segunda segmentação.

Figura 62: Segunda segmentação realizada

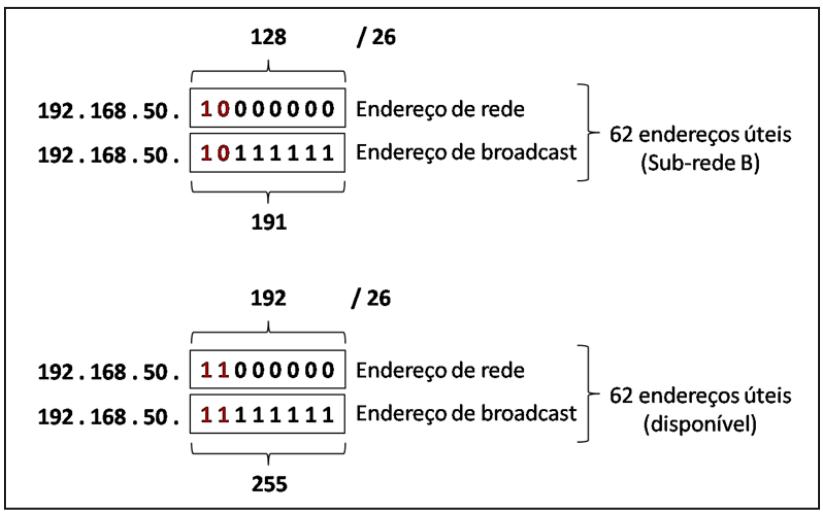
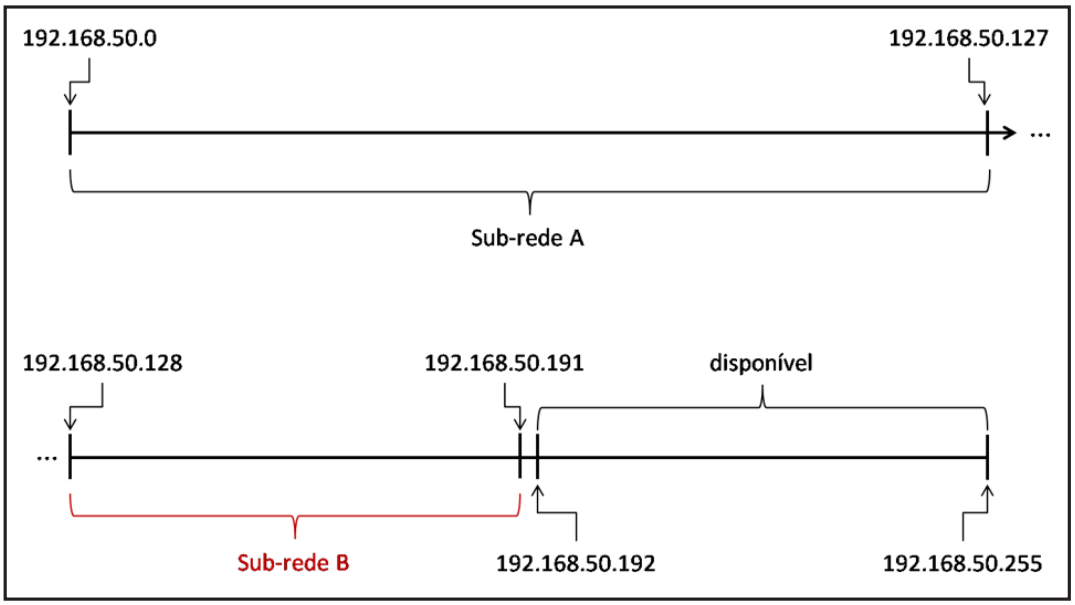


Figura 63: Efeito gráfico da segunda segmentação



A Figura 64 ilustra a terceira segmentação realizada, dividindo a sub-rede 192.168.50.192/26 nas sub-redes 192.168.50.192/27 e 192.168.50.224/27. Neste processo, mais um bit é "roubado" da porção de *host* da sub-rede0, original e incorporado à porção de rede das subirdes derivadas. Também mais uma vez, este bit é estabelecido em 0 (zero) na primeira sub-rede e em 1 (um) na segunda sub-rede. Cada uma das sub-redes tem capacidade para suportar 30 endereços úteis (25 – 2), e qualquer uma delas pode ser a sub-rede C, que requer 15 *hosts*. Prosseguindo na convenção interna de escolher sempre a primeira opção, a sub-rede C passou a ser 192.168.50.192/27. A Figura 65 ilustra o efeito gráfico da terceira segmentação.

Figura 64: Terceira segmentação realizada

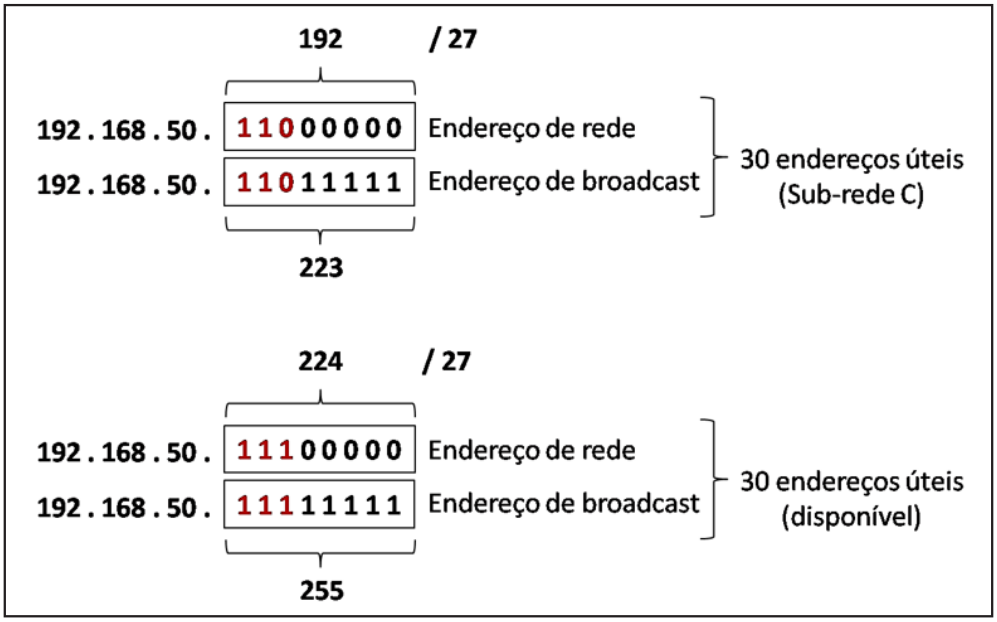
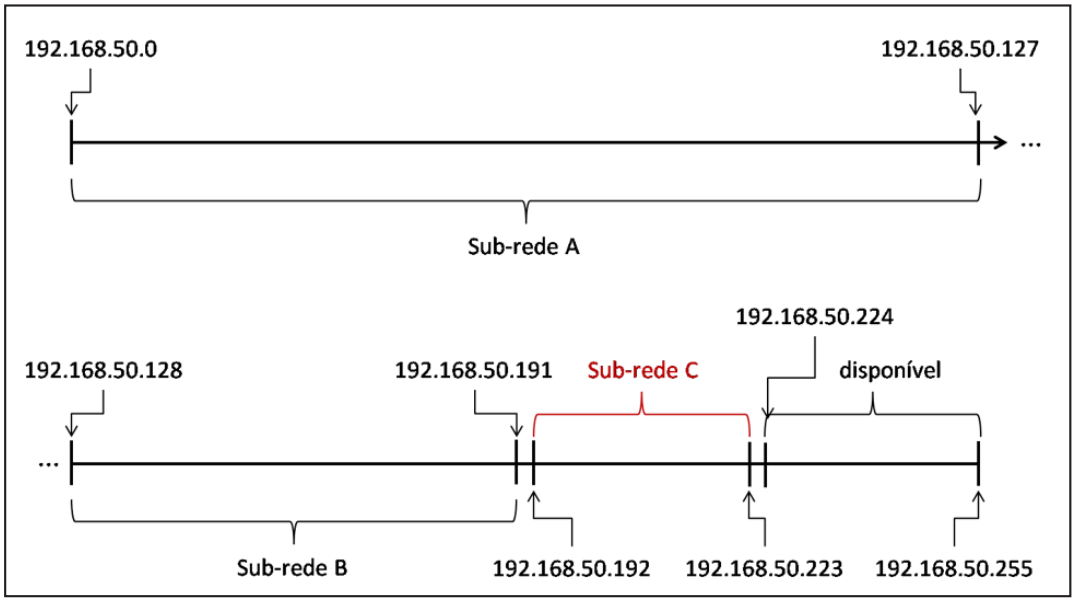


Figura 65: Efeito gráfico da terceira segmentação



A Figura 66 ilustra a quarta segmentação realizada, dividindo a sub-rede 192.168.50.224/27 nas sub-redes 192.168.50.224/28 e 192.168.50.240/28. Neste processo, mais um bit é "roubado" da porção de *host* da sub-rede original e incorporado à porção de rede das sub- redes derivadas. Também mais uma vez, este bit é estabelecido em 0 (zero) na primeira sub-rede e em 1 (um) na segunda sub-rede. Cada uma das sub-redes tem capacidade para suportar 14 endereços úteis (24 – 2), e qualquer uma delas pode ser a sub-rede D, que requer sete *hosts*. Prosseguindo na convenção interna de escolher sempre a primeira opção, a sub-rede D passou a ser 192.168.50.224/28. A Figura 67 ilustra o efeito gráfico da quarta segmentação.

Figura 66: Quarta segmentação realizada

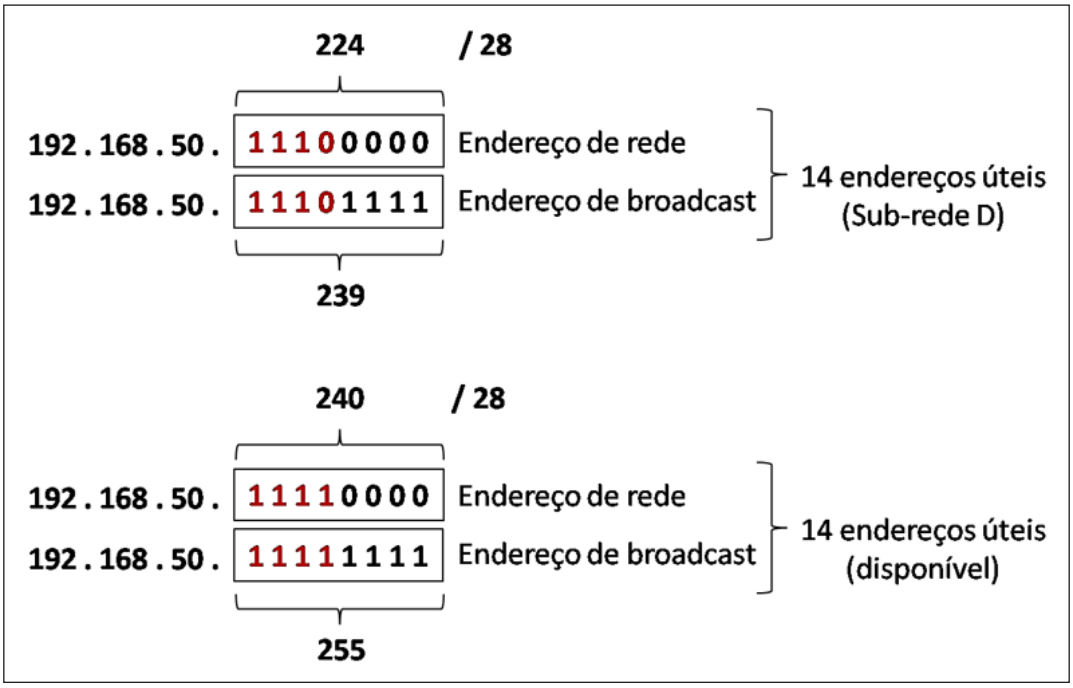
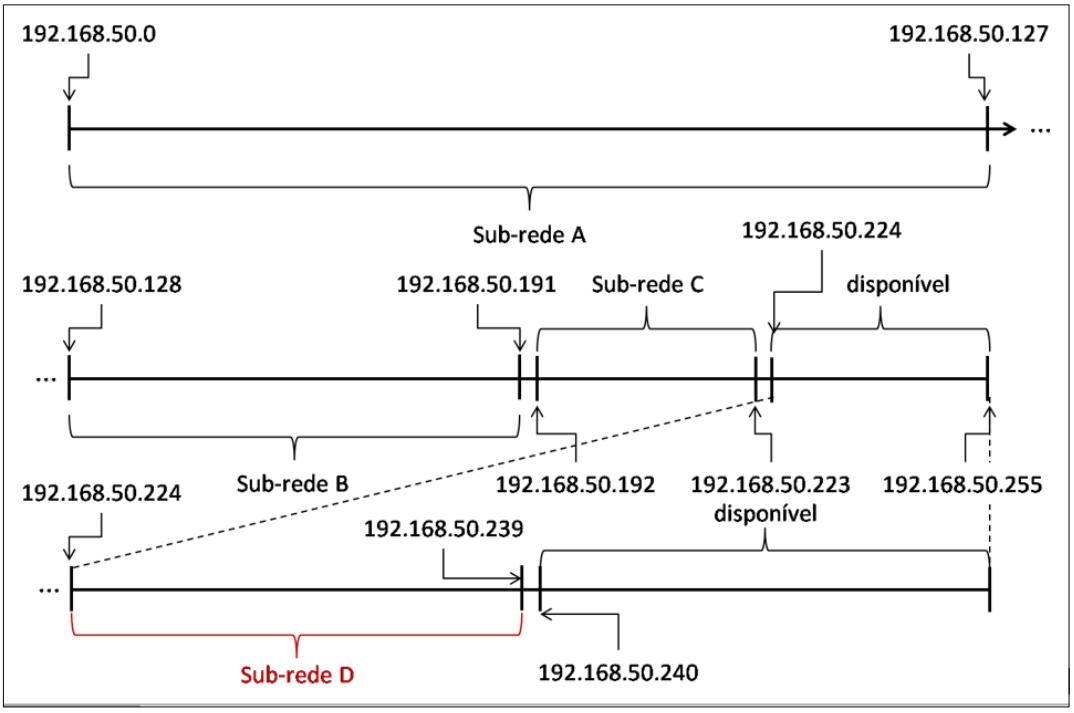


Figura 67: Efeito gráfico da quarta segmentação



A Figura 68 ilustra a quinta segmentação realizada, dividindo a sub-rede 192.168.50.240/28 nas sub-redes 192.168.50.240/29 e 192.168.50.248/29. Neste processo, mais um bit é "roubado" da porção de *host* da sub-rede original e incorporado à porção de rede das sub-redes derivadas. Também mais uma vez, este bit é estabelecido em 0 (zero) na primeira sub-rede e em 1 (um) na segunda sub-rede. Cada uma das sub-redes tem capacidade para suportar 6 endereços úteis (23 – 2). Embora qualquer uma delas pudesse ser atribuída à sub-rede D ou E, a solução mais eficiente é segmentar ainda mais uma destas sub-redes. A razão para isto é que as sub-redes D e E são segmentos ponto-a-ponto, que sob hipótese alguma experimentarão algum crescimento em termos de quantidade de endereços utilizados. A Figura 69 ilustra o efeito gráfico da quinta segmentação.

Figura 68: Quinta segmentação realizada

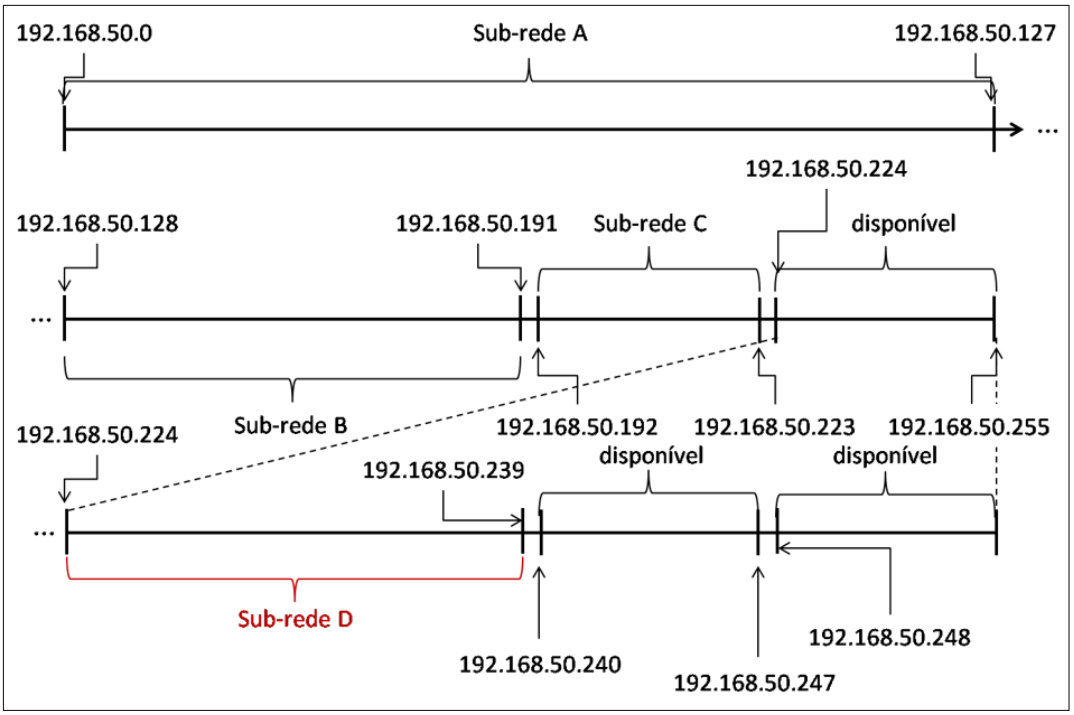
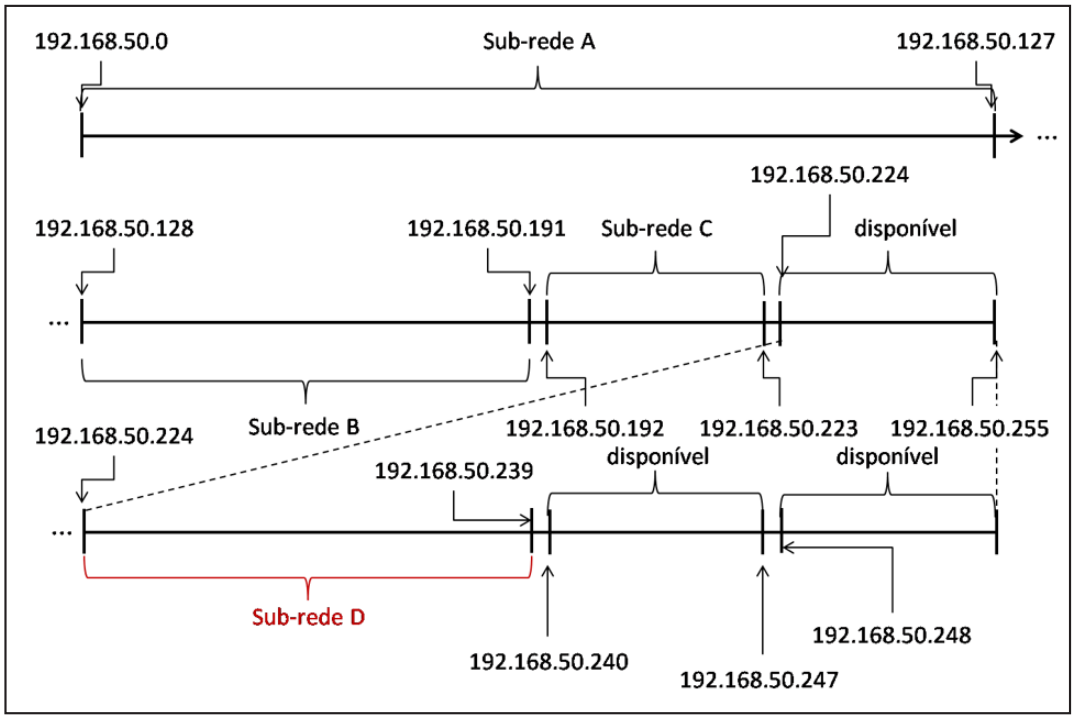


Figura 69: Efeito gráfico da quinta segmentação



A Figura 70 ilustra a sexta segmentação realizada, dividindo a primeira das sub-redes da (192.168.50.240/29) nas sub-redes 192.168.50.240/30 e 192.168.50.244/30. Neste processo, mais um bit é "roubado" da porção de *host* da sub-rede original e incorporado à porção de rede das sub-redes derivadas. Também mais uma vez, este bit é estabelecido em 0 (zero) na primeira sub-rede e em 1 (um) na segunda sub-rede. Cada uma das sub-redes tem capacidade para suportar 2 endereços úteis (22 – 2). A primeira delas é atribuída à sub-rede D e a segunda é atribuída à sub-rede E. A Figura 71 ilustra o efeito gráfico da sexta segmentação.

Figura 70: Sexta segmentação realizada

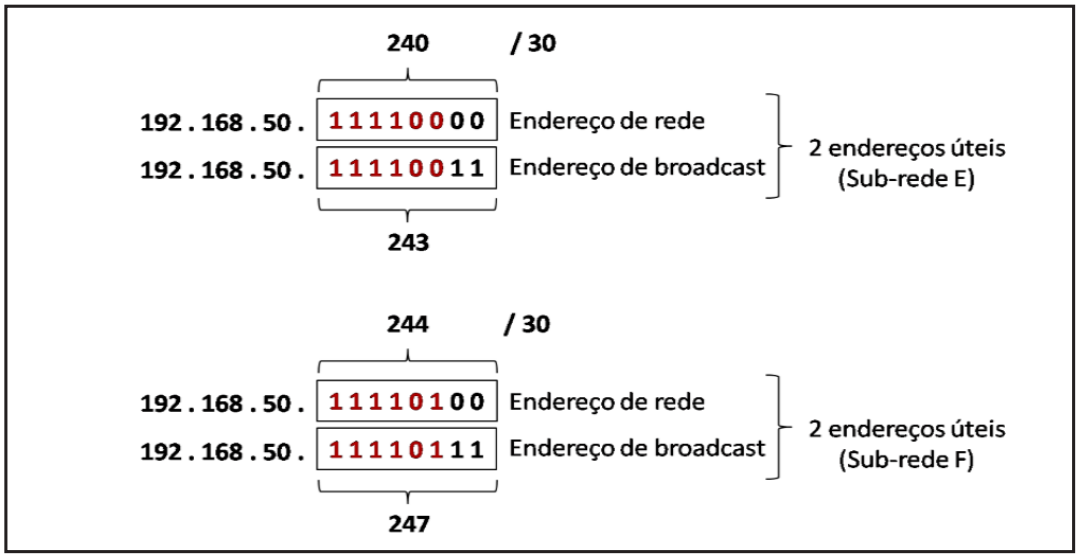
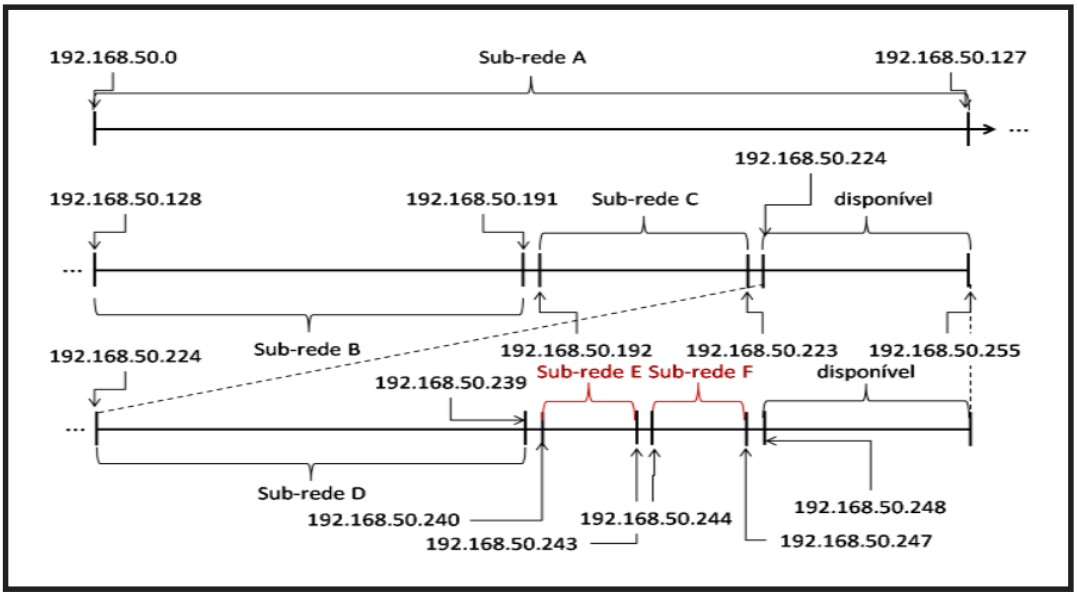


Figura 71: Efeito gráfico da sexta segmentação



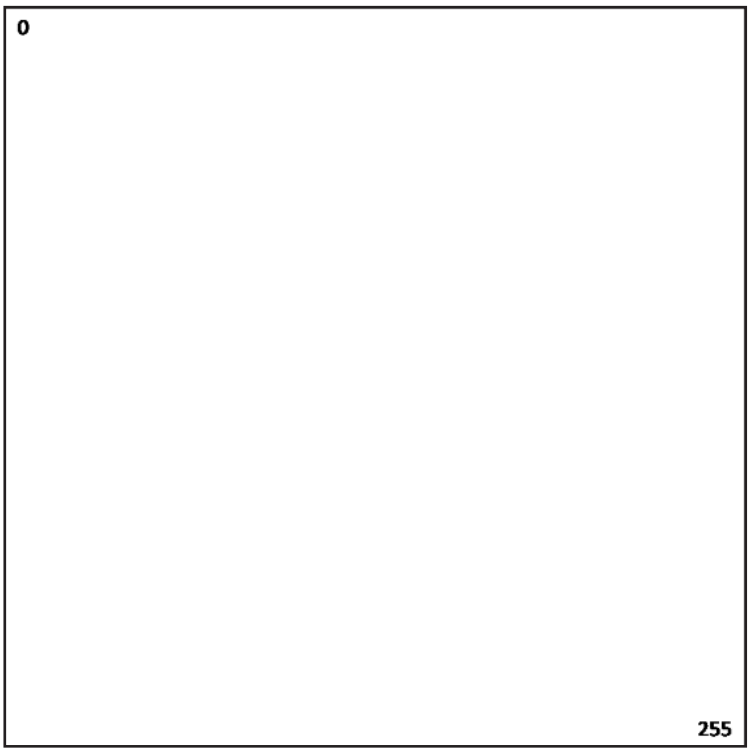
A Tabela 7 ilustra o resultado final da segmentação. Todas as demandas foram atendidas e percebe-se que há ainda uma faixa de endereços disponível (192.168.50.248/29) para futuras ampliações na interconexão.

Tabela 7: Resultado final da segmentação

Sub-rede	Endereço de Sub-rede/Máscara	Faixa de Endereços Utilizáveis	Endereço de Broadcast
A	192.168.50.0/25	192.168.50.1 – 192.168.50.126	192.168.50.127
B	192.168.50.128/26	192.168.50.129 – 192.168.50.190	192.168.50.191
C	192.168.50.192/27	192.168.50.193 – 192.168.50.222	192.168.50.223
D	192.168.50.224/28	192.168.50.225 – 192.168.50.238	192.168.50.239
E	192.168.50.240/30	192.168.50.241 – 192.168.50.242	192.168.50.243
F	192.168.50.244/30	192.168.50.245 – 192.168.50.246	192.168.50.247
disponível	192.168.50.248/29	192.168.50.249 – 192.168.50.254	192.168.50.255

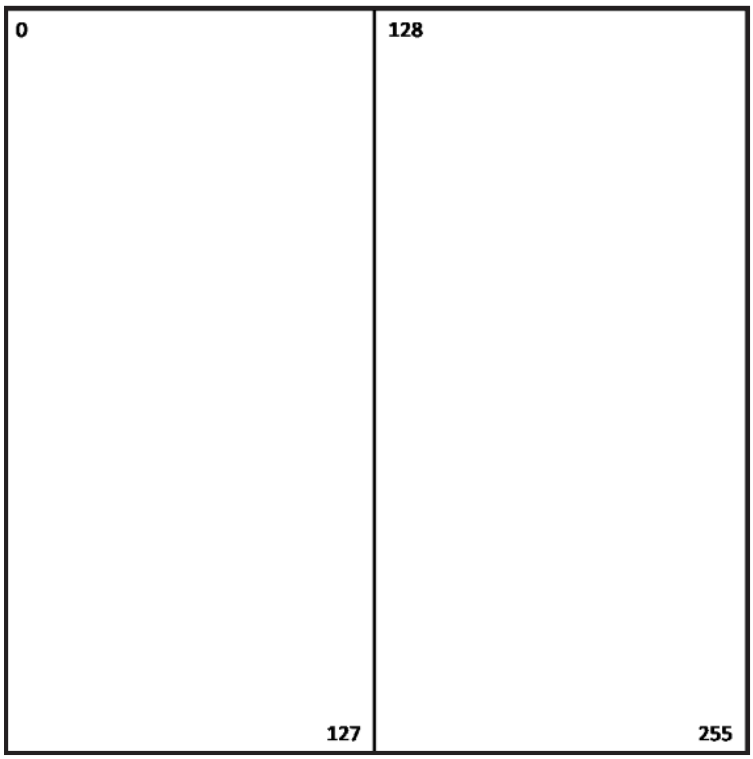
A segmentação de uma faixa de endereços pode ser simplificada ainda mais com o método da caixa. Parte-se de uma caixa inicial, ilustrada pela Figura 72, representando uma rede cuja máscara é /24 (255.255.255.0) e, portanto, cuja capacidade de alocação é de 254 endereços úteis. Os endereços inicial e final são posicionados nos extremos da caixa.

Figura 72: A caixa representa uma rede cuja máscara é /24 (255.255.255.0)



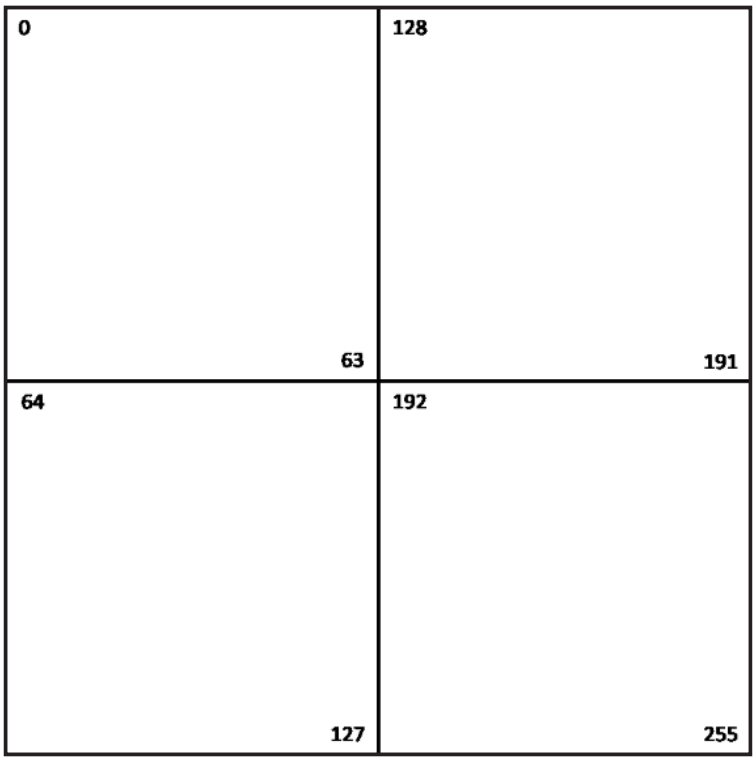
Em seguida, a caixa é dividida em duas, conforme ilustra a Figura 73. Cada uma delas representa uma sub-rede cuja máscara é /25 (255.255.255.128), com capacidade de alocação de 126 endereços úteis.

Figura 73: Cada caixa representa uma sub-rede cuja máscara é /25 (255.255.255.128)



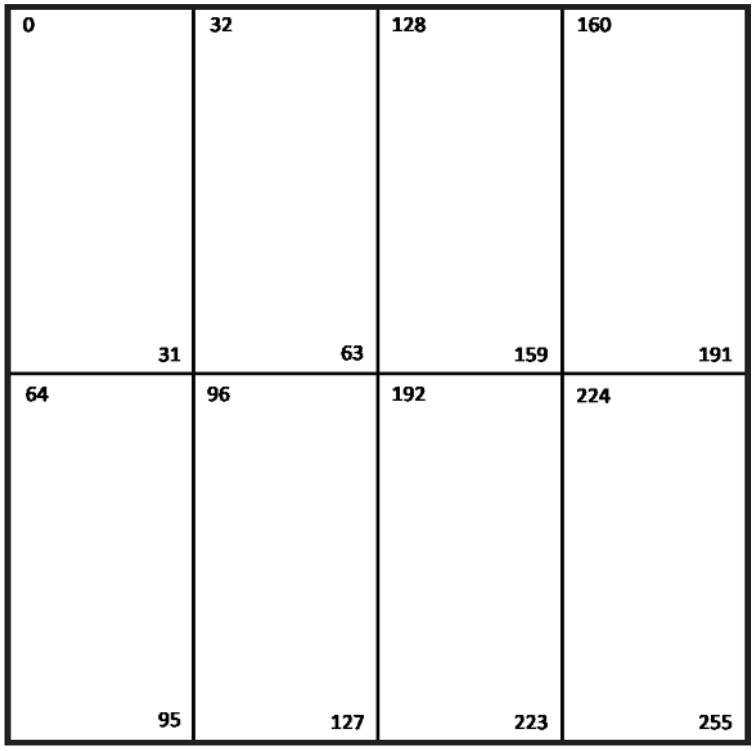
Na Figura 74, cada uma das duas caixas é dividida em duas. Cada caixa resultante representa uma sub-rede cuja máscara é /26 (255.255.255.192), com capacidade de alocação de 62 endereços úteis.

Figura 74: Cada caixa representa uma sub-rede cuja máscara é /26 (255.255.255.192)



A Figura 75 ilustra a divisão de cada uma das quatro caixas em duas outras. Cada uma das oito caixas resultantes representa uma sub-rede cuja máscara é /27 (255.255.255.224), com capacidade de alocação de 30 endereços úteis.

Figura 75: Cada caixa representa uma sub-rede cuja máscara é /27 (255.255.255.192)



A Figura 76 ilustra a divisão de cada uma das oito caixas em duas. Cada uma das 16 caixas resultantes representa uma sub-rede cuja máscara é /28 (255.255.255.240), com capacidade de alocação de 14 endereços úteis.

Figura 76: Cada caixa representa uma sub-rede cuja máscara é /28 (255.255.255.240)

0 15	32 47	128 143	160 175
16 31	48 63	144 159	176 191
64 79	96 111	192 207	224 239
80 95	112 127	208 223	240 255

A Figura 77 ilustra a divisão de cada uma das 16 caixas em duas. Cada uma das 32 caixas resultantes representa uma sub-rede cuja máscara é /29 (255.255.255.248), com capacidade de alocação de seis endereços úteis.

Figura 77: Cada caixa representa uma sub-rede cuja máscara é /29 (255.255.255.248)

0 7	8 15	32 39	40 47	128 135	136 143	160 167	168 175
16 23	24 31	48 55	56 63	144 151	152 159	176 183	184 191
64 71	72 79	96 103	104 111	192 199	200 207	224 231	232 239
80 87	88 95	112 119	120 127	208 215	216 223	240 247	248 255

Finalmente, a Figura 78 ilustra a divisão de cada uma das 32 caixas em duas. Cada uma das 64 caixas resultantes representa uma sub-rede cuja máscara é /30 (255.255.255.252), com capacidade de alocação de dois endereços úteis. Tipicamente, sub-redes com esta máscara são alocadas para segmentos ponto-a-ponto – conforme visto anteriormente.

Figura 78: Cada caixa representa uma sub-rede com máscara /30 (255.255.255.252)

0 3	8 11	32 35	40 43	128 131	136 139	160 163	168 171
4 7	12 15	36 39	44 47	132 135	140 143	164 167	172 175
16 19	24 27	48 51	56 59	144 147	152 155	176 179	184 187
20 23	28 31	52 55	60 63	148 151	156 159	180 183	188 191
64 67	72 75	96 99	104 107	192 195	200 203	224 227	232 235
68 71	76 79	100 103	108 111	196 199	204 207	228 231	236 239
80 83	88 91	112 115	120 123	208 211	216 219	240 243	248 251
84 87	92 95	116 119	124 127	212 215	220 223	244 247	252 255

A Figura 78 pode ser utilizada diretamente para a definição das sub-redes ilustradas pela Figura 59. Não é necessário construí-la desde o início. A Figura 79 ilustra o resultado final da aplicação do método da caixa, que leva ao mesmo resultado do método anterior.

Figura 79: Resultado da aplicação do método da caixa

Sub-rede A
192.168.50.0/25
Sub-rede B
192.168.50.128/26
Sub-rede B
192.168.50.192/27
Sub-rede C
192.168.50.224/28
Sub-rede D
192.168.50.240/30
Sub-rede E
192.168.50.244/30

0 3	8 11	32 35	40 43	128 131	136 139	160 163	168 171
4 7	12 15	36 39	44 47	132 135	140 143	164 167	172 175
16 19	24 27	48 51	56 59	144 147	152 155	176 179	184 187
20 23	28 31	52 55	60 63	148 151	156 159	180 183	188 191
64 67	72 75	96 99	104 107	192 195	200 203	224 227	232 235
68 71	76 79	100 103	108 111	196 199	204 208	228 231	236 239
80 83	88 91	112 115	120 123	208 211	216 219	240 243	248 251
84 87	92 95	116 119	124 127	212 215	220 223	244 247	252 255

A Figura 80 ilustra a aplicação da segmentação na interconexão. *Hubs* são utilizados nas sub-redes que não são ponto-a-ponto. Uma representação mais compacta é fornecida pela Figura 81. Nela, as sub-redes são representadas, simplesmente, por segmentos de reta, sem *hubs* ou computadores.

Figura 80: Aplicação da segmentação na interconexão

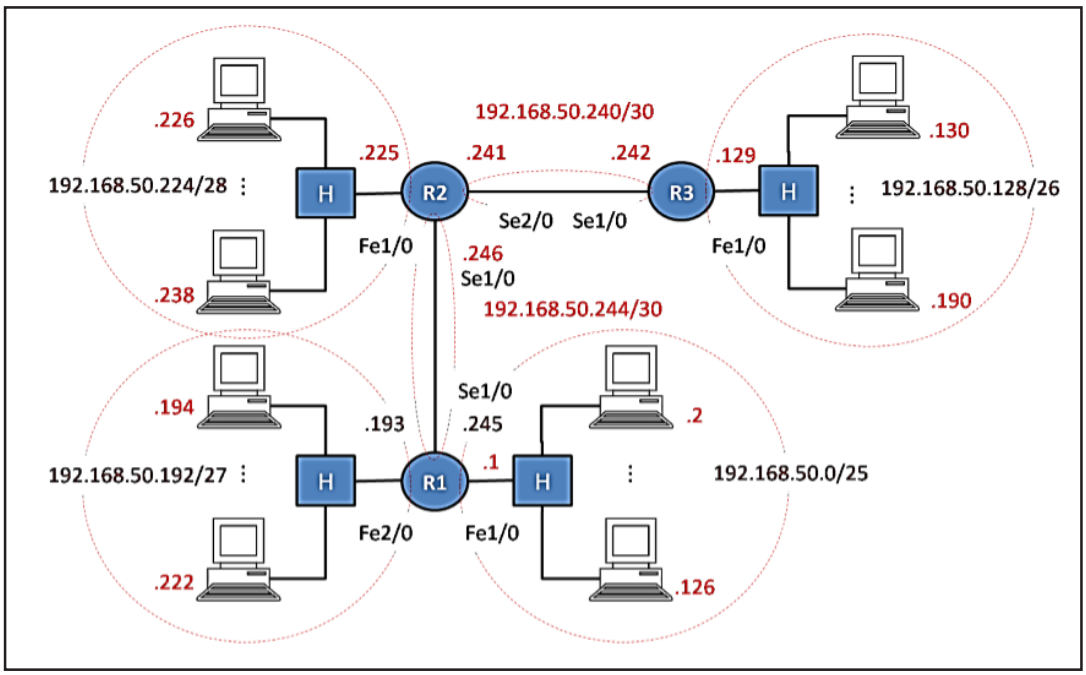
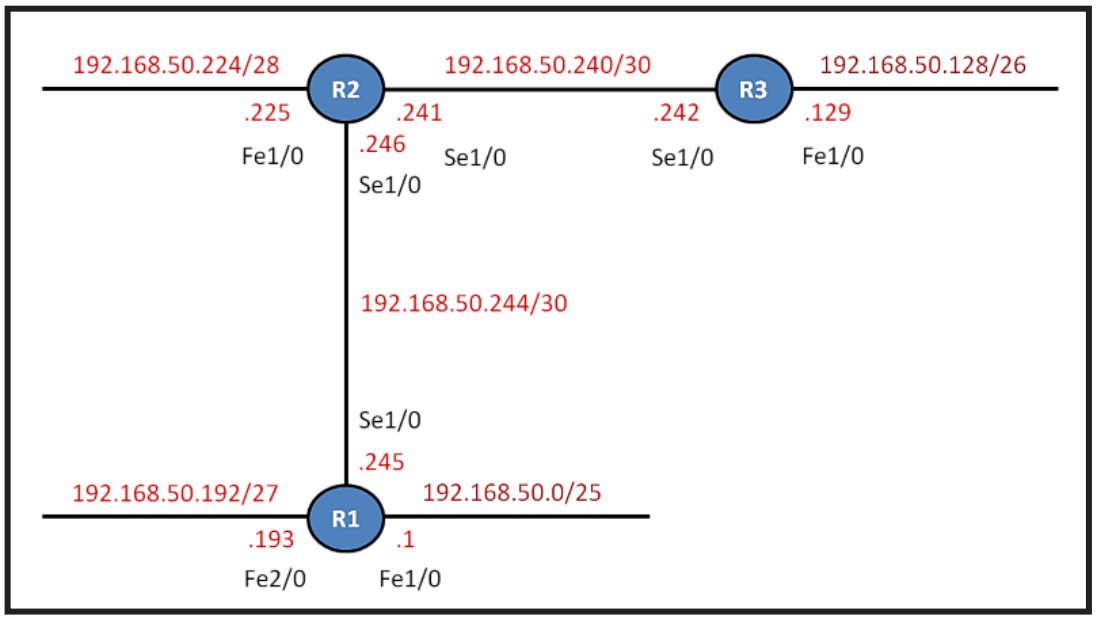


Figura 81: Forma alternativa de representação da interconexão



A Figura 82 ilustra a tabela de rotas de R1. Nela, as sub-redes 192.168.50.0/25, 192.168.50.192/27 e 192.168.50.244/30 estão presentes automaticamente pelo fato de R1 estar diretamente conectado a elas. As demais entradas, correspondentes às sub-redes 192.168.50.224/28, 192.168.50.240/30 e 192.168.50.128/26, foram adicionadas estaticamente – isto é, mediante intervenção humana. O próximo salto para todas aquelas sub-redes corresponde a 192.168.50.246, que é o endereço IP do roteador R2 na sub-rede a qual R1 faz parte.

Figura 82: Tabela de rotas de R1

Rede	Máscara	Próximo Salto	Interface
192.168.50.0	/25	—	Fe1/0
192.168.50.192	/27	—	Fe2/0
192.168.50.244	/30	—	Se1/0
192.168.50.224	/28	192.168.50.246	Se1/0
192.168.50.240	/30	192.168.50.246	Se1/0
192.168.50.128	/26	192.168.50.246	Se1/0

A Figura 83 ilustra a tabela de rotas de R2. Nela, as sub-redes 192.168.50.224/28, 192.168.50.240/30 e 192.168.50.244/30 estão presentes automaticamente pelo fato de R2 estar diretamente conectado a elas. As demais entradas, correspondentes às sub-redes 192.168.50.0/25, 192.168.50.192/27 e 192.168.50.128/26, foram adicionadas estaticamente. O próximo salto para 192.168.50.0/25 e 192.168.50.192/27 corresponde a 192.168.50.245, que é o endereço IP do roteador R1 na sub-rede a qual R2 faz parte. O próximo salto para 192.168.50.128/26 corresponde a 192.168.50.242, que é o endereço IP do roteador R3 na sub-rede a qual R2 faz parte.

Figura 83: Tabela de rotas de R2

Rede	Máscara	Próximo Salto	Interface
192.168.50.224	/28	—	Fe1/0
192.168.50.240	/30	—	Se2/0
192.168.50.244	/30	—	Se1/0
192.168.50.0	/25	192.168.50.245	Se1/0
192.168.50.192	/27	192.168.50.245	Se1/0
192.168.50.128	/26	192.168.50.242	Se2/0

A Figura 84 ilustra a tabela de rotas de R3. Nela, as sub-redes 192.168.50.128/26 e 192.168.50.240/30 estão presentes, automaticamente, pelo fato de R3 estar diretamente conectado a elas. As demais entradas, correspondentes às sub-redes 192.168.50.0/25, 192.168.50.192/27, 192.168.50.224/28 e 192.168.50.244/30, foram adicionadas estaticamente. O próximo salto para todas estas últimas sub-redes corresponde a 192.168.50.241, que é o endereço IP do roteador R2 na sub-rede a qual R3 faz parte.

Figura 82: Tabela de rotas de R1

Rede	Máscara	Próximo Salto	Interface
192.168.50.128	/26	—	Fe1/0
192.168.50.240	/30	—	Se1/0
192.168.50.0	/25	192.168.50.241	Se1/0
192.168.50.192	/27	192.168.50.241	Se1/0
192.168.50.224	/28	192.168.50.241	Se1/0
192.168.50.244	/30	192.168.50.241	Se1/0

Agregação de rotas

As tabelas de rota de R1, R2 e R3 contêm seis entradas, cada. Estas entradas correspondem as seis sub-redes da interconexão. O tamanho da tabela de rotas de um roteador pode ser reduzido por meio do processo denominado agregação de rotas. Duas ou mais rotas podem ser substituídas por uma rota agregada quando elas, na tabela de rotas, possuem o mesmo "próximo salto". O efeito da agregação de rotas é a redução do tamanho da tabela de rotas. A exigência de memória para um roteador é, portanto, menor do que seria para o armazenamento da tabela original. Além disso, o roteador pode realizar o roteamento de um datagrama mais rapidamente porque há menos entradas a serem examinadas.

O processo de agregação de rotas resulta em um endereço de rede e uma máscara de rede para a rota agregada. O endereço de rede leva em consideração as similaridades entre os endereços de rede das rotas a serem agregadas. Em particular, as similaridades são replicadas até que seja identificada uma dissimilaridade. A partir do ponto de dissimilaridade, todos os bits do endereço de rede da rota agregada são preenchidos com o bit 0 (zero). A máscara de rede da rota agregada é preenchida com bits 1 (um) até o ponto em que ocorreu a dissimilaridade entre os endereços de rede das rotas a serem agregadas. A partir daquele ponto, ela é preenchida com bits 0 (zero).

Conforme ilustrado pela Figura 85, na tabela de rotas de R1 as três últimas entradas podem ser substituídas por uma única rota agregada porque elas possuem o mesmo "próximo salto" – no caso, 192.168.50.246. Utilizando o procedimento descrito no parágrafo anterior, chega-se a uma rota agregada representada por 192.168.50.128/25. A Figura 86 ilustra a nova tabela de rotas de R1 com a rota agregada.

Figura 85: Agregação de rotas na tabela de rotas de R1

Rede	Máscara	Próximo Salto	Interface
192.168.50.0	/25	—	Fe1/0
192.168.50.192	/27	—	Fe2/0
192.168.50.244	/30	—	Se1/0
192.168.50.224	/28	192.168.50.246	Se1/0
192.168.50.240	/30	192.168.50.246	Se1/0
192.168.50.128	/26	192.168.50.246	Se1/0

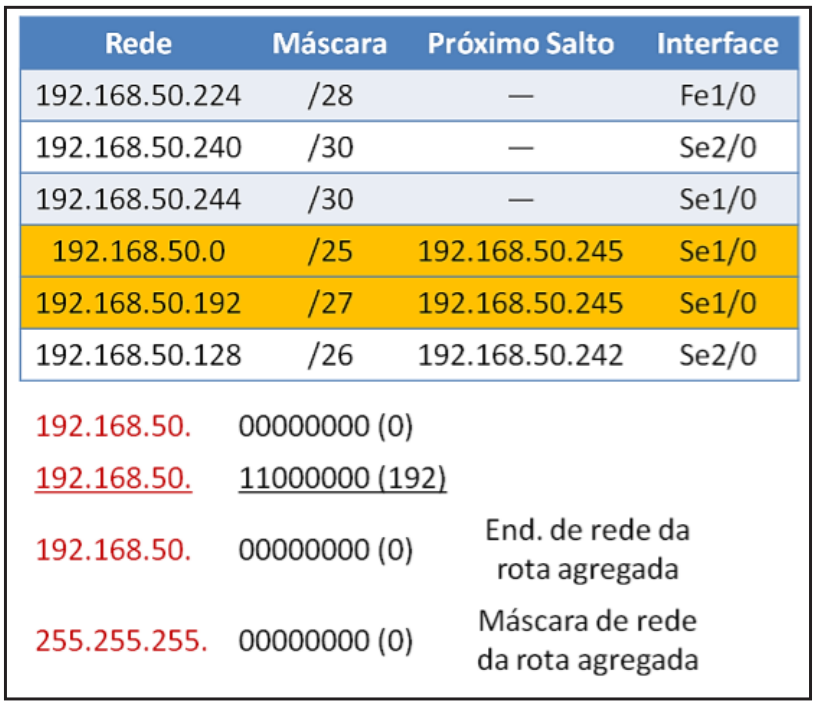
192.168.50.	11100000 (224)	
192.168.50.	11110000 (240)	
192.168.50.	10000000 (128)	
192.168.50.	10000000 (128)	End. de rede da rota agregada
255.255.255.	10000000 (128)	Máscara de rede da rota agregada

Figura 86: Tabela de rotas de R1 com a rota agregada

Rede	Máscara	Próximo Salto	Interface
192.168.50.0	/25	—	Fe1/0
192.168.50.192	/27	—	Fe2/0
192.168.50.244	/30	—	Se1/0
192.168.50.128	/25	192.168.50.246	Se1/0

A Figura 87 ilustra que a quarta e quinta entradas na tabela de rotas de R2 podem ser agregadas, porque ambas possuem o mesmo "próximo salto" – no caso, 192.168.50.245. A nova tabela de rotas de R2, com a rota agregada substituindo as entradas, é ilustrada pela Figura 88.

Figura 87: Agregação de rotas na tabela de rotas de R2



Rede	Máscara	Próximo Salto	Interface
192.168.50.224	/28	—	Fe1/0
192.168.50.240	/30	—	Se2/0
192.168.50.244	/30	—	Se1/0
192.168.50.0	/25	192.168.50.245	Se1/0
192.168.50.192	/27	192.168.50.245	Se1/0
192.168.50.128	/26	192.168.50.242	Se2/0

Figura 88: Tabela de rotas de R2 com a rota agregada

Rede	Máscara	Próximo Salto	Interface
192.168.50.224	/28	—	Fe1/0
192.168.50.240	/30	—	Se2/0
192.168.50.244	/30	—	Se1/0
192.168.50.0	/24	192.168.50.245	Se1/0
192.168.50.128	/26	192.168.50.242	Se2/0

A Figura 89 ilustra que as quatro últimas entradas na tabela de rotas de R3 podem ser agregadas, porque ambas possuem o mesmo "próximo salto" – no caso, 192.168.50.241. A nova tabela de rotas de R3, com a rota agregada substituindo as entradas, é ilustrada pela Figura 90.

Figura 89: Agregação de rotas na tabela de rotas de R3

Rede	Máscara	Próximo Salto	Interface
192.168.50.128	/26	—	Fe1/0
192.168.50.240	/30	—	Se1/0
192.168.50.0	/25	192.168.50.241	Se1/0
192.168.50.192	/27	192.168.50.241	Se1/0
192.168.50.224	/28	192.168.50.241	Se1/0
192.168.50.244	/30	192.168.50.241	Se1/0
192.168.50.	00000000 (0)		
192.168.50.	11000000 (192)		
192.168.50.	11100000 (224)		
192.168.50.	11110100 (244)		
192.168.50.	00000000 (0)	End. de rede da rota agregada	
255.255.255.	00000000 (0)	Máscara de rede da rota agregada	

Figura 88: Tabela de rotas de R2 com a rota agregada

Rede	Máscara	Próximo Salto	Interface
192.168.50.128	/26	—	Fe1/0
192.168.50.240	/30	—	Se1/0
192.168.50.0	/24	192.168.50.241	Se1/0

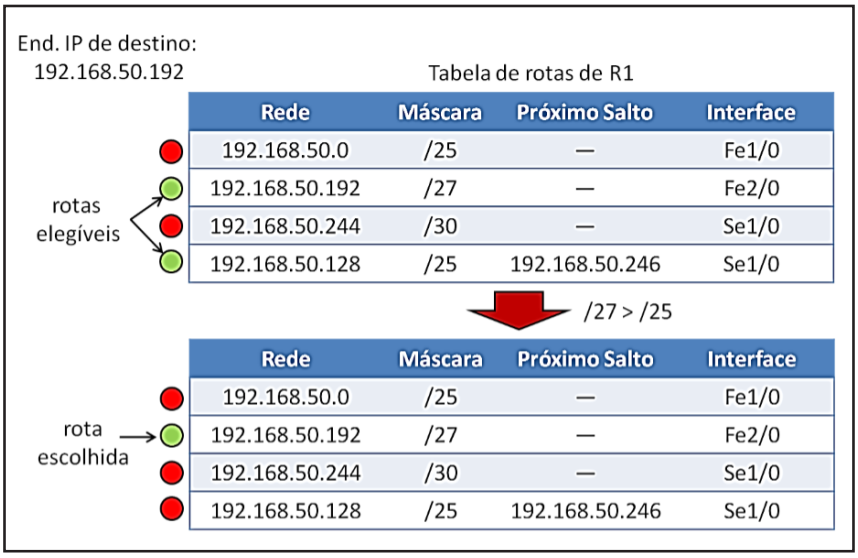
Regra de Compatibilidade com o Prefixo mais Longo

O algoritmo executado por um roteador para o roteamento de um datagrama possui uma etapa relacionada à resolução de conflitos, que ocorrem quando há duas ou mais rotas elegíveis. Em particular, a rota a ser escolhida dentre várias elegíveis é fruto da aplicação da regra de compatibilidade com o prefixo mais longo. Esta regra diz que, havendo duas ou mais rotas elegíveis, a rota a ser escolhida será aquela que tiver a maior máscara. A ideia subjacente à regra de compatibilidade com o prefixo mais longo é privilegiar a rota mais específica dentre todas as elegíveis.

A Figura 91 ilustra o que acontece quando R1 se depara com um datagrama endereçado a 192.168.50.194. Trata-se de um datagrama endereçado a um computador da sub-rede 192.168.50.192/27. Porém, quando o roteador computa o AND bit-a-bit entre 192.168.50.194 e as máscaras de sub-rede em cada uma das entradas da sua tabela de rotas, há duas

correspondências com os respectivos endereços de rede. Há então, em princípio, duas rotas elegíveis àquele destino, que correspondem à segunda e à quarta entradas da tabela de rotas de R1. A regra da compatibilidade com o prefixo mais longo é aplicada, e verifica-se que a máscara da segunda rota (/27) é maior que a máscara da quarta (/25). Logo, a rota escolhida é aquela que corresponde à segunda entrada.

Figura 91: Aplicação da regra de compatibilidade com o prefixo mais longo em R1



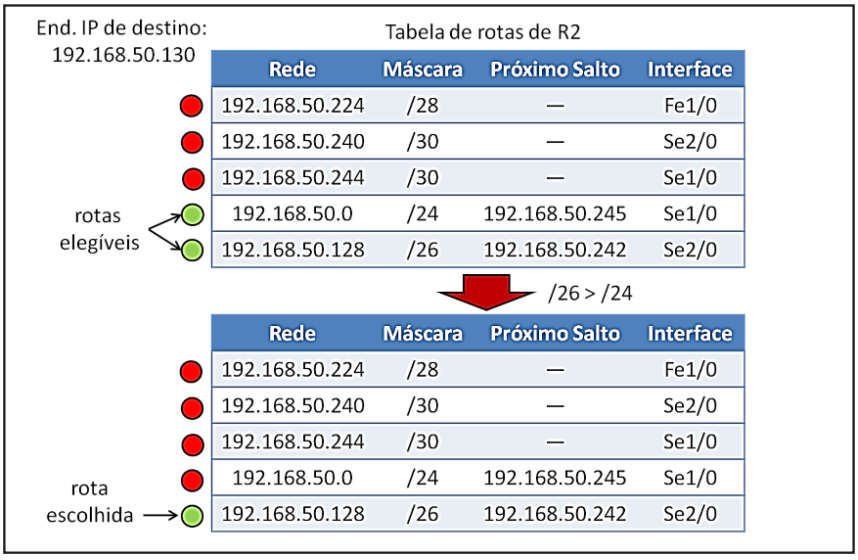
Rede	Máscara	Próximo Salto	Interface
192.168.50.0	/25	—	Fe1/0
192.168.50.192	/27	—	Fe2/0
192.168.50.244	/30	—	Se1/0
192.168.50.128	/25	192.168.50.246	Se1/0

Rede	Máscara	Próximo Salto	Interface
192.168.50.0	/25	—	Fe1/0
192.168.50.192	/27	—	Fe2/0
192.168.50.244	/30	—	Se1/0
192.168.50.128	/25	192.168.50.246	Se1/0

A Figura 92 ilustra o que acontece quando R2 se depara com um datagrama endereçado a 192.168.50.130. Trata-se de um datagrama endereçado a um computador da sub-rede 192.168.50.128/26. Porém, quando o roteador computa o AND bit-a-bit entre 192.168.50.130 e as máscaras de sub-rede em cada uma das entradas da sua tabela de rotas, há também duas correspondências com os respectivos endereços de rede. Logo, há também duas rotas elegíveis àquele destino, que correspondem às duas últimas entradas da tabela de rotas de R2.

A regra da compatibilidade com o prefixo mais longo é aplicada, e verifica-se que a máscara da última rota (/26) é maior que a máscara da penúltima (/24). Logo, a rota escolhida é aquela que corresponde à última entrada.

Figura 92: Aplicação da regra de compatibilidade com o prefixo mais longo em R2



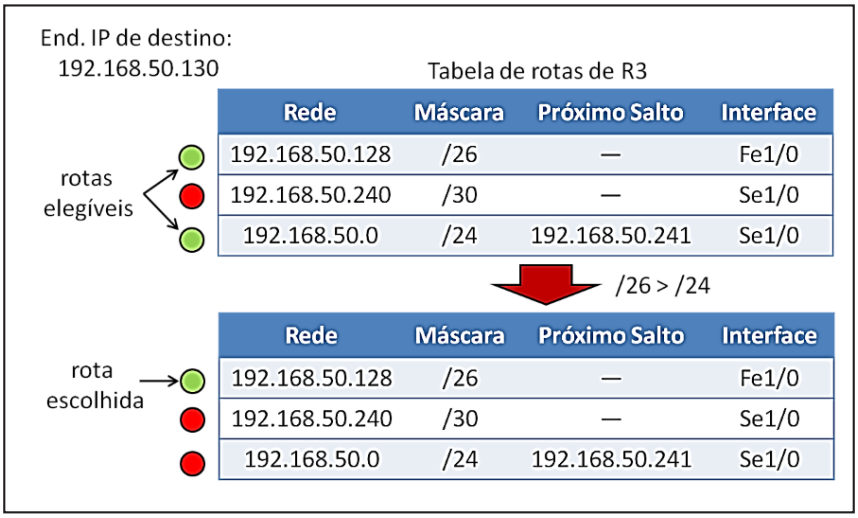
Rede	Máscara	Próximo Salto	Interface
192.168.50.224	/28	—	Fe1/0
192.168.50.240	/30	—	Se2/0
192.168.50.244	/30	—	Se1/0
192.168.50.0	/24	192.168.50.245	Se1/0
192.168.50.128	/26	192.168.50.242	Se2/0

Rede	Máscara	Próximo Salto	Interface
192.168.50.224	/28	—	Fe1/0
192.168.50.240	/30	—	Se2/0
192.168.50.244	/30	—	Se1/0
192.168.50.0	/24	192.168.50.245	Se1/0
192.168.50.128	/26	192.168.50.242	Se2/0

A Figura 93 ilustra o que acontece quando R3 se depara com um datagrama endereçado a 192.168.50.130. Conforme visto há pouco, trata-se de um datagrama endereçado a um computador da sub-rede 192.168.50.128/26. Porém, quando o roteador computa o AND bit-a-bit entre 192.168.50.130 e as máscaras de sub-rede em cada uma das entradas da sua tabela de rotas, há também duas correspondências com os respectivos endereços de rede.

Logo, há também duas rotas elegíveis àquele destino, que correspondem às primeira e terceira entradas da tabela de rotas de R3. A regra da compatibilidade com o prefixo mais longo é aplicada, e verifica-se que a máscara da primeira rota (/26) é maior que a máscara da terceira (/24). Logo, a rota escolhida é aquela que corresponde à primeira entrada.

Figura 93: Aplicação da regra de compatibilidade com o prefixo mais longo em R3



NAT

Foi visto há pouco que a adoção do CIDR para alocar endereços IPv4 em quantidades mais próximas da demanda dos clientes é uma das soluções paliativas para a escassez de endereços. A outra solução paliativa atende pelo nome de NAT (*Network Address Translation*). Especificado pela RFC 1631, o NAT é muito popular em redes domésticas de clientes de um ISP (*Internet Service Provider*). A ideia básica é fazer com que o ISP atribua à rede de cada cliente um único endereço IP válido. Por outro lado, dentro da rede de cada cliente cada computador utiliza um endereço IP inválido. A RFC 1918 define que os endereços inválidos estão compreendidos nas seguintes redes:

• 10.0.0.0/8, que compreende 16.777.214 endereços úteis – que vão de 10.0.0.1 a 10.255.255.254;

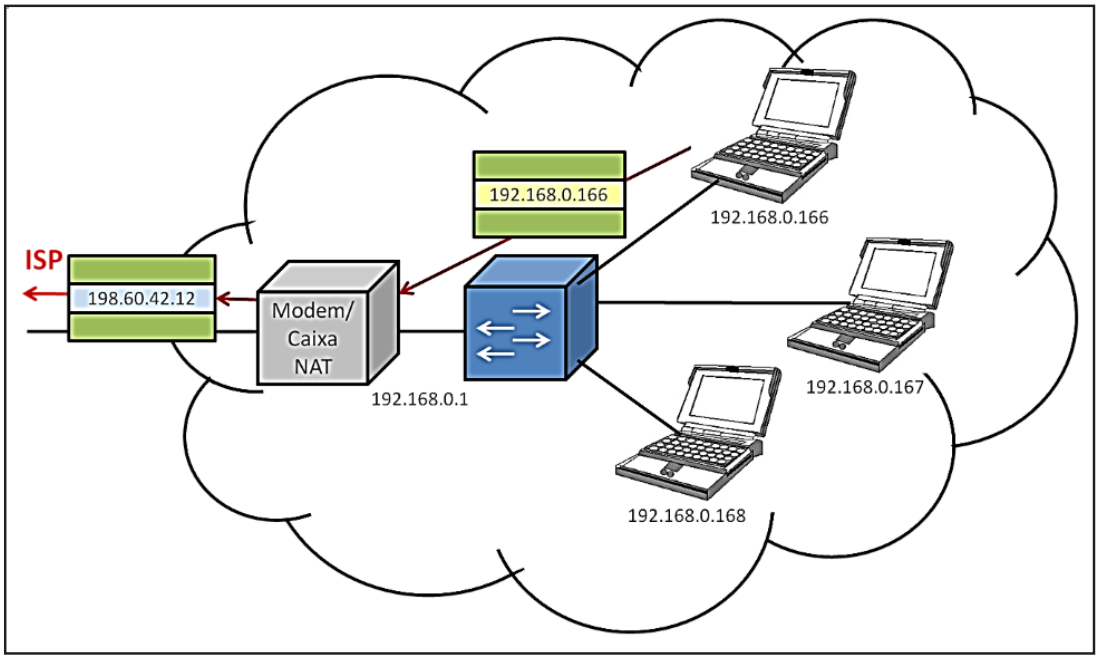
• 172.16.0.0/12, que compreende 1.048.574 endereços úteis – que vão de 172.16.0.1 a 172.31.255.254;

• 192.168.0.0/16, que compreende 65.534 endereços úteis – que vão de 192.168.0.1 a 192.168.255.254.

Um fato aparentemente contraditório é que nenhum datagrama contendo endereços inválidos circula pela internet. De fato, quando um datagrama sai da rede do cliente em direção ao ISP ocorre uma tradução de endereços realizada pela caixa NAT – função tipicamente assumida pelo *modem* fornecido pelo ISP. Considere-se, por exemplo, que na Figura 94 o computador que

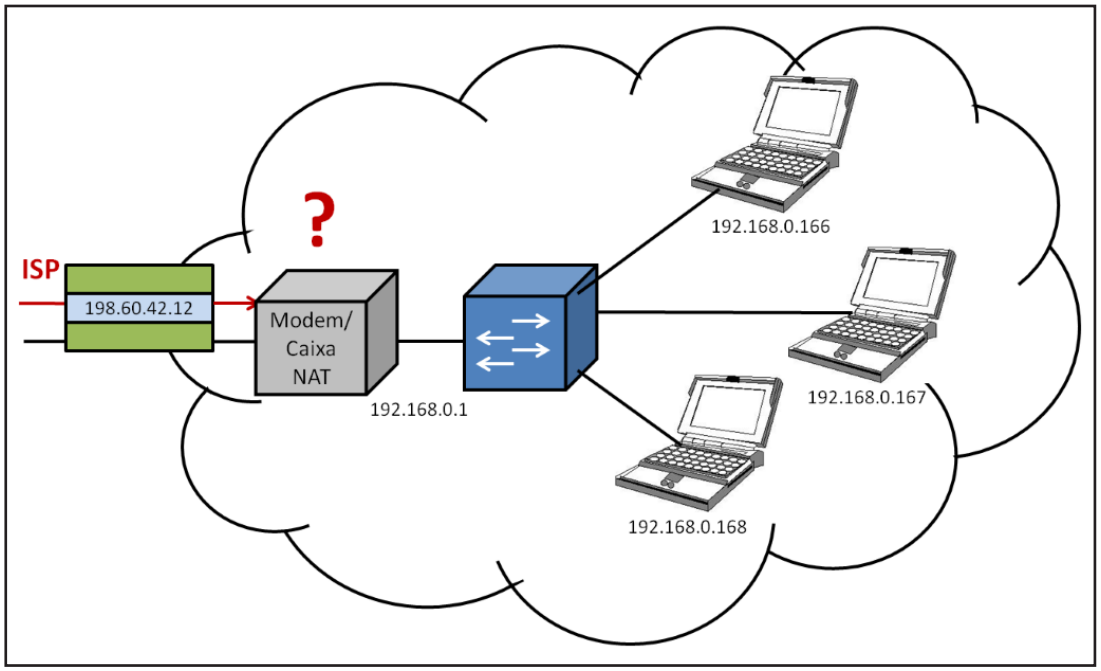
possui o endereço inválido 192.168.0.166 envia um datagrama que encapsula um segmento TCP – que, por sua vez, encapsula uma solicitação HTTP endereçada a um servidor localizado fora da rede doméstica do cliente. Quando este datagrama chega à caixa NAT, o endereço IP inválido é substituído pelo endereço válido daquele *modem* (198.60.42.12).

Figura 94: Tradução de endereço no envio de um datagrama



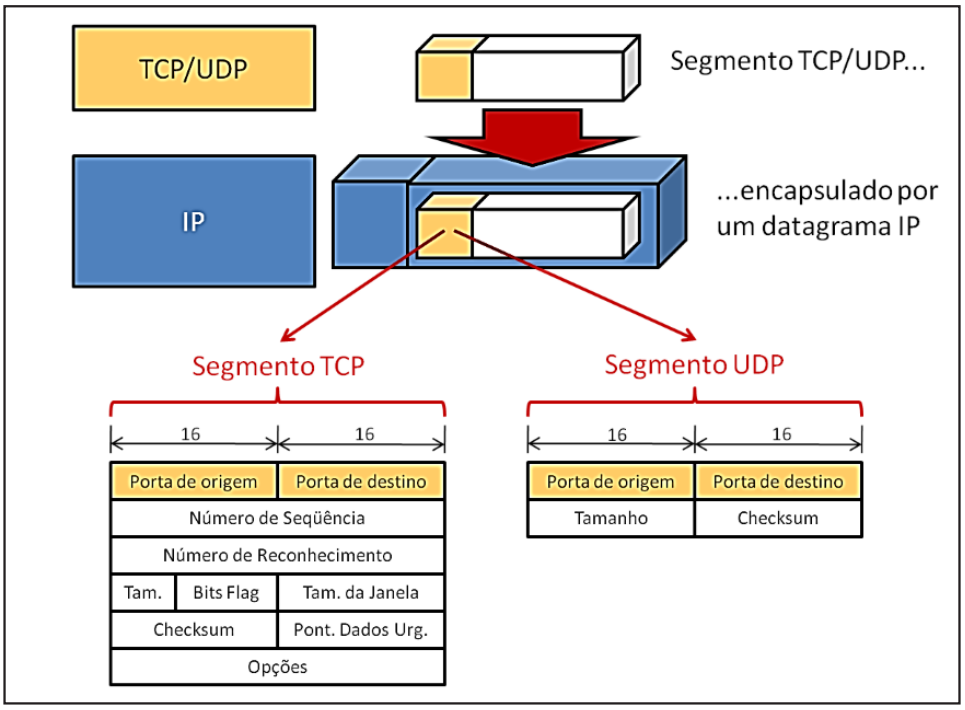
O fato de que todos os endereços inválidos da rede doméstica do cliente são traduzidos para um mesmo endereço válido faz com que um ISP possa ter N computadores conectados sem que seja necessário obter N endereços IPv4 junto à entidade de registro regional. Por outro lado, como a caixa NAT realiza a tradução do seu endereço válido para o endereço inválido correto, quando ela recebe o datagrama que encapsula o segmento TCP que, por sua vez, encapsula (parte da) resposta HTTP à solicitação anterior? Este questionamento é ilustrado pela Figura 95.

Figura 95: Tradução de endereço na recepção de um datagrama



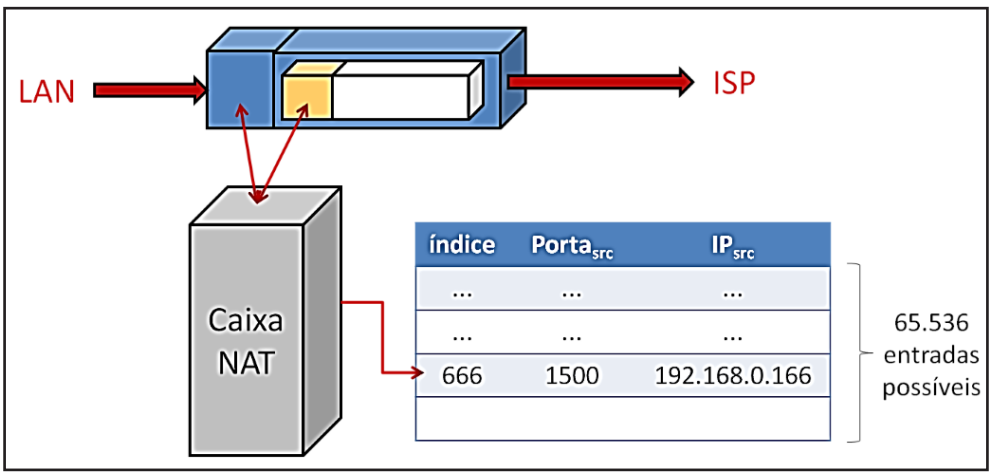
Para responder ao questionamento do parágrafo anterior, é necessário investigar a fundo o processo de tradução de endereços no NAT. A Figura 96 relembra que datagramas encapsulam segmentos, quer sejam TCP ou UDP.

Figura 96: Encapsulamento de segmentos em um datagrama



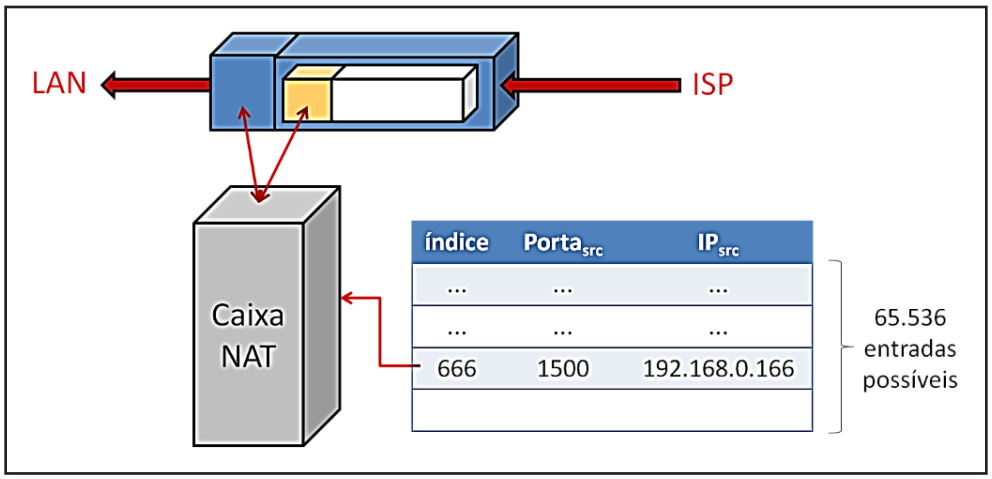
Conforme ilustrado pela Figura 97, a caixa NAT mantém internamente uma tabela indexada por um índice (chave primária) cujo tamanho é de 16 bits – o que leva a tabela a ter no máximo 65.536 entradas. Quando o datagrama ilustrado pela Figura 94 chega à caixa NAT, seu endereço IP de origem (192.168.0.166) é substituído pelo endereço da interface externa do *modem* (198.60.42.12), conforme comentado há pouco. Além disso, o valor da porta de origem do segmento encapsulado (1500) é substituído pelo índice 666. Os valores originais da porta e do endereço IP de origem são armazenados na tabela na entrada identificada pelo índice 666. Por haver também uma tradução de porta, o NAT é, por vezes, denominado NAPT (*Network and Port Translation*).

Figura 97: Detalhe da tradução de endereço no envio de um datagrama



A Figura 98 ilustra o que acontece quando o datagrama em resposta àquele da figura anterior chega à caixa NAT. Ela inspeciona o campo de dados do datagrama em busca da porta de destino do segmento encapsulado. A caixa NAT utiliza aquele valor para consultar sua tabela e encontrar os valores originais de endereço IP e porta. Assim, o datagrama segue para seu destino correto.

Figura 98: Detalhe da tradução de endereço na recepção de um datagrama



Alguns fatos merecem destaque. O primeiro deles é que, muito embora a tabela mantida pela caixa NAT possua 65.536 entradas possíveis, esta quantidade é tipicamente insuficiente para rastrear todas as comunicações. Assim, as entradas mais antigas são substituídas por outras mais novas. O segundo fato de destaque é que, no processo de tradução de endereços ilustrados pelas Figuras 97 e 98, os campos de *checksum* do segmento e do datagrama que o encapsula são recalculados pela caixa NAT, haja vista que tanto o endereço IP quanto a porta são alterados por ela.

Embora bastante popular, o NAT coleciona algumas críticas. Uma delas é que ele acabou com a ideia de que cada computador na internet possuiria um endereço IP diferente dos demais. Assim, um endereço IP deixou de identificar inequivocamente um único computador em todo o mundo. Outra crítica é a de que o NAT viola uma regra fundamental da distribuição de protocolos em níveis, na medida em que a caixa NAT inspeciona o cabeçalho do segmento. Há também a crítica que afirma que, uma vez que a caixa NAT sofrer uma pane, todas as comunicações estarão comprometidas.

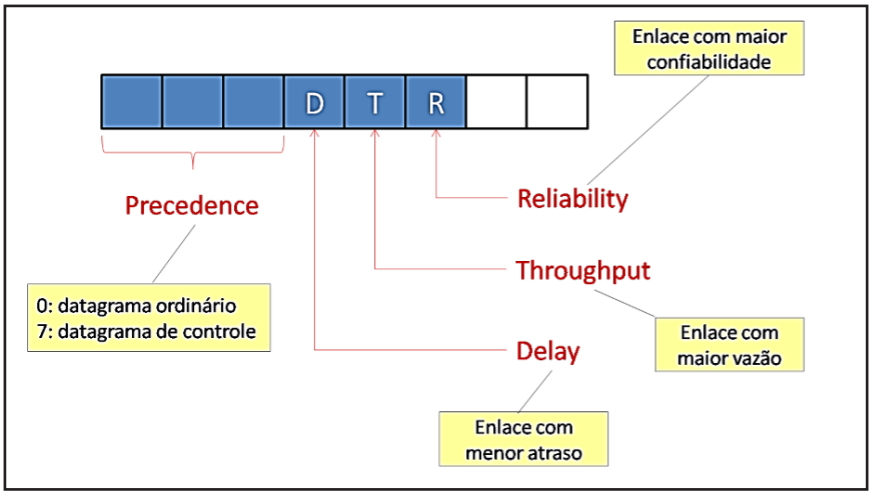
Diferenciação de Datagramas

Na Figura 99, os oito bits que compõem o campo ToS (Type of Service) foram concebidos para que um roteador promovesse um tratamento diferenciado a um datagrama em relação a outros. Esta diferenciação seria necessária porque enquanto há datagramas carregando dados não sensíveis ao atraso e variação do atraso, tais como fragmentos de arquivos HTML, há outros datagramas cujos dados são sensíveis àqueles parâmetros, tais como amostras de áudio e vídeo, por exemplo. Porém, há que ser salientado que o estudo deste campo tem caráter meramente histórico: tipicamente todos os oito bits são iguais a 0 (zero) e o campo é solenemente ignorado por um roteador.

Conforme ilustra a Figura 99, os primeiros três bits mais significativos são denominados *precedence*, e seriam utilizados para classificar os datagramas quanto ao seu nível de prioridade. Nos extremos, datagramas ordinários possuiriam um valor para o precedence igual a 0, enquanto datagramas

que encapsulam mensagens ICMP (um protocolo que será visto adiante) possuiriam um valor para o *precedence* igual a 7. Os próximos três bits são as *flags* D (*Delay* - Atraso), T (*Throughput* - Vazão) e R (*Reliability* - Confiabilidade). Estas *flags* seriam utilizadas como critérios de desempate quando houvesse duas ou mais rotas elegíveis. Os dois últimos bits do ToS não possuem função, sendo inutilizados.

Figura 99: Campo ToS - *Type of Service*



Havendo duas ou mais rotas elegíveis, um datagrama com a *flag* D ativada seria roteado por meio daquela cujo enlace proporcionasse o menor atraso. Um enlace cujo atraso fosse de 100 ms seria preferido em relação a outro cujo atraso fosse de 100 ms, por exemplo. Por outro lado, um datagrama com a *flag* T ativada seria roteado por meio de uma rota cujo enlace proporcionasse a maior vazão. Por exemplo, um enlace *Fast Ethernet* (cuja taxa nominal de transmissão é de 100 Mbps) seria preferido em relação a um enlace T1 – cuja taxa nominal de transmissão é de 1,544 Mbps.

Uma possível confusão seria estabelecer uma equivalência entre maior vazão e menor atraso, o que nem sempre é verdadeiro: um enlace via satélite, por exemplo, pode ter uma taxa de transmissão altíssima. Porém, dada a enorme distância a ser percorrida, o atraso experimentado por um datagrama transmitido por meio deste enlace também seria alto. Por fim, um datagrama com a *flag* R ativada seria roteado através de uma rota cujo enlace proporcionasse maior confiabilidade. Neste contexto, um enlace cabeado seria preferido em relação a um enlace não-cabeado (*wireless*).

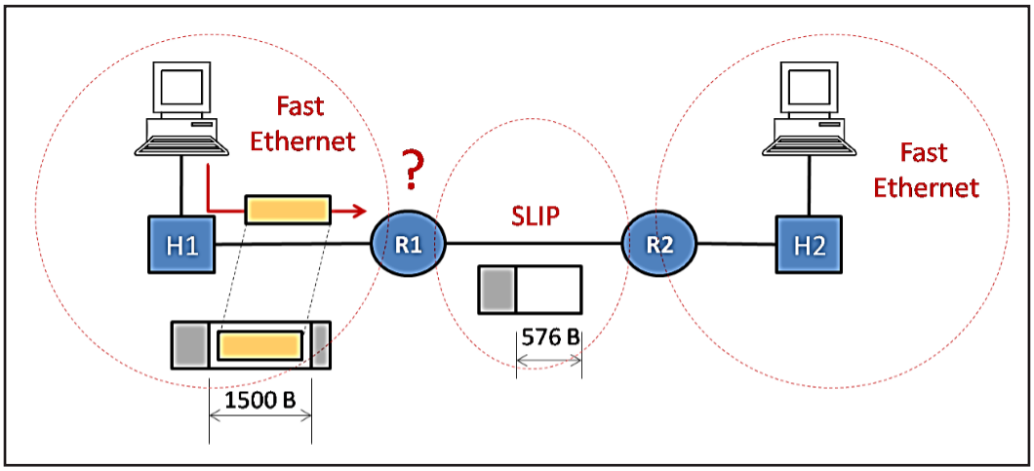
Conforme dito anteriormente, o campo ToS não é utilizado. As razões para isso estão relacionadas à falta de consenso sobre o que vem a ser, de fato, priorizar um datagrama em relação a outros. Seria roteá-los imediatamente, na frente dos demais que estão na fila? Seria intercalar o roteamento entre datagramas prioritários e outros não prioritários? Caso fosse, qual seria a forma (razão) de intercalação? Um datagrama prioritário a cada outro não prioritário? Um datagrama prioritário a cada dois não prioritários? Além disso, o que impediria um computador de marcar todos os seus datagramas com um valor para o *precedence* igual a 7, independentemente das características dos dados transportados, só para conseguir melhor tratamento? De fato, havia muitas questões que permaneciam sem resposta. E ainda há.

Fragmentação de datagramas

É fato que o TCP particiona a cadeia de dados enviada por uma aplicação e transmite cada uma das partes encapsulada em um segmento. Uma das razões por trás deste particionamento é o fato de que cada tecnologia utilizada no nível de enlace possui um limite máximo para a quantidade de dados que pode ser transmitida em seus quadros (denominação dada às PDUs do nível de enlace). Este limite máximo é denominado MTU (*Maximum Transmission Unit*). O valor da MTU define, em termos práticos, o tamanho máximo do datagrama IP. Por exemplo, a MTU de um quadro *Fast Ethernet* é de 1.500 bytes, o que significa que um computador que utiliza uma interface Fast Ethernet pode construir um datagrama de, no máximo, 1.500 Bytes – incluindo-se aí o cabeçalho, cujo tamanho médio é de 20 bytes.

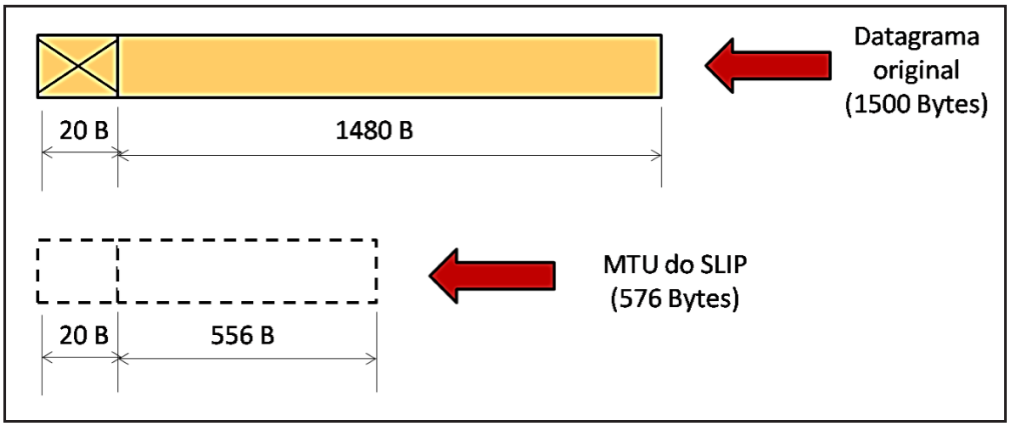
O que acontece quando um datagrama é roteado através de uma interface cuja MTU é menor do que o tamanho daquele datagrama? Na Figura 100, o datagrama foi construído obedecendo-se à MTU do *Fast Ethernet*, que é de 1.500 Bytes. Porém, o roteador R1 enfrenta um dilema ao constatar que o datagrama tem que ser encaminhado para R2 através de um enlace ponto-a-ponto serial SLIP (*Serial Line Interface Protocol*), cuja MTU é de 576 bytes. O datagrama simplesmente não "cabe" no campo de dados de um quadro SLIP. É em situações assim que ocorre a fragmentação do datagrama original em datagramas menores.

Figura 100: Necessidade de fragmentação de um datagrama



Para entender como é feita a fragmentação, há de considerar-se o tamanho do cabeçalho e do campo de dados tanto do datagrama original quanto de cada fragmento. Conforme ilustra a Figura 101, um datagrama de 1.500 bytes possui um campo de dados de 1.480 bytes e um cabeçalho de 20 bytes – este é o tamanho típico do cabeçalho IP, haja vista que o campo de opções é raramente utilizado. Por outro lado, considerando-se que cada fragmento é também um datagrama e que a MTU do SLIP é de 576 bytes, conclui-se que cada fragmento poderia conter inicialmente 576 bytes, no máximo. Estas informações levam à conclusão de que, no presente caso, serão necessários $^{1.480}/_{556} = 2{,}7 \cong 3$ (três) fragmentos para transportar todos os 1.480 bytes do datagrama original.

Figura 101: Datagrama original e tamanho máximo dos fragmentos



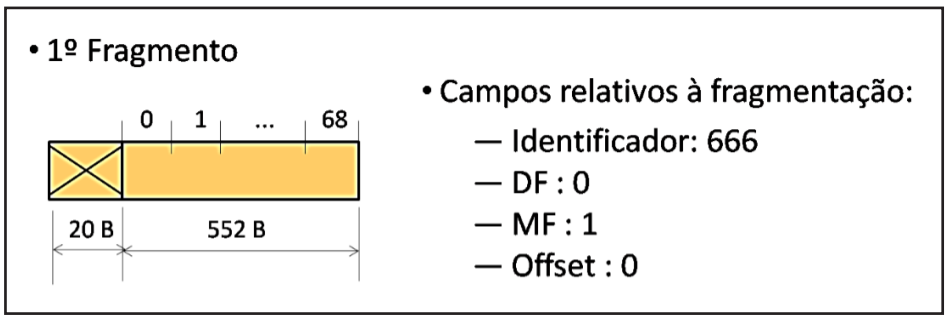
A unidade de fragmentação de datagramas é de 8 (oito) bytes. Isto significa que todos os fragmentos, à exceção do último, possuem campos de dados compostos por múltiplos de 8 bytes. Portanto, cada fragmento pode conter até $556/8 = 69{,}5 = 69$ conjuntos de 8 bytes.

Um datagrama possui três campos relacionados à fragmentação, que são ilustrados pela Figura 53. O Identificador é um campo cujo valor, seja ele qual for, deve ser o mesmo para todos os fragmentos a fim de estabelecer que eles são relacionados a um mesmo datagrama original. Seguem-se àquele campo duas *flags,* denominadas DF (*Don't Fragment*) e MF (*More Fragment*). Conforme indicam suas respectivas denominações, a *flag* DF é ativada caso deseje-se que um datagrama original não seja fragmentado – na imensa maioria das vezes, esta *flag* está desativada; a *flag* MF é ativada para todos os fragmentos de um datagrama original – à exceção do último fragmento, naturalmente.

O Deslocamento de Fragmentação, também denominado *Offset*, especifica a numeração do primeiro conjunto de 8 bytes dos dados que compõem o campo de dados de um fragmento – este campo pode ser entendido como uma espécie de número de sequência aplicado aos conjuntos de 8 bytes que compõem as unidades de fragmentação.

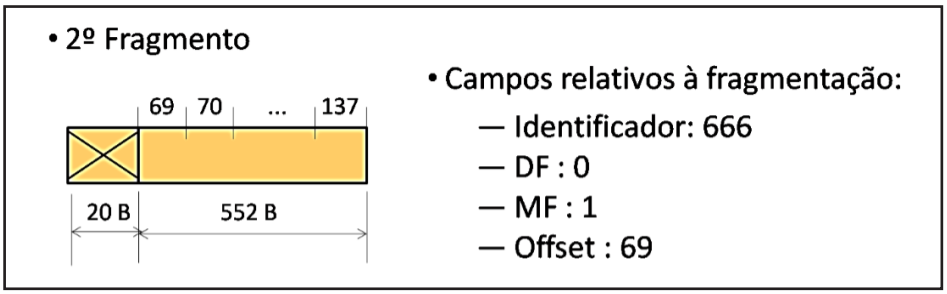
A Figura 102 ilustra detalhes do primeiro fragmento, exibindo seus campos relativos à fragmentação. Seu tamanho (572 bytes) é menor do que os 576 bytes ilustrados pela Figura 101, pelo fato de a fragmentação ocorrer em unidades de 8 bytes.

Figura 102: Campos de fragmentação do primeiro fragmento



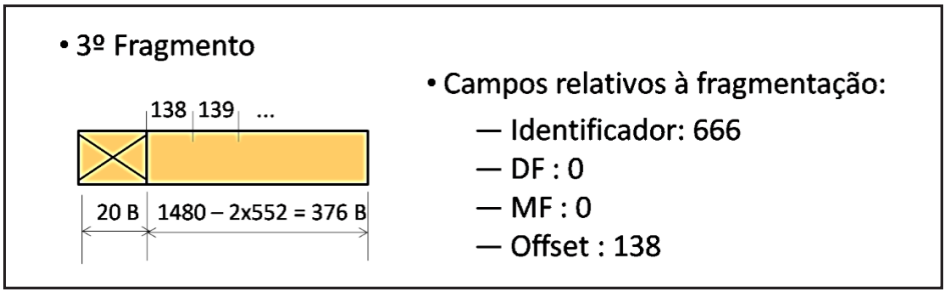
A Figura 103 ilustra detalhes do segundo fragmento, exibindo seus campos relativos à fragmentação. Seu tamanho é idêntico ao do primeiro fragmento.

Figura 103: Campos de fragmentação do segundo fragmento



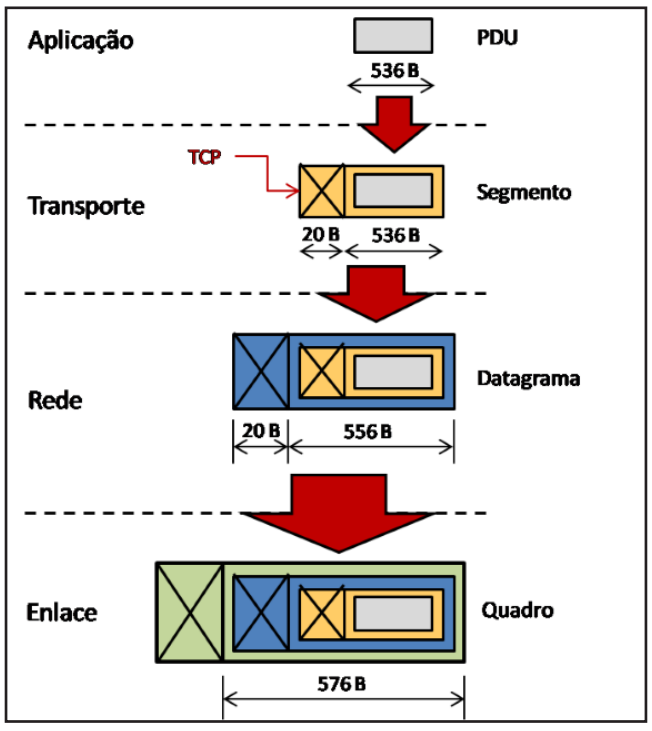
A Figura 104 ilustra detalhes do terceiro e último fragmentos, exibindo seus campos relativos à fragmentação. Seu tamanho é menor do que o de seus antecessores.

Figura 104: Campos de fragmentação do terceiro fragmento



A menor MTU na Internet é 576 bytes. Assim, conforme ilustra a Figura 77, a fragmentação pode ser evitada ao se transmitir um datagrama que encapsule um segmento cujo MSS (*Maximum Segment Size*) seja igual 536 bytes – esta denominação é inadequada porque não define o tamanho máximo de um segmento, mas o tamanho máximo de seu campo de dados. Tipicamente, porém, computadores transmitem datagramas obedecendo a MTU de sua interface local.

Figura 105: Fragmentação é evitada utilizando MSS igual a 536 bytes

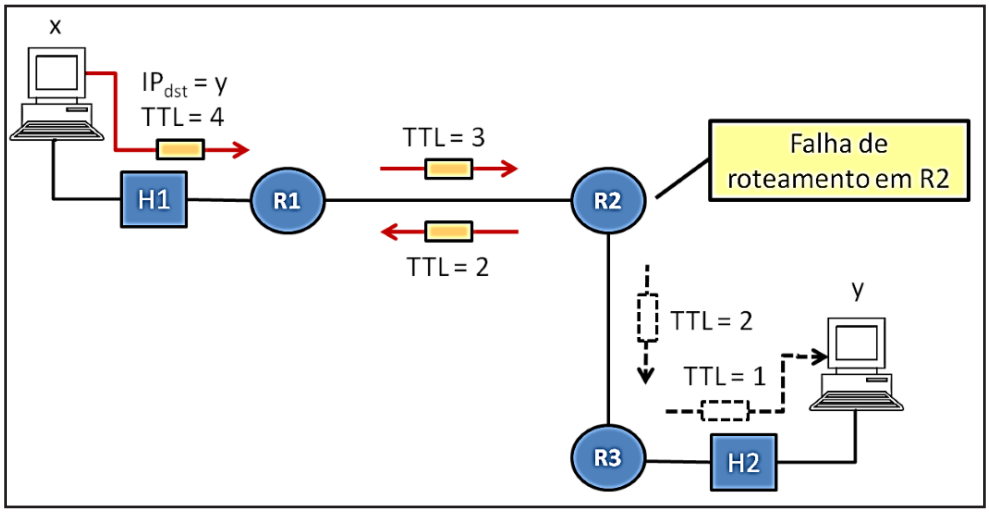


O datagrama original deve ser remontado a partir dos seus fragmentos, de forma que a PDU encapsulada por ele possa ser entregue ao protocolo acima do IP. Esta remontagem não está a cargo de nenhum roteador. De fato, roteadores são equipamentos que, normalmente, trabalham sob grande intensidade. Além de suas atuais responsabilidades, eles precisariam armazenar antecipadamente todos os fragmentos de um datagrama e verificar se a remontagem do datagrama original é possível. Assim, a remontagem de um datagrama original é feita pelo seu destinatário final.

Aliviando os Efeitos em Laços de Roteamento

Erros eventuais na configuração de um roteador ou um simples mau-funcionamento podem dar origem a um laço de roteamento, o que se caracteriza pelo fato de que um datagrama enviado por um roteador retorna a ele, de alguma forma. O campo TTL (*Time-To-Live*) é utilizado para fazer com que datagramas não permaneçam indefinidamente em um laço de roteamento. O TTL é um contador que é decrementado por cada roteador que processa o datagrama ao longo do seu caminho. Quando o TTL chega a 0 (zero), o datagrama é descartado pelo roteador. Como exemplo, a Figura 106 ilustra que R2 envia datagramas endereçados a y para R3, ao invés de seguir a prática recomendada de enviá-los para R3.

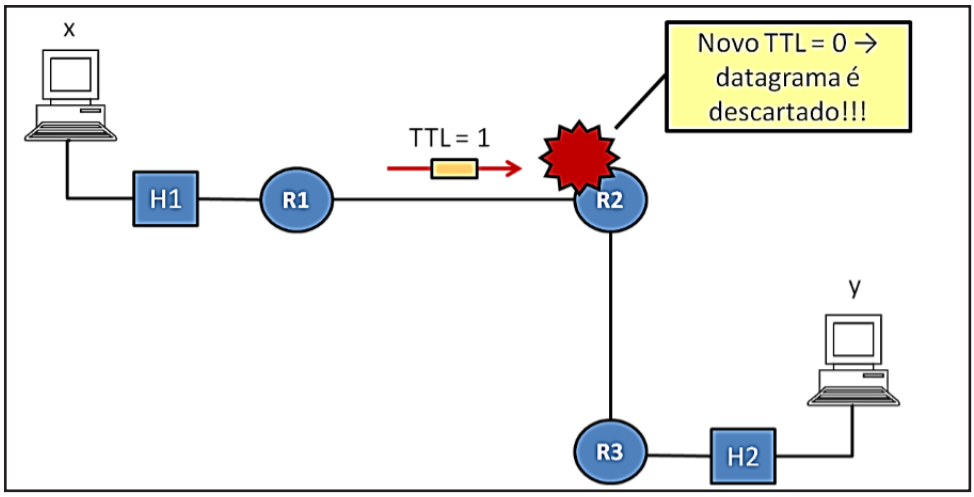
Figura 106: Exemplo de laço de roteamento



Se não existisse o TTL, o datagrama em questão permaneceria indefinidamente no laço de roteamento formado pelos roteadores R1 e R2, haja vista que o primeiro possui o comportamento (adequado) de enviar datagramas endereçados a y para R2.

A Figura 107 ilustra que R2, ao decrementar o TTL do datagrama, verifica que este é igual a 0 (zero) e o descarta. Assim, o TTL é uma forma de aliviar os efeitos da formação do laço de roteamento – enlaces não ficam permanentemente ocupados com um datagrama envolto em um laço, e roteadores não desperdiçam indefinidamente seus recursos processando aquele datagrama. Porém, há de se considerar que o TTL não evita a formação de laços de roteamento.

Figura 107: Datagrama é descartado quando o TTL chega a 0 (zero)



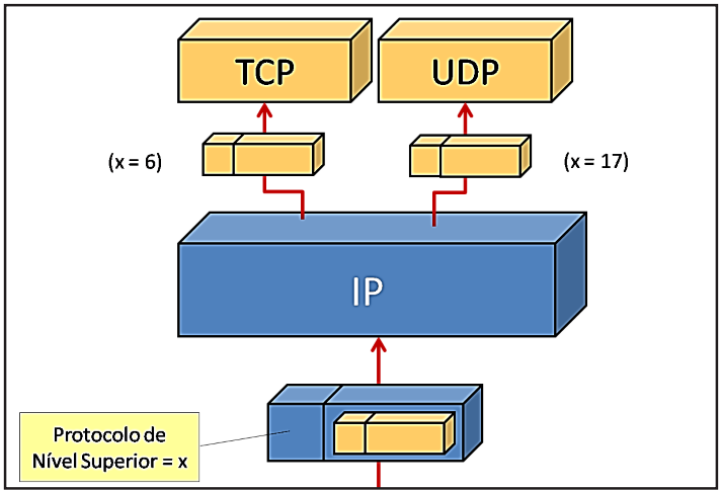
O decremento do TTL expõe outra etapa do processamento realizado pelos roteadores. Além de varrerem suas tabelas de rotas para escolher uma rota, dentre potencialmente várias rotas elegíveis, eles devem ainda recalcular o *checksum* de cada datagrama. Isto ocorre porque o *checksum* é calculado levando-se em conta todo o datagrama – cujo cabeçalho foi alterado pelo decremento do TTL por parte do roteador.

Especificando a Quem Entregar o Conteúdo Encapsulado

O IP é um protocolo do nível de rede, o que implica que ele transporta dados de protocolos localizados acima dele. A questão é: quando o *driver* IP do destinatário recebe um datagrama, como ele sabe para quem entregar o conteúdo encapsulado no seu campo de dados? Até então, foi visto que datagramas IP encapsulam segmentos TCP e UDP. Na verdade, há outras PDUs que são encapsuladas por datagramas IP, conforme veremos. De qualquer forma, a questão permanece: para quem o *driver* IP entregará o conteúdo encapsulado no seu campo de dados? Certamente o driver IP não investigará o campo de dados do datagrama, pois isto seria uma violação do modelo arquitetônico do TCP/IP.

A resposta para esta questão está no campo protocolo de nível superior, que contém um número que identifica o protocolo cuja PDU está encapsulada no datagrama. Conforme ilustra a Figura 108, os valores referentes ao TCP e ao UDP são 6 e 17, respectivamente.

Figura 107: Datagrama é descartado quando o TTL chega a 0 (zero)

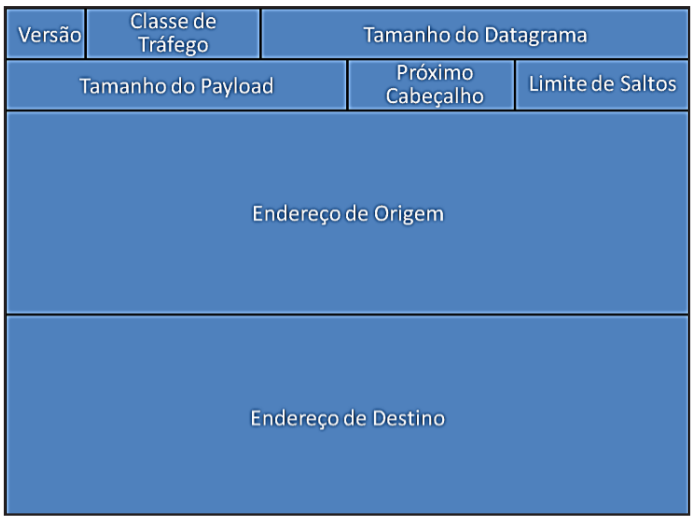


IPv6 – Um Novo Começo

Conforme citado anteriormente, a solução para o problema da escassez de endereços passa pela substituição do IPv4 pelo IPv6. Diferentemente do que possa parecer, o IPv6 não é apenas um IPv4 "anabolizado" – com um campo de endereços maior. Ele é um protocolo inteiramente diferente. De fato, a compreensão de todos os aspectos do IPv6 demandaria um capítulo dedicado inteiramente a ele. O que será visto a seguir é apenas uma introdução.

A Figura 109 ilustra o formato do cabeçalho base de um datagrama IPv6. Este cabeçalho é, propositalmente, mais simples que o cabeçalho IPv4, porque especificidades devem ser cobertas por outros cabeçalhos. Assim, há cabeçalhos IPv6 específicos para segurança (através do uso da autenticação e privacidade), para inserção de uma rota pré-definida, para fragmentação, etc. Para saber qual é o próximo cabeçalho (caso ele exista), o roteador examina o campo Próximo Cabeçalho no cabeçalho base do datagrama IPv6. Este protocolo revisita a ideia de diferenciar datagramas em relação a outros através do campo Classe de Tráfego, que é análogo ao campo ToS do IPv4. Da mesma forma, o Limite de Saltos é um campo análogo ao TTL do IPv4.

Figura 109: Formato do cabeçalho IPv6



Os maiores campos de um datagrama IPv6 são os Endereços de Origem e Destino. Eles possuem 128 bits, ao invés dos 32 bits do endereço IPv4. Isto representa um aumento absurdo na quantidade de endereços disponíveis. Basta pensar que se tivesse acrescido apenas 1 (um) bit ao campo de endereço, a quantidade de endereços disponíveis dobraria. Exatamente pelo fato de possuírem 128 bits, a notação utilizada para representar endereços IPv4 não é adequada à representação de endereços IPv6. Por exemplo, o endereço IPv6 a seguir é representado utilizando a notação decimal separada por pontos do IPv4:

105.220.136.100.255.255.255.255.0.0.18.128.140.10.255.255

Assim, os endereços IPv6 são manipulados pelos seres humanos utilizando a notação hexadecimal separada por dois pontos. O mesmo endereço anterior é assim representado utilizando esta notação:

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

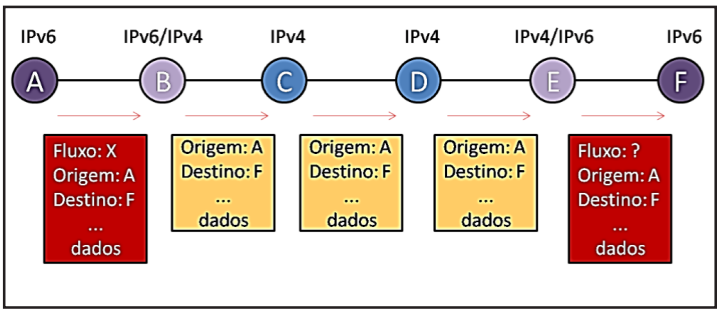
Uma estratégia denominada compressão de zeros pode ser utilizada para compactar ainda mais a representação de um endereço sob determinadas circunstâncias. Por exemplo, o endereço FF0C:0:0:0:0:0:0:B1 pode ser representado por FF0C::B1:

Atualmente, é comum sistemas operacionais já terem a capacidade de construir e entender datagramas IPv6. Mas a adoção do IPv6 requer que todos os roteadores ao longo da Internet sejam capazes de manipular tais datagramas.

Esta é a tarefa mais difícil, tanto em termos operacionais quanto financeiros. Em termos operacionais, trata-se de um novo protocolo com problemas inéditos potenciais. Em termos financeiros, é fácil imaginar que substituir o *firmware* de todos os roteadores é algo muito custoso. O consenso, porém, é de que a adoção do IPv6 é irreversível e ele vai começar das bordas da internet para o centro dela – isto é, considera-se a passagem de datagramas IPv6 por um *backbone* IPv4. Neste contexto, as estratégias de migração do IPv4 para o IPv6 mais populares atendem pelas denominação de Pilha Dupla e Tunelamento.

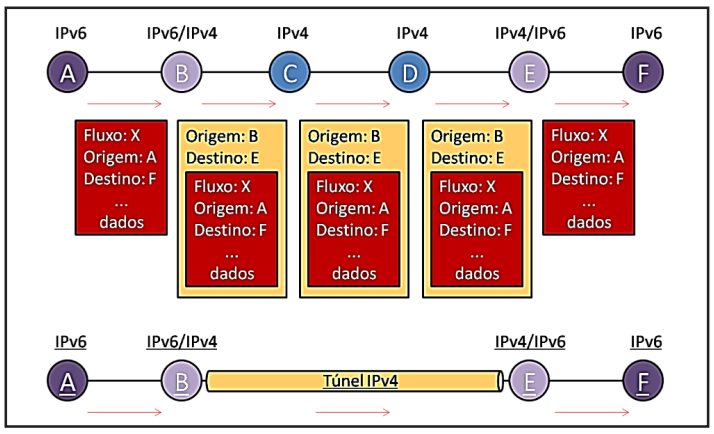
A estratégia de Pilha Dupla, ilustrada pela Figura 110, é baseada na conversão inicial de um datagrama IPv6 em um datagrama IPv4 e na conversão final daquele datagrama IPv4 em um outro datagrama IPv6. Na figura, a conversão inicial é realizada pelo roteador B, ao passo que a conversão final é realizada pelo roteador E. Esta estratégia requer que endereços IPv4 sejam encapsulados nos 32 últimos bits de endereços IPv6 – os 96 bits anteriores seriam iguais 0s (zeros). Uma crítica comum a ela é que, na conversão inicial de datagramas, os valores de alguns campos IPv6 são perdidos devido ao fato de eles simplesmente não existirem no IPv4.

Figura 110: Estratégia de Pilha Dupla



A Figura 111 ilustra a estratégia alternativa à Pilha Dupla, denominada Tunelamento. Ela é baseada no encapsulamento inicial de um datagrama IPv6 por um datagrama IPv4, e no posterior desencapsulamento. Na figura, o roteador B encapsula o datagrama em um datagrama IPv4, enquanto o roteador E realiza o desencapsulamento. A estratégia de Tunelamento recebe esta denominação porque se pode compreender (utilizando certo nível de abstração) que o datagrama IPv6, uma vez encapsulado em um datagrama IPv4, trafega dentro de um túnel IPv4.

Figura 111: Estratégia de Tunelamento



SAIBA MAIS

A compreensão aprofundada das novas características providas pelo IPv6 requer a leitura de uma publicação especializada. Para saber mais, consulte (BRITO, Samuel H. Bucke, 2013).

Introdução ao Roteamento Dinâmico

Conforme visto há pouco, um roteador encaminha datagramas com base na sua tabela de roteamento. Cada roteador possui uma tabela de roteamento minimalista, formada pelas redes às quais ele está diretamente conectado. Novas entradas podem ser adicionadas àquela tabela de duas formas distintas:

1. Manualmente, pelo administrador. Estas rotas são classificadas como estáticas, haja vista que apenas o administrador pode removê-las ou alterá-las;

2. Automaticamente, por um protocolo de roteamento. Estas rotas são classificadas como dinâmicas, haja vista que o protocolo pode removê-las ou alterá-las ao sabor das mudanças que ocorrem na topologia da interconexão.

Um protocolo de roteamento implementa um algoritmo de roteamento. Há duas classes de algoritmos de roteamento: vetor de distância (DV – *Distance Vector*) e estado de enlace (LS – *Link State*). Estas classes possuem características bem distintas. Roteadores cujos protocolos de roteamento implementam um algoritmo do tipo vetor de distância (cujo representante mais popular é o algoritmo de Bellman-Ford) não precisam conhecer a topologia completa da interconexão para calcular o melhor caminho até um destino; seu conhecimento topológico resume-se a seus vizinhos. Por outro lado, roteadores cujos protocolos de roteamento implementam um algoritmo do tipo estado de enlace (cujo representante mais popular é o algoritmo de Dijkstra) precisam conhecer a topologia completa da interconexão para calcular o melhor caminho até um destino.

Um conceito muito importante para a compreensão do roteamento dinâmico é o de sistema autônomo (AS – *Autonomous System*). O AS é uma interconexão cujos roteadores estão sob uma mesma administração. A internet é composta por ASes de diversos tamanhos – alguns com mais e outros com menos roteadores. Um protocolo de roteamento é classificado como IGP (*Interior Gateway Protocol*) ou EGP (*Exterior Gateway Protocol*), dependendo de onde ele é executado. Um protocolo classificado como IGP é executado dentro de um AS – exemplos deste tipo de protocolo são o RIP (*Routing Information Protocol*) e o OSPF (*Open Shortest Path First*). Por outro lado, um protocolo classificado como EGP é executado entre ASes – o protocolo EGP utilizado na prática é o BGP (*Border Gateway Protocol*). Todos os protocolos citados serão cobertos minimamente nas seções seguintes.

RIP

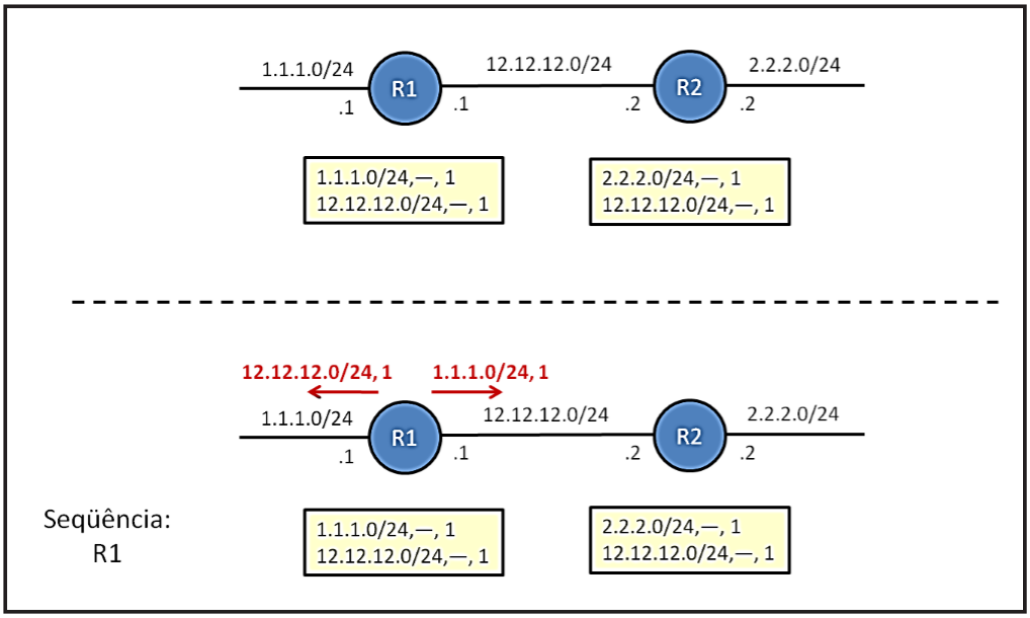
O RIP (*Routing Information Protocol*) é um dos protocolos de roteamento mais antigos. Apesar de ser um protocolo que intervém na tabela de rotas do roteador, adicionando entradas, o RIP é um protocolo do nível de aplicação. Ele é usuário do UDP – mais especificamente, processos RIP associam-se à porta 520 do UDP. A primeira versão do RIP, denominada RIPv1, é especificada pela RFC 1058. O RIPv1 é um protocolo classful, o que significa que ele pressupõe que a internet seja composta apenas por redes classe A, B e C; em outras palavras, redes cujas máscaras sejam /8, /16 e /24, apenas. O efeito desta consideração é que atualizações RIPv1 (mensagens que contêm rotas para redes) trazem consigo apenas os endereços das redes, porque as máscaras estariam implícitas na própria definição daqueles endereços. Sabe-se que o conceito de classes há muito deixou de ser aplicado. A segunda versão do RIP, especificada pela RFC 1723 e denominada RIPv2, reflete este fato ao definir que suas atualizações devem conter tanto os endereços das redes quanto as suas respectivas máscaras. O RIPv2, assim, é considerado um protocolo *classless*.

No RIP, o custo de cada enlace no RIP é igual a 1 (um). Portanto, o melhor caminho até um determinado destino é necessariamente o caminho mais curto em termos da quantidade de

saltos até se chegar lá. Uma rede diretamente conectada tem custo igual a 1 (um), e uma rede inatingível tem custo igual a 16. Este último custo foi definido para aliviar os efeitos de um fenômeno conhecido como contagem até o infinito, e cujo detalhamento está além da pretensão deste texto.

Um efeito da definição deste custo é o fato de o RIP ser utilizado apenas em ASes de pequeno porte – no qual uma rede está a no máximo 15 roteadores de distância de um remetente. A Figura 112 ilustra dois roteadores, R1 e R2, com suas respectivas tabelas de rotas minimalistas – contendo apenas as redes às quais eles estão diretamente conectados. Em seguida, R1 anuncia suas rotas. À esquerda, ele envia um anúncio para a rota 12.12.12.0/24 com custo igual a 1 (um) e à direita ele envia um anúncio para a rota 1.1.1.0/24 com o mesmo custo.

Figura 112: R1 envia seus anúncios

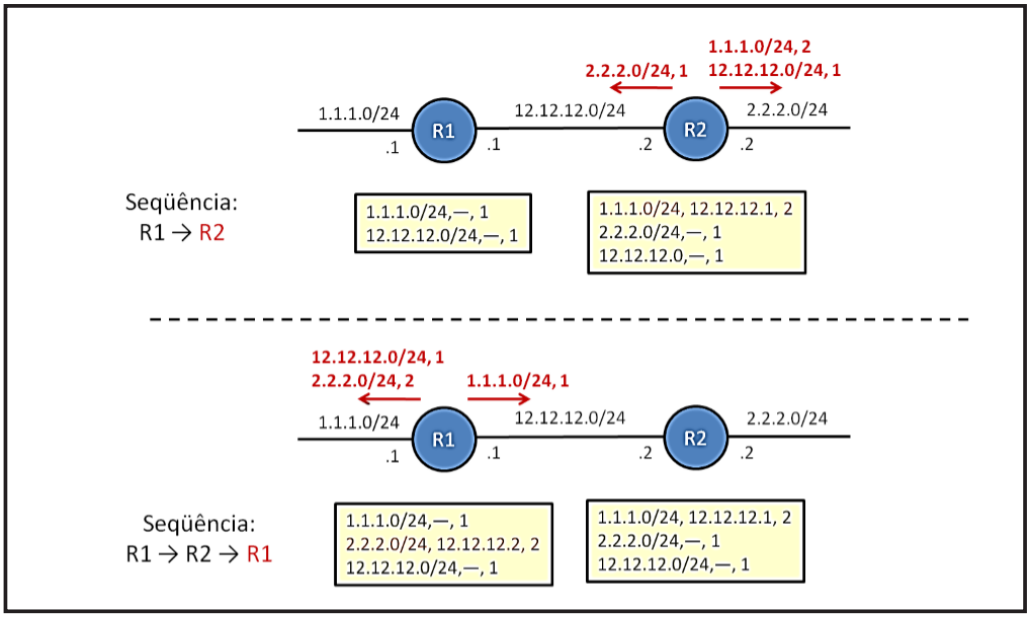


Na primeira metade da Figura 113, R2 recebe o anúncio de R1 e atualiza sua tabela de rotas adicionando uma entrada para 1.1.1.0/24 com custo igual a 2 (dois). Este custo é a soma do custo de R1 até 1.1.1.0/24 (1) com o custo de R1 a R2 (1). Nesta mesma metade, R2 divulga suas rotas através de suas interfaces. O anúncio inclui a rota recém-aprendida para 1.1.1.0/24, juntamente com seu respectivo custo (2).

Na segunda metade da Figura 113, R1 recebe o anúncio de R2 para a rede 2.2.2.0/24. Ele atualiza sua tabela de rotas adicionando uma entrada para aquela rede com custo igual a 2 (dois). Este custo é a soma do custo de R2 até 2.2.2.0/24 (1) com o custo de R2 a R1 (1). Nesta mesma metade, R1 divulga suas rotas através de suas interfaces. O anúncio inclui a rota recém-aprendida para 2.2.2.0/24, juntamente com seu respectivo custo (2).

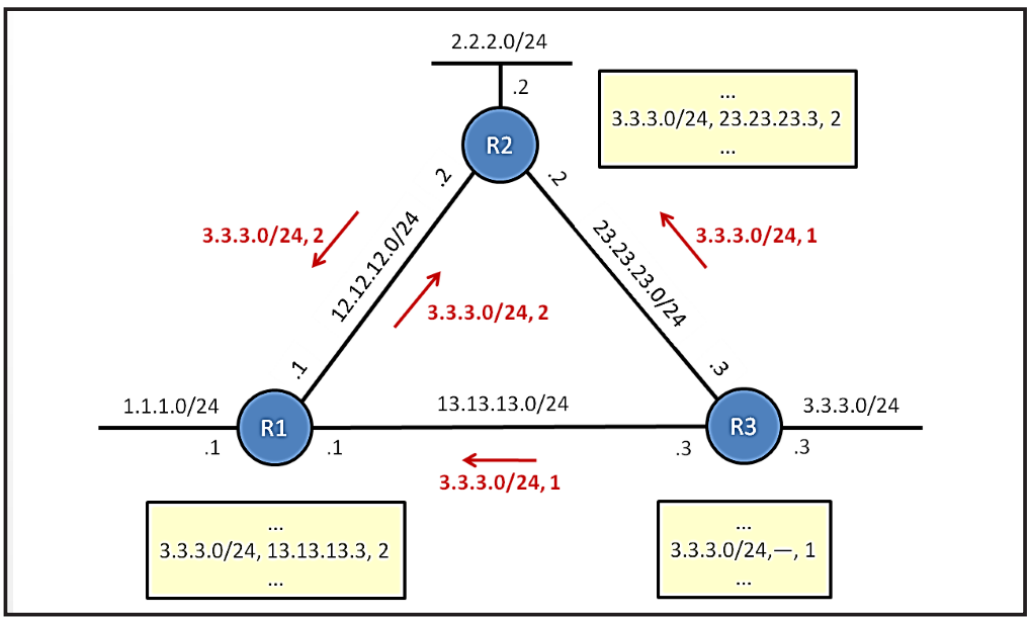
Conforme comentado há pouco, o RIP implementa o algoritmo de Bellman-Ford, classificado como vetor de distâncias. Esta denominação pode ser mais bem compreendida a partir da Figura 112 e da Figura 113, que ilustram que um anúncio RIP é composto por um conjunto (vetor) de rotas para redes contendo seus respectivos custos (distâncias até elas). A forma como as rotas são propagadas ao longo da interconexão é por vezes denominada roteamento por rumor.

Figura 113: R2 recebe o anúncio de R1, o processa e envia seus anúncios



A interconexão ilustrada pelas figuras anteriores é tal que cada roteador possui apenas um caminho para atingir cada uma das redes. Isto não ocorre na Figura 114, que ilustra uma interconexão na qual o roteador R3 transmite anúncios para R2 e R1 referentes à rede diretamente conectada 3.3.3.0/24. Tanto R1 quando R2 transmitem, um a outro, anúncios referentes àquela rede. Portanto, tanto R1 quanto R2 possuem duas rotas para 3.3.3.0/24. Ambos, porém, escolhem a rota com o menor custo – isto é, escolhem R3 como próximo salto.

Figura 114: R1 e R2 escolhem a rota para 3.3.3.0/24, que possui o menor custo



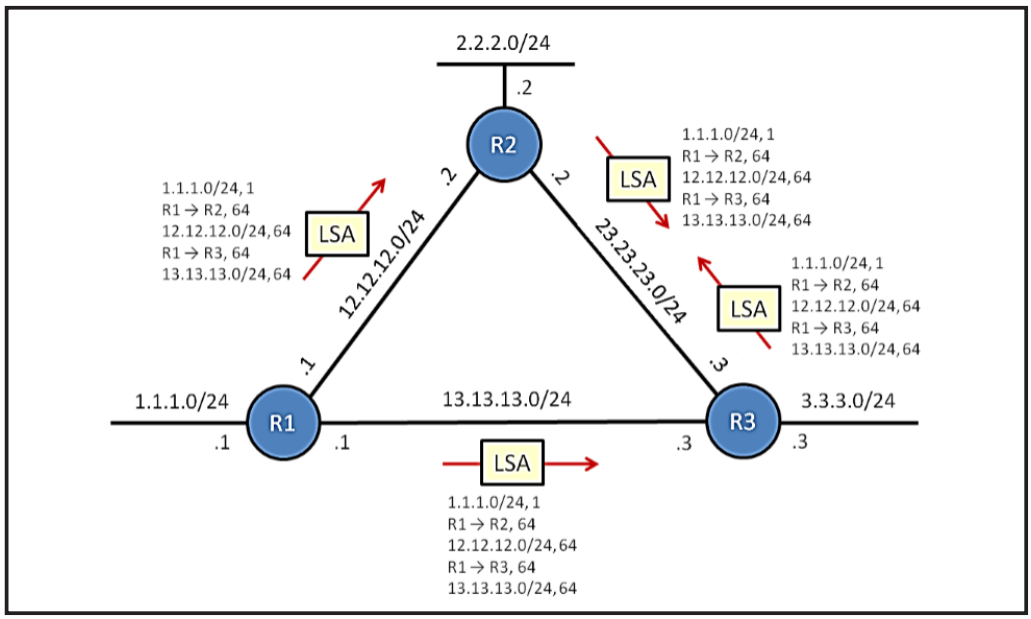
FTP

O OSPF (*Open Shortest Path First*) é o IGP destinado ao uso em ASes maiores. Sua versão atual (OSPFv2) é especificada pela RFC 2328. Mensagens OSPF são encapsuladas diretamente em datagramas IP, cujo valor do campo de cabeçalho "Protocolo de Nível Superior" é igual a 89. Conforme mencionado anteriormente, o OSPF implementa o algoritmo de Dijkstra, cuja classificação como estado de enlaces implica o conhecimento da topologia completa da interconexão para o cálculo do melhor caminho para cada destino. Cada roteador OSPF possui este conhecimento a partir de LSAs (*Link State Advertisements*) anunciados pelos demais roteadores da interconexão. Estes LSAs são armazenados na LSDB (*Link State Data Base*) de cada roteador.

No OSPF, uma adjacência é constituída entre dois roteadores vizinhos quando ambos conhecem um a outro. Este processo inclui a transmissão periódica de mensagens Hello. Uma vez estabelecida uma adjacência entre dois roteadores OSPF, eles sincronizam suas respectivas LSDBs um com o outro. Os LSAs recebidos por um roteador através de uma adjacência são transmitidos pelas demais adjacências das quais ele participa. Este processo, denominado *flooding*, faz com que cada LSA chegue a todos os roteadores do AS. A Figura 115 ilustra a transmissão de um LSA por parte de R1 através das suas respectivas adjacências com R2 e R3, bem como a propagação daquele mesmo LSA entre estes dois últimos roteadores.

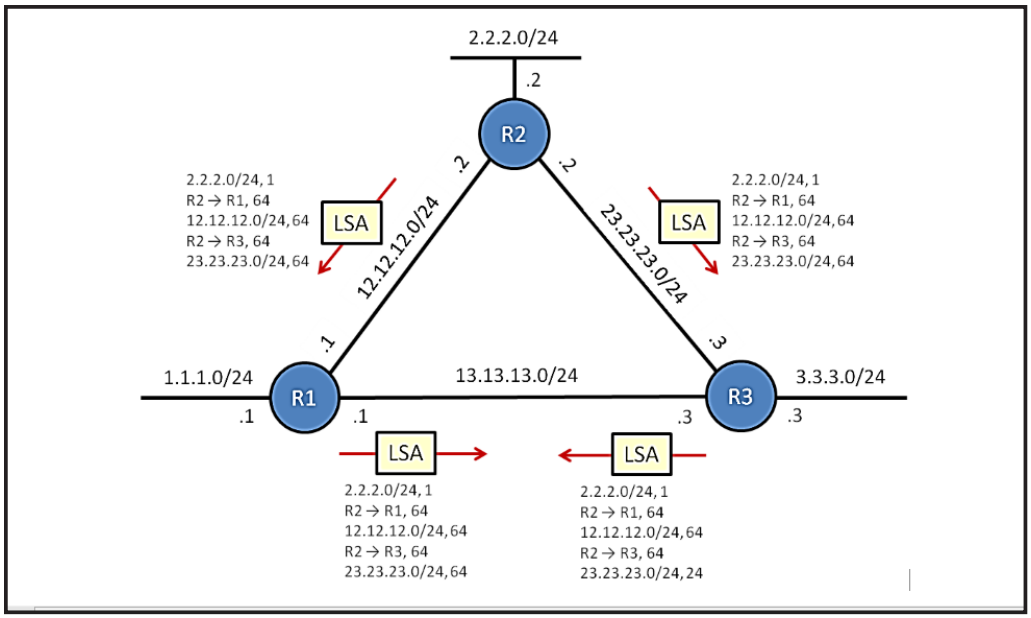
Este LSA é denominado LSA do tipo 1, ou LSA de Roteador. O LSA ilustrado pela figura descreve os enlaces de R1 ao informar que ele possui a rede diretamente conectada 1.1.1.0/24; que se conecta a R2 através da rede 12.12.12.0/24; e que se conecta a R3 através da rede 13.13.13.0/24. O valor do custo de um enlace não é especificado na RFC 2328. Em roteadores Cisco, por exemplo, o custo de um enlace é igual a 108 dividido pela largura de banda. Assim, enlaces seriais T1 (cuja largura de banda é de 1,544 Mbps) possuem custo igual a 64. Por outro lado, segmentos *Fast Ethernet* (cuja largura de banda é de 100 Mbps) possuem custo igual a 1 (um).

Figura 115: LSA enviado por R1 a R2 e R3, e sua consequente propagação por eles.



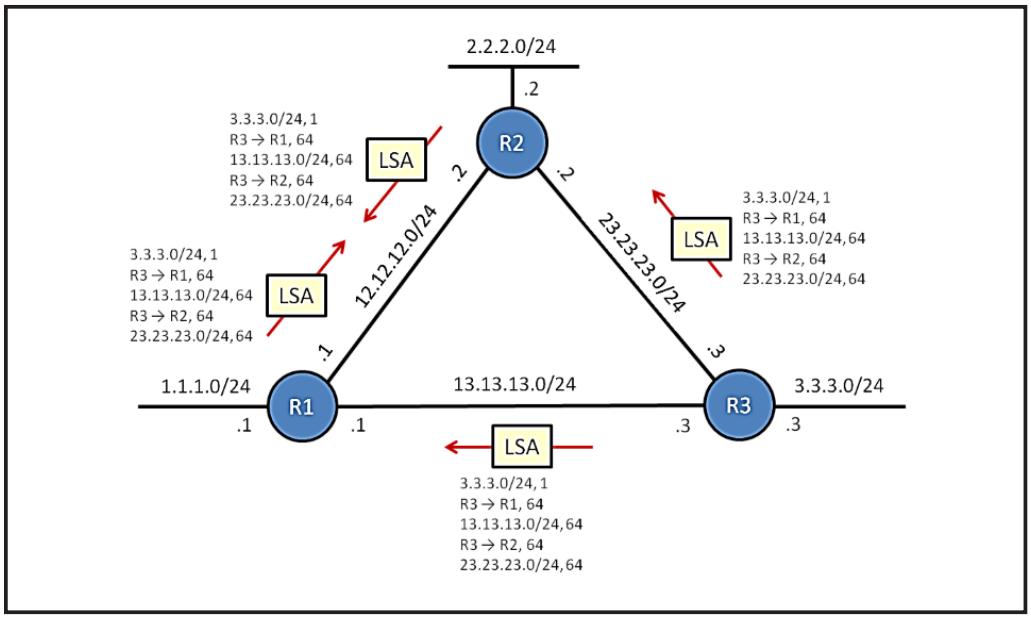
A Figura 116 ilustra a transmissão de um LSA de roteador por parte de R2 através das suas respectivas adjacências com R1 e R3, além da propagação daquele mesmo LSA por parte daqueles dois últimos roteadores. O LSA descreve todos os enlaces de R2 ao informar que ele possui a rede diretamente conectada 2.2.2.0/24, com custo igual a 1; que se conecta a R1 através da rede 12.12.12.0/24, com custo igual a 64; e que se conecta a R3 através da rede 23.23.23.0/24, com custo igual a 64.

Figura 116: LSA enviado por R2 a R1 e R3, e sua consequente propagação por eles.



A Figura 117 ilustra a transmissão de um LSA de Roteador por parte de R3 através das suas respectivas adjacências com R1 e R2, e a propagação daquele mesmo LSA por parte daqueles dois últimos roteadores. O LSA descreve todos os enlaces de R3 ao informar que ele possui a rede diretamente conectada 3.3.3.0/24, com custo igual a 1; que se conecta a R1 através da rede 13.13.13.0/24, com custo igual a 64; e que se conecta a R2 através da rede 23.23.23.0/24, com custo igual a 64.

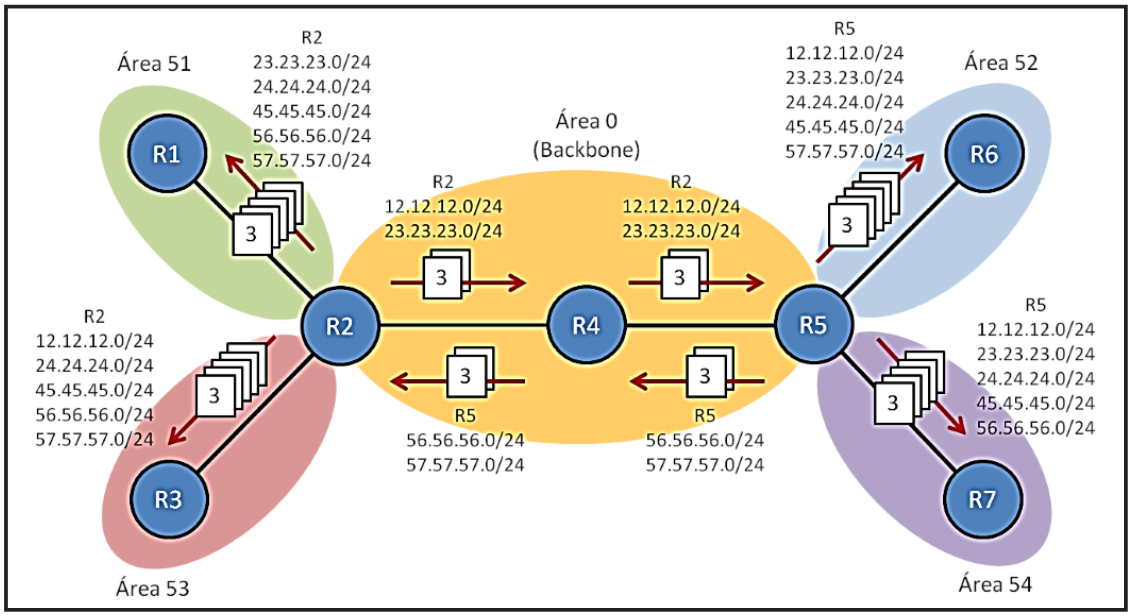
Figura 117: LSA enviado por R3 a R1 e R2, e sua consequente propagação por eles.



Conforme mencionado anteriormente, o OSPF é um IGP projetado para utilização em ASes de maior porte. Porém, quanto maior for a dimensão do AS, maior será a LSDB de cada roteador, exigindo maior capacidade de armazenamento. A duração da execução do algoritmo de Dijkstra também será naturalmente mais lenta em ASes de maior porte. Por estes motivos, um AS OSPF de maior porte é projetado para ser dividido em áreas designadas numericamente. Uma delas deve, obrigatoriamente, ser definida como área de *backbone* (área 0) e as outras devem ser diretamente conectadas a ela.

A Figura 118 ilustra um AS no qual as áreas 12, 23, 56 e 57 estão diretamente conectadas a área de *backbone* (área 0). O AS em questão não é, a rigor, complexo a ponto de ter que ser dividido em cinco áreas – a escolha da divisão é, puramente, didática. A divisão de um AS em áreas faz com que um ou mais dos seus roteadores sejam classificados como ABRs (*Area Border Routers* – Roteadores de Borda de Área). Um ABR é um roteador conectado a mais de uma área, sendo uma delas impreterivelmente a área de *backbone*. Na Figura 118, os roteadores R2 e R5 são ABRs.

Figura 118: Um AS OSPF dividido em várias áreas, e a transmissão de LSAs do tipo 3



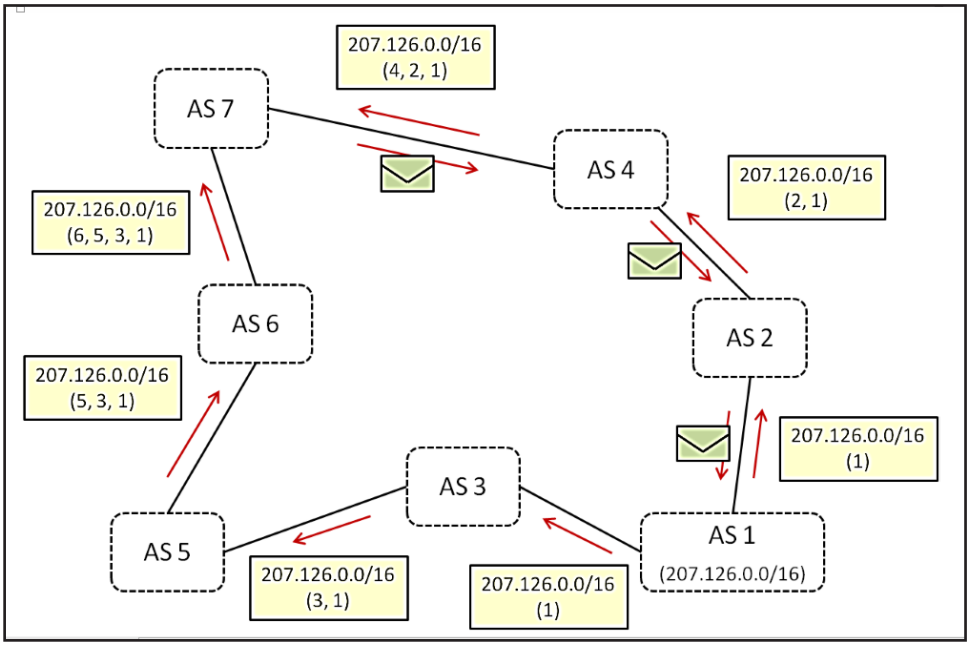
O ABR é responsável pela transmissão de LSAs específicos, denominados LSAs do tipo 3, ou LSAs de Resumo de Rede. Ele transmite para a área de backbone LSAs do tipo 3 que descrevem as redes localizadas em outras áreas. Na Figura 118, por exemplo, R2 transmite LSAs do tipo 3 para a área de *backbone* para informar acerca da existência das redes 12.12.12.0/24 e 23.23.23.0/24, localizadas nas áreas 12 e 23, respectivamente. Similarmente, R5 transmite LSAs do tipo 3 para a área de *backbone* para informar acerca da existência das redes 56.56.56.0/24 e 57.57.57.0/24, localizadas nas área 56 e 57, respectivamente. Por outro lado, um ABR transmite para uma área não-*backbone* LSAs do tipo 3 que descrevem as redes localizadas na área de *backbone* e nas demais áreas.

É o que fazem R2 e R5 na Figura 118, com relação às suas áreas não-*backbone*. O objetivo do LSA do tipo 3 é simples: fazer com que os roteadores de uma área passem a conhecer as redes localizadas nas demais áreas, porém sem necessitar conhecer exatamente onde elas estão localizadas naquelas áreas; aliás, esta é a razão pela qual o LSA do tipo 3 é denominado de Resumo de Rede. Há vários outros tipos de LSA além do tipo 1 e do tipo 3. De fato, o OSPF é complexo o suficiente para lidar com ASes de maior porte, e é um dos IGPs mais populares utilizados.

BGP

O RIP e o OSPF, convém lembrar, são exemplos de IGPs. O BGP (*Border Gateway Protocol*), por outro lado, é o EGP utilizado na prática na internet. Sua versão atual, denominada BGP-4, é especificada pela RFC 4721. O BGP implementa um algoritmo classificado como vetor de caminhos. A Figura 119 ilustra um conjunto de sete ASes, cada um deles identificado com seu respectivo ASN (*AS Number*). Os ASNs são fornecidos pela IANA, organização responsável pela atribuição de endereços IP. Os ASNs ilustrados pela Figura 119 são de caráter meramente didático.

Figura 119: O BGP implementa um algoritmo classificado como vetor de caminhos



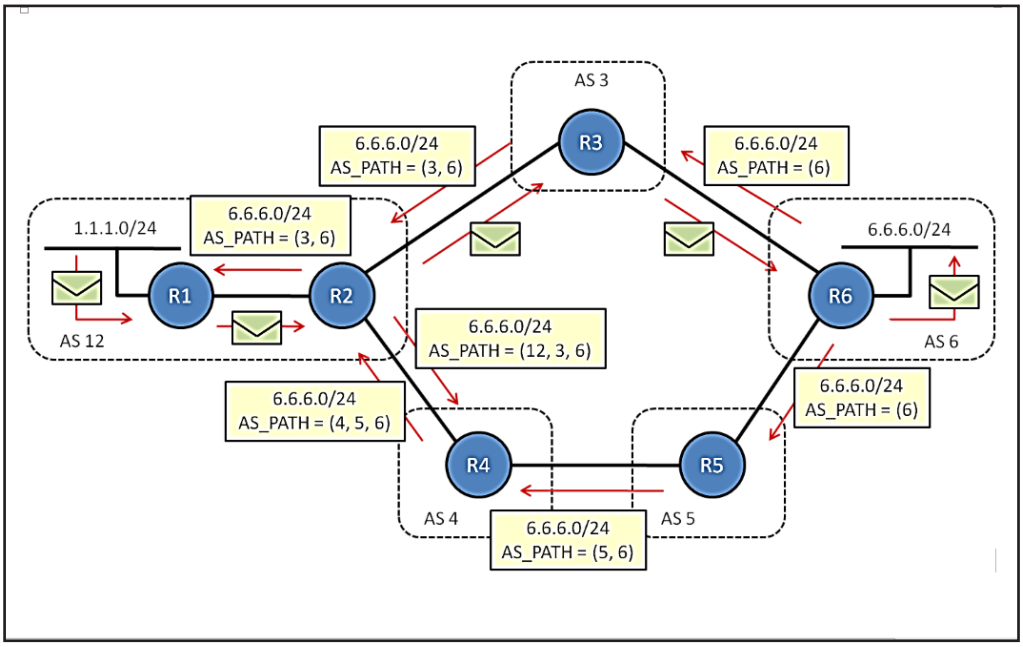
Na Figura 119, o AS 1 anuncia a sua rede interna 207.126.0.0/16 para os seus ASes vizinhos – 2 e 3. Um anúncio BGP contém parâmetros denominados atributos de caminho, dentre os quais um dos mais importantes é o AS_PATH. O AS_PATH é simplesmente uma lista dos ASes percorridos pelo anúncio. Assim, ao sair do AS 1, o AS_PATH do anúncio é 1 (um). Os ASes 2 e 3 adicionam seus próprios ASNs ao anúncio quando o propagam aos seus vizinhos – os ASes 4 e 5, respectivamente. Quando o AS 7 recebe os anúncios relativos a 207.126.0.0/16 dos ASes 4 e 6, seus roteadores de borda escolhem o anúncio contendo o menor AS_PATH.

Portanto, o caminho escolhido pelos roteadores de borda do AS 7 para escoar datagramas destinados a 207.126.0.0/16 será encaminhá-lo para o AS 4, haja vista que eles percorrerão uma quantidade menor de ASes até chegar ao seu destino do que se eles fossem encaminhados para o AS 6. Naturalmente, a quantidade de roteadores percorrida nos ASes 2 e 4 pode ser maior do que aquela que seria percorrida nos ASes 3, 5 e 6. Os enlaces presentes no caminho escolhido podem possuir menor largura de banda do que aqueles presentes no caminho alternativo.

Para contemplar os contextos supracitados, o processo decisório do BGP inclui a análise de outros atributos de caminho. A Figura 120 ilustra uma interconexão entre cinco ASes, propositalmente simplificados em suas respectivas estruturas. O roteador R6 anuncia para seus pares externos (R3 e R5) a rede 6.6.6.0/24. Percebe-se que o caminho escolhido para aquela rede, a partir do AS 12, passa pelo AS 3 pelo fato de ele transmitir ao AS 12 uma rota para 6.6.6.0/24 com menor AS_PATH. Porém, se os enlaces do caminho alternativo (AS 12 – AS 4 – AS 5 – AS 6) tiverem uma largura de

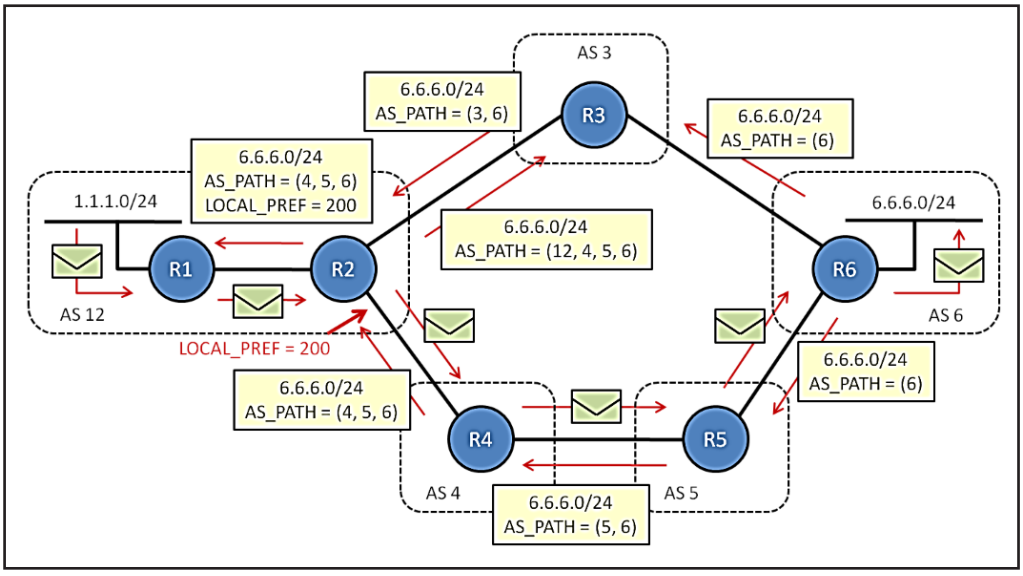
banda maior que os enlaces do caminho escolhido (AS 12 – AS 3 – AS 6), haveria possibilidade de aquela escolha ser revertida?

Figura 120: Rota escolhida para 6.6.6.0/24 é a que possui o menor AS_PATH



Há um atributo de caminho, denominado LOCAL_PREFERENCE, que possui precedência sobre o AS_PATH na escolha do melhor caminho. Seu objetivo é influenciar o caminho percorrido por um tráfego de saída em um AS. Na Figura 121, o valor do LOCAL_PREFERENCE é configurado em 200 no roteador R2 na rota recebida a partir do AS 4. Esta rota é agora a rota preferida para rotear datagramas destinados a 6.6.6.0/24, a despeito de possuir um AS_PATH maior.

Figura 121: Rota escolhida para 6.6.6.0/24 é a que possui o maior LOCAL_PREFERENCE

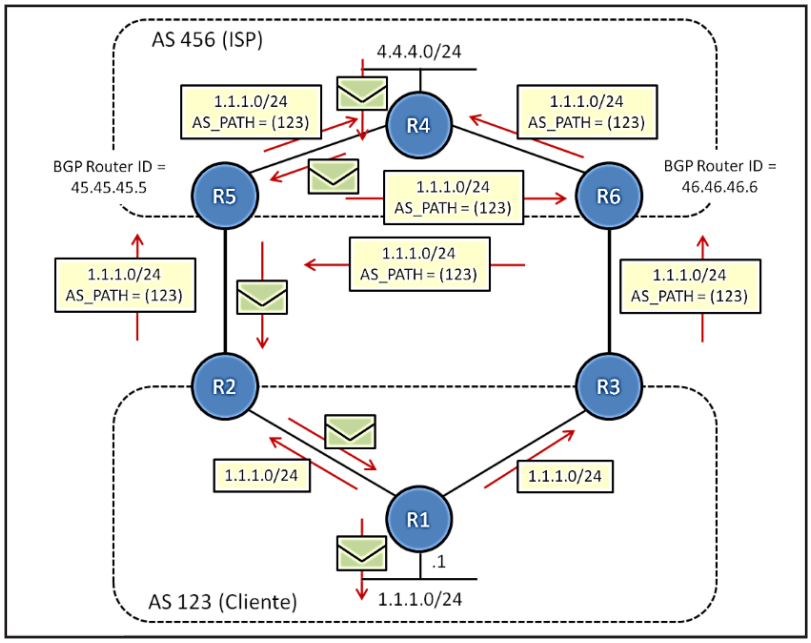


Decisões aparentemente pouco ortodoxas como a demonstrada há pouco são exemplos da aplicação de políticas de roteamento, e isto é algo bem comum no BGP; sua flexibilidade, aliás, serve a este propósito. O processo decisório do BGP contempla uma série de passos. Na Figura 122, por exemplo, o caminho percorrido por datagramas endereçados a 1.1.1.0/24 a partir do roteador R4 é definido pela comparação dos valores dos identificadores BGP de seus pares – R5 e R6. Este passo

é dado em virtudes das rotas enviadas por R5 e R6 com relação a 1.1.1.0/24 possuírem o mesmo AS_PATH, 123.

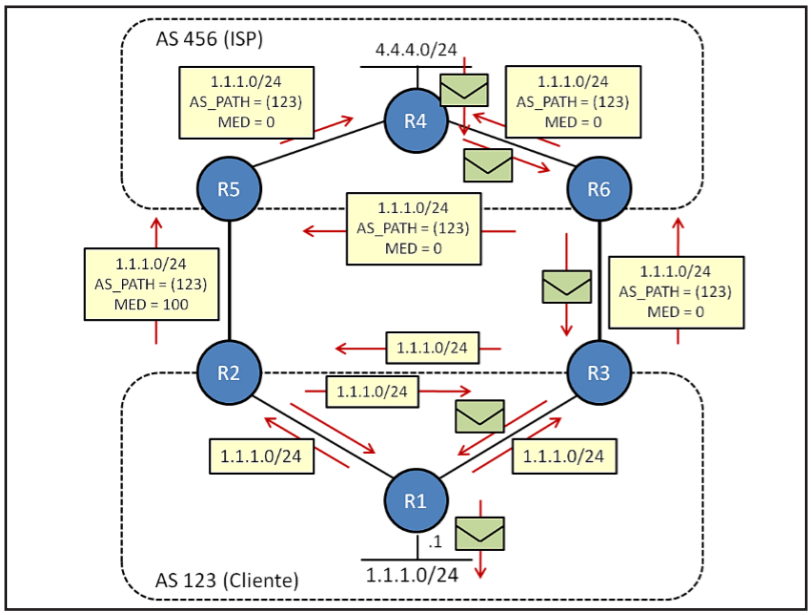
De acordo com o processo decisório do BGP, o roteador que possuir o menor identificador BGP é aquele que detém a rota escolhida. Isto, em última análise, faz com que o tráfego destinado a 1.1.10/24 ingresse no AS 123 a partir do enlace R2 – R5. Haveria possibilidade de reverter tal decisão, fazendo com que aquele tráfego ingresse pelo enlace R3 – R6?

Figura 122: Rota é escolhida em função do menor identificador BGP



Há um atributo de caminho, denominado MULTI_EXIT_DISCRIMINATOR (comumente abreviado por MED), que se presta justamente a influenciar o tráfego de ingresso em um AS. Na Figura 123, os valores do MED em ambos os enlaces supracitados são ajustados de tal sorte que o MED presente no anúncio enviado por R3 é menor que o MED presente no anúncio enviado por R2. O processo decisório do BGP estabelece que, havendo empate com relação ao tamanho do AS_PATH, os valores do MED das rotas devem ser comparados antes do exame dos identificadores BGP. Desta forma, o enlace R3 – R6 é agora o caminho escolhido.

Figura 123: Rota é escolhida em função do menor valor do MED



O BGP possui muitos outros atributos de caminho, além do AS_PATH, LOCAL_PREFERENCE e MED. O conjunto destes atributos proporciona ao protocolo uma flexibilidade enorme, capaz de tornar realidade as mais complexas políticas de roteamento. Naturalmente, esta capacidade se dá à custa de uma maior complexidade, evidenciada por um processo decisório composto por mais de uma dezena de passos.

O roteamento dinâmico é um tópico muito complexo. Para compreensão adequada, um curso inteiro deve ser dedicado ao assunto. Aqui, o objetivo foi esclarecer os pontos nevrálgicos acerca do que é um protocolo de roteamento e quais as características principais dos maiores representantes desta classe tão importante de protocolos.

SAIBA MAIS

A compreensão aprofundada dos protocolos de roteamento dinâmico requer a leitura de publicações especializadas. Consulte Doyle (1998) para saber mais sobre o RIP e o OSPF, e (Doyle; Carrol (2001) para saber mais sobre o BGP.

III

OS NÍVEIS DE ENLACE E FÍSICO

Dando prosseguimento à nossa abordagem *top-down*, esta unidade é dedicada aos níveis de enlace e físico. Inicialmente, porém, ela descreve alguns protocolos importantíssimos para o funcionamento prático das redes: trata-se dos protocolos ICMP, ARP e DHCP. Em seguida, as redes locais *Ethernet* são abordadas e, com elas, as estratégias para restrição dos domínios de colisão (VLANs) e prevenção de tempestades de *broadcast* infinitas (STP). As redes locais *WiFi* são então apresentadas e, por fim, são fornecidas algumas noções acerca dos padrões de cabeamento estruturado.

Objetivos:

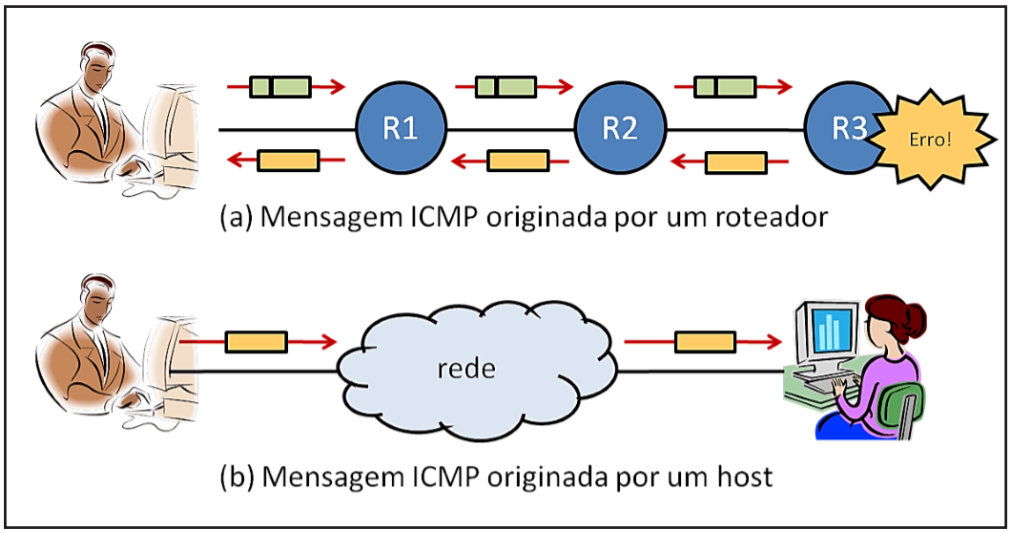
- Fornecer ao aluno uma compreensão holística acerca da configuração de rede de sistemas finais e de roteadores;
- Descrever como funcionam as tecnologias mais populares utilizadas em redes locais – *Ethernet* e *WiFi*;
- Prover noções de cabeamento estruturado.

ICMP

O ICMP (*Internet Control Message Protocol*) é especificado pela RFC 792. Trata-se de um protocolo utilizado para detecção de problemas na rede, tais como a ausência de rotas para um determinado destino ou a formação de laços de roteamento, por exemplo. O ICMP é utilizado nas ferramentas de diagnóstico *ping* e *traceroute*.

Geralmente, uma mensagem ICMP é criada por um roteador e tem como destino o emissor original do datagrama que lhe deu origem, conforme ilustra a Figura 124. Porém, conforme ilustrado pela mesma figura, mensagens ICMP também podem ser originadas por *hosts*. Estas mensagens têm como destino outros *hosts* ou roteadores.

Figura 124: Mensagens ICMP enviadas por um roteador e por um host



A Figura 125 esclarece que uma mensagem ICMP é encapsulada diretamente em um datagrama IP. Ela também ilustra que a mensagem ICMP contém um cabeçalho-base que especifica qual é o seu tipo. De fato, há vários tipos de mensagem ICMP, mas alguns deles caíram em desuso. A Figura 126 ilustra os tipos mais populares de mensagens ICMP.

Figura 125: Mensagens ICMP são encapsuladas diretamente em datagramas IP

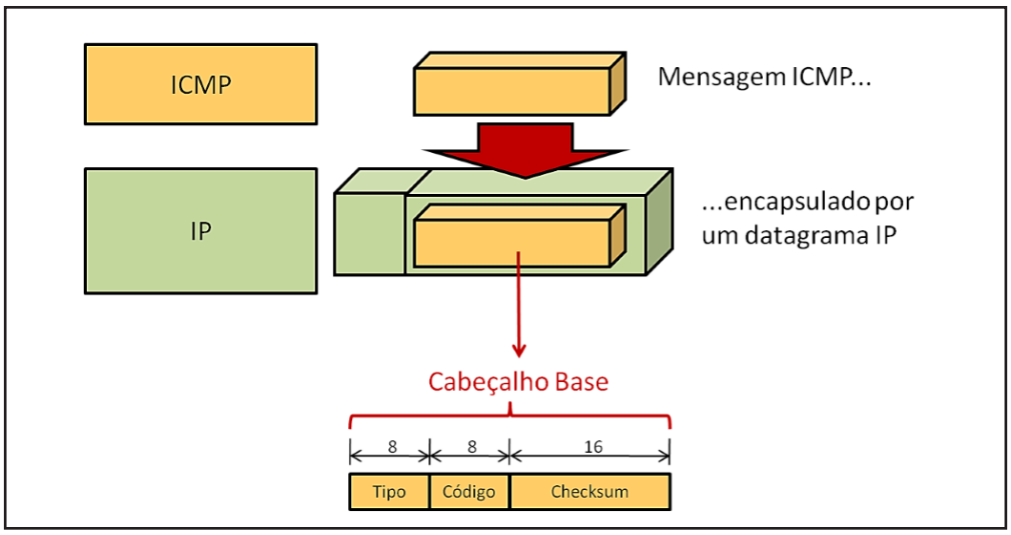
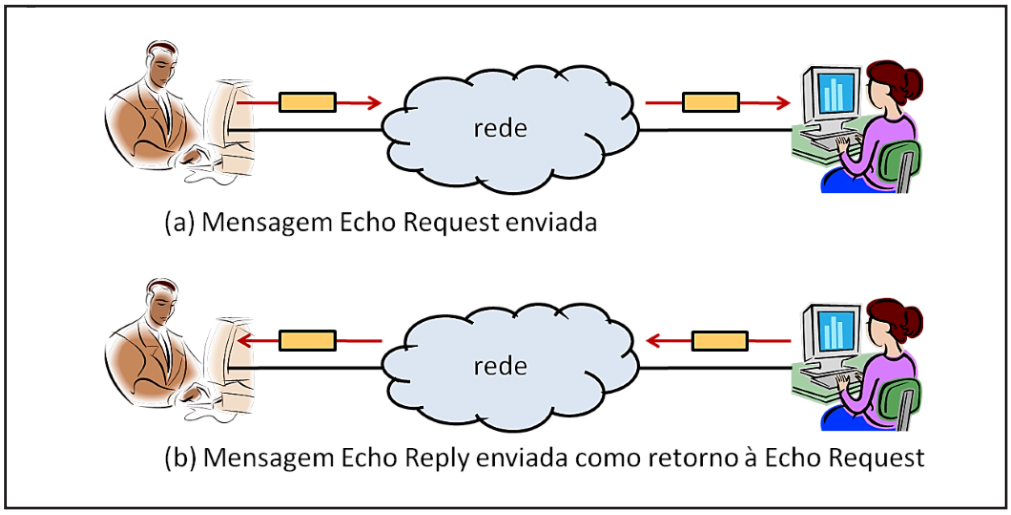


Figura 126: Tipos populares de mensagens ICMP

Tipo	Mensagem
0	*Echo Reply*
3	*Destination Unreachable*
8	*Echo Request*
11	*Time Exceeded for a Datagram*

As mensagens *Echo Request e Echo Reply* são utilizadas pela ferramenta *ping*, tipicamente executada em um host, para verificar se um determinado destino está atingível – quer seja ele outro *host*, quer seja um roteador. Esta ferramenta baseia-se no princípio de que qualquer *driver* ICMP que recebe uma mensagem *Echo Request* deve retornar uma mensagem *Echo Reply*. A Figura 127 ilustra a troca destas mensagens.

Figura 127: Troca de mensagens Echo Request e Echo Reply



O *ping* foi criado por Mike Muuss, e seu nome foi inspirado no som emitido por um sonar na detecção de um obstáculo por meio do eco. A Figura 128 ilustra a execução desta ferramenta, que possui versões para diferentes sistemas operacionais.

Figura 128: Exemplo de execução do ping

```
C:\Users\Helcio>ping www.ufersa.edu.br

Disparando agroweb.ufersa.edu.br [200.137.6.8] com 32 bytes
   de dados:
Resposta de 200.137.6.8: bytes=32 tempo=137ms TTL=115
Resposta de 200.137.6.8: bytes=32 tempo=151ms TTL=115
Resposta de 200.137.6.8: bytes=32 tempo=150ms TTL=115
Resposta de 200.137.6.8: bytes=32 tempo=134ms TTL=115

Estatísticas do Ping para 200.137.6.8:
    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 0 (0%
       de perda),
Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 134ms, Máximo = 151ms, Média = 143ms
```

A mensagem Destination Unreachable é utilizada para informar que o destinatário de um datagrama está inatingível. Ela é endereçada ao remetente daquele datagrama. Dentre as possíveis razões para que seu destinatário não esteja atingível estão:

- O datagrama necessitava ser fragmentado, mas sua *flag* DF encontrava-se ativada e ele foi descartado;
- O datagrama chegou a um roteador que não possuía uma rota para ele, descartando-o;
- O destinatário não existe, estava desconectado ou desligado;
- O datagrama encapsulava um segmento em cuja porta de destino não havia nenhuma aplicação no destinatário;
- O datagrama encapsulava uma PDU desconhecida pelo *driver* IP do destinatário.

A mensagem *Time Exceeded* pode ser originada tanto por um roteador quanto por um *host*, quando ele for o destinatário final de um datagrama. Ela é originada por um roteador quando ele decrementa o TTL de um datagrama e este chega a 0 (zero), o que o força a descartá-lo. A mensagem é endereçada ao remetente original do datagrama. Quando originada por um *host*, a mensagem *Time Exceeded* indica que expirou o temporizador associado à recepção de fragmentos de um datagrama. Mais uma vez, a mensagem é endereçada ao remetente original do datagrama.

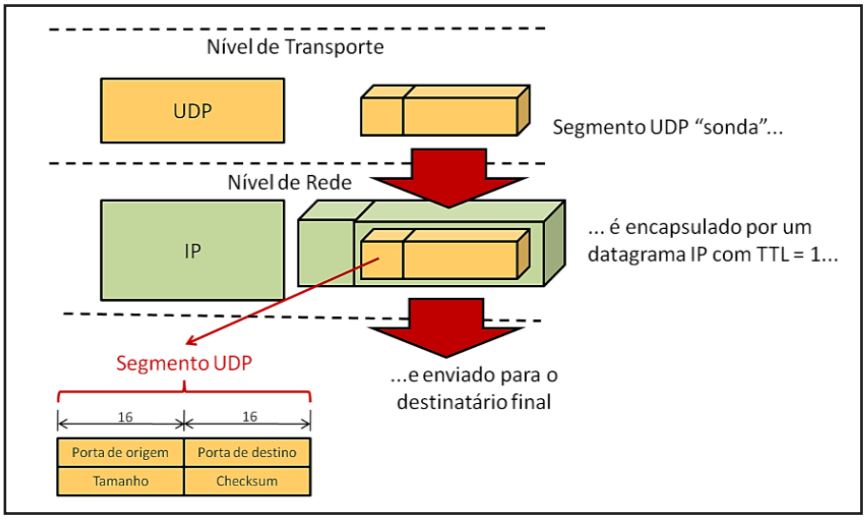
Além do *ping*, outra ferramenta de diagnóstico muito popular é o *tracert* – em máquinas Linux, seu análogo chama-se *traceroute*. O *tracert* exibe a rota seguida até um determinado destinatário listando de roteadores que um datagrama percorreria até aquele destinatário. A Figura 129 ilustra um exemplo de execução desta ferramenta.

Figura 129: Exemplo de execução do *tracert*

```
C:\Users\Helcio>tracert www.ufersa.edu.br
Rastreando a rota para agroweb.ufersa.edu.br [200.137.6.8]
com no máximo 30 saltos:
  1    <1 ms    <1 ms    <1 ms  192.168.1.1
  2     *        *        *     Esgotado o tempo limite do pedido.
  3    11 ms    11 ms    12 ms  172.16.1.1
  4    59 ms    59 ms    61 ms  189-51-34-205.unotel.com.br [189.51.34.205]
  5    67 ms    58 ms    59 ms  189-51-32-3.unotel.com.br [189.51.32.3]
  6    58 ms    78 ms    58 ms  as1916.sp.ptt.br [187.16.216.4]
  7    66 ms    70 ms    68 ms  xe-3-1-0-3000-r0-mg.bkb.rnp.br [200.143.252.74]
  8   116 ms   117 ms   121 ms  xe-3-1-1-3000-r0-ce.bkb.rnp.br [200.143.252.142]
  9   127 ms   135 ms   126 ms  xe-3-1-0-3000-r0-rn.bkb.rnp.br [200.143.252.137]
 10   128 ms   127 ms   133 ms  2910-r3-rn.bkb.rnp.br [200.143.255.202]
 11   125 ms   124 ms   126 ms  rt1.pop-rn.rnp.br [200.137.0.66]
 12   140 ms   130 ms   130 ms  192.168.11.2
 13   138 ms   137 ms   135 ms  agroweb.ufersa.edu.br [200.137.6.8]
 14   131 ms   132 ms   131 ms  agroweb.ufersa.edu.br [200.137.6.8]
Rastreamento concluído.
```

O *tracert* utiliza o UDP para enviar um segmento sonda, caracterizado por possuir uma porta de destino aleatória e ter um *payload* contendo dados sem significado prático. Este segmento sonda é encapsulado inicialmente por um datagrama cujo TTL é definido em 1 (um). Este processo é ilustrado pela Figura 130.

Figura 130: Segmento sonda construído pelo *tracert*

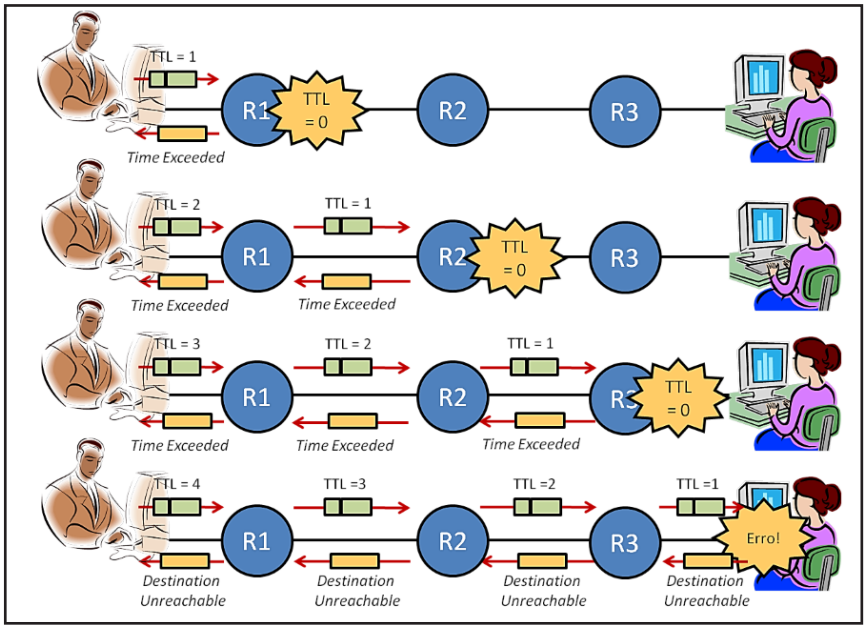


Na Figura 131, o segmento sonda é enviado pelo *host* e imediatamente descartado pelo primeiro roteador no caminho até o destinatário (R1) pela expiração do TTL. Uma mensagem *Time Exceeded* é enviada para o host. Ela é encapsulada por um datagrama cujo endereço IP de origem é o endereço de R1. Este endereço é então impresso na tela do usuário, e um novo datagrama encapsulando o segmento sonda é enviado, desta vez com o TTL igual a 2 (dois).

Este datagrama é descartado agora por R2, que envia uma nova mensagem Time Exceeded para o host e tem seu endereço impresso na tela. Um novo datagrama encapsulando o segmento sonda é enviado com o TTL igual a 3 (três), o que faz com que ele seja descartado agora por R3. Este roteador envia uma nova mensagem Time Exceeded para o host e tem seu endereço impresso na tela.

O novo datagrama com TTL igual a 4 (quatro) atravessa todos os três roteadores. Porém, não há nenhum processo de aplicação associado à porta de destino do segmento sonda. Por isto, uma mensagem *Destination Unreachable* é enviada para o remetente original do datagrama, que exibe o endereço do destinatário e conclui a execução do *tracert*.

Figura 131: Funcionamento do *tracert*



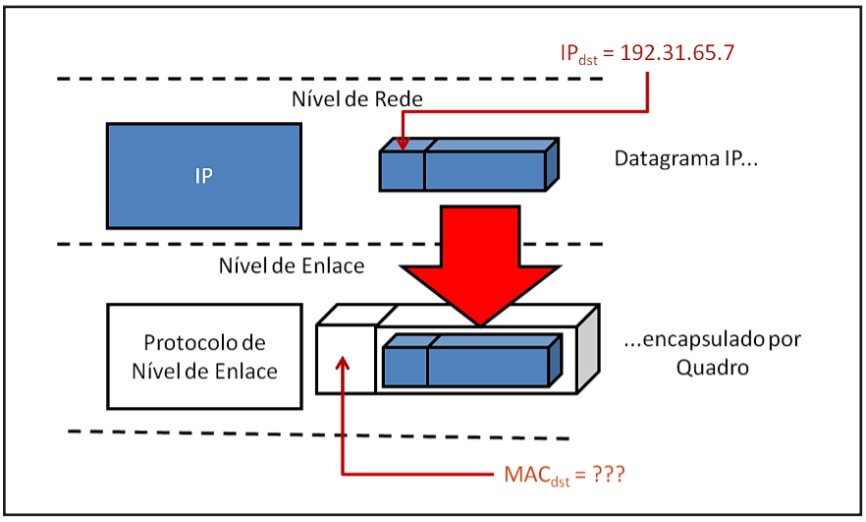
FTP

O uso de endereços IP é condição necessária à comunicação de rede. Porém, ela não é por si só suficiente. O fato é que *hosts* comunicam-se entre si trocando quadros – denominação dada à PDU do nível de enlace. Portanto, quando um *host* deseja se comunicar com outro, ele deve descobrir:

1. O endereço IP do outro *host*, utilizando o protocolo DNS;
2. O endereço MAC de destino, utilizando o ARP (*Address Resolution Protocol*).

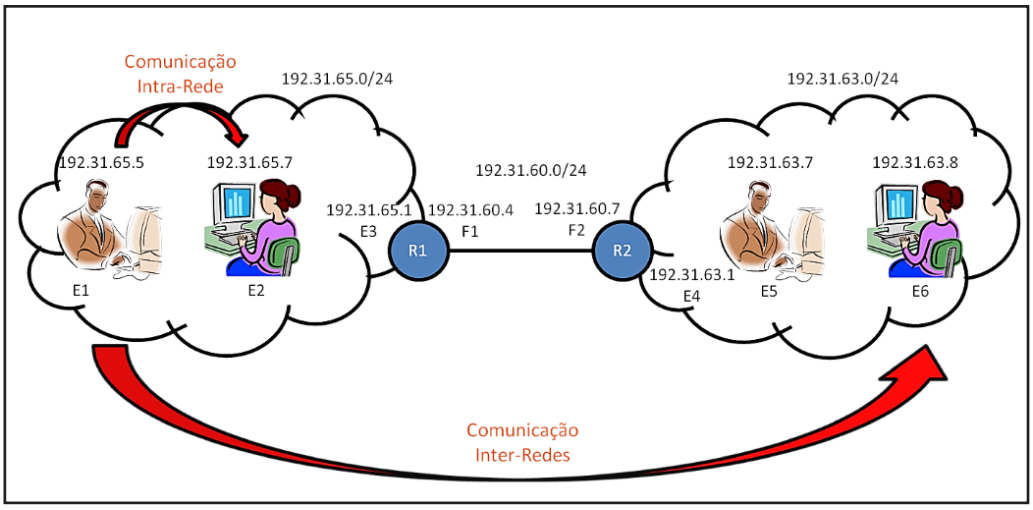
A Figura 132 ilustra o segundo requisito: um datagrama endereçado ao host 192.31.65.7 é encapsulado por um quadro cujo endereço MAC de destino é, *a priori,* desconhecido. Não há, portanto, como transmitir aquele datagrama sem esta informação.

Figura 132: A necessidade de conhecimento do endereço MAC de destino



A Figura 133 servirá como base para explicar de quem será o endereço MAC de destino a ser descoberto utilizando o ARP. Ela ilustra três redes interconectadas pelos roteadores R1 e R2. São consideradas duas comunicações a partir do *host* 192.31.65.5: uma delas com o *host* 192.31.65.7, na mesma rede (comunicação intra-rede); e outra com o *host* 192.31.63.8, em outra rede (comunicação inter-redes). Os endereços MAC ilustrados são os mais genéricos possíveis, haja vista que o funcionamento do ARP independe do tipo de protocolo de enlace utilizado.

Figura 133: Comunicação intra-rede e inter-redes

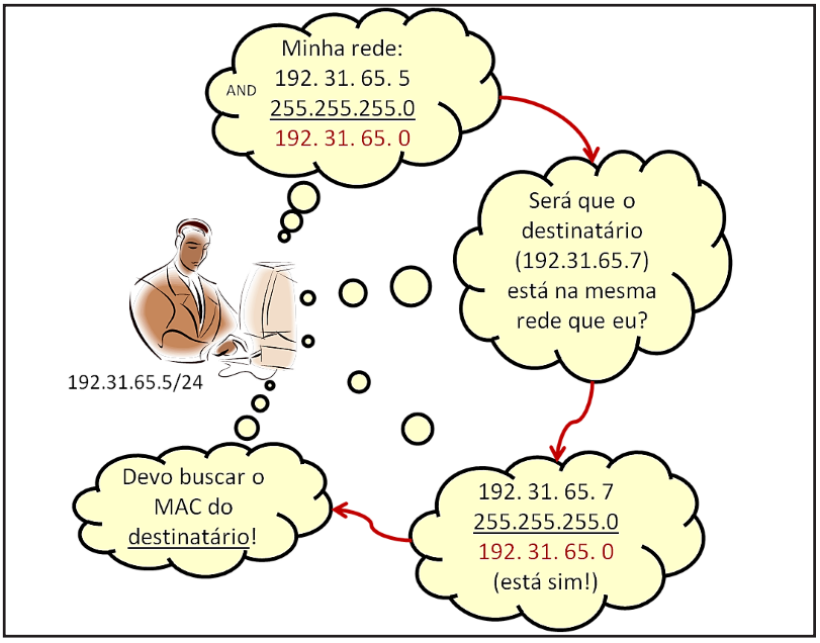


Para saber de quem será o endereço MAC a ser descoberto, um *host* deve descobrir se o destinatário pertence à mesma rede que ele: se o destinatário pertencer à mesma rede que ele, o endereço MAC a ser descoberto é o endereço MAC do destinatário; por outro lado, se o destinatário pertencer à outra rede, o endereço MAC a ser descoberto é o endereço MAC do gateway.

Naturalmente, o procedimento de descoberta citado no parágrafo anterior requer que um *host* saiba antecipadamente qual é a rede a que ele próprio pertence. O *host* obtém esta informação simplesmente pela computação do AND bit-a-bit entre o seu endereço IP e a sua máscara de rede. Para saber se o destinatário está na mesma rede que ele, o *host* computa o AND bit-a-bit entre o endereço IP do destinatário e a sua própria máscara de rede – convém esclarecer que não há como o host saber qual é a máscara de rede do destinatário.

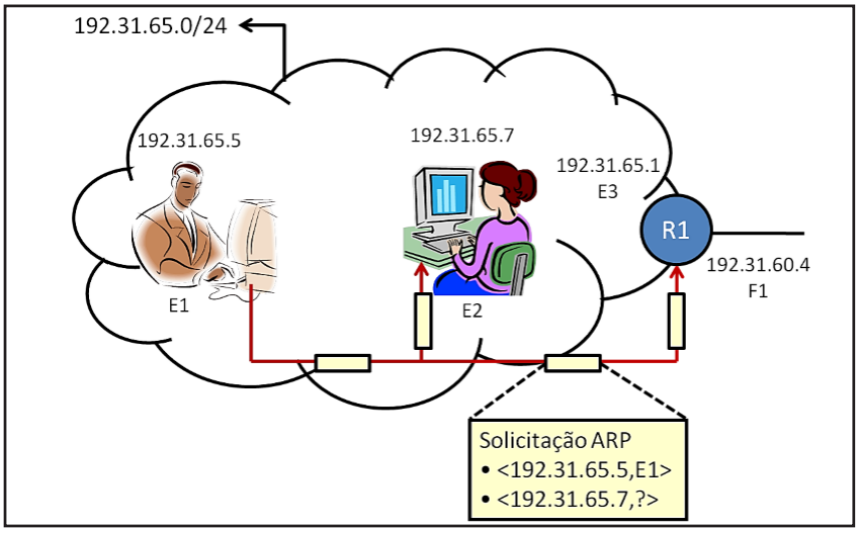
A Figura 134 ilustra o processo de descoberta realizado pelo *host* cujo endereço e máscara são 192.31.65.5 e 255.255.255.0 quando ele deseja enviar um datagrama para 192.31.65.7. O *host* descobre inicialmente que sua rede é a 192.31.65.0. Em seguida, ele descobre que o destinatário também está naquela rede. Portanto, o endereço MAC a ser descoberto é o endereço MAC do destinatário.

Figura 134: Remetente conclui que destinatário se encontra na mesma rede



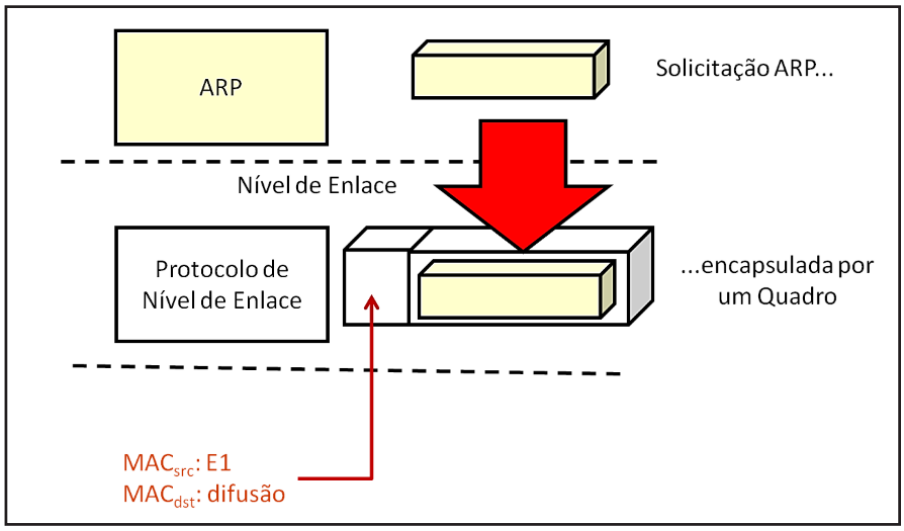
O ARP é um protocolo simples, baseado no modelo solicitação-resposta. Na Figura 135, uma solicitação ARP é difundida na rede do remetente. Ela apresenta o solicitante, exibindo seu endereço IP (192.31.65.5) e seu endereço MAC (E1), e busca descobrir o endereço MAC associado ao destinatário (192.31.65.7).

Figura 135: Envio de solicitação ARP para descobrir endereço MAC do destinatário



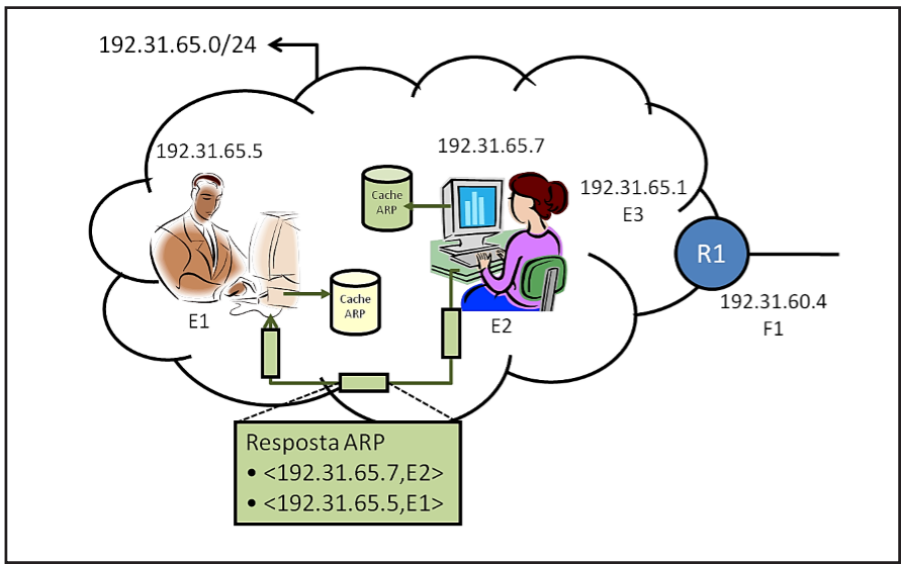
A Figura 136 ilustra a localização do ARP na pilha de protocolos. Conforme exibido, a solicitação ARP é encapsulada diretamente em um quadro cujo endereço MAC de destino corresponde a um endereço de difusão.

Figura 136: Encapsulamento de uma solicitação ARP em um quadro *broadcast*



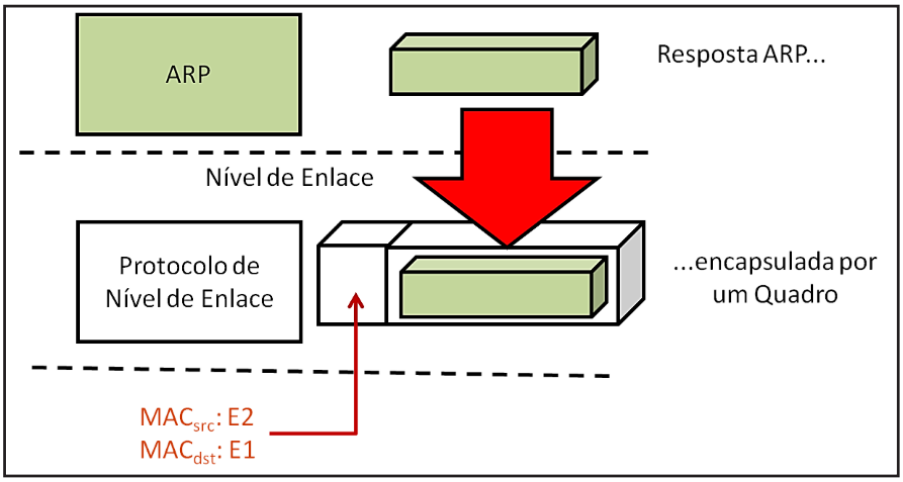
A Figura 137 ilustra o envio da resposta ARP pelo detentor oficial da informação – isto é, o destinatário do datagrama a ser transmitido. A resposta ARP fornece o endereço MAC do destinatário (E2) e ecoa os dados contidos na solicitação. Ambos os hosts envolvidos armazenam em caches os endereços MAC apresentados, de forma a não mais precisarem enviar solicitações em busca deles.

Figura 137: Envio da resposta ARP



Diferentemente do que ocorre com a solicitação, a resposta ARP é encapsulada em um quadro unicast, conforme ilustrado pela Figura 138.

Figura 138: Encapsulamento de uma resposta ARP em um quadro *unicast*



A Figura 139 ilustra o datagrama endereçado a 192.31.65.7 sendo encapsulado pelo quadro, já com a informação de que seu endereço MAC de destino é E2. A Figura 140 ilustra a transmissão do datagrama.

Figura 139: Datagrama encapsulado por um quadro endereçado ao destinatário final

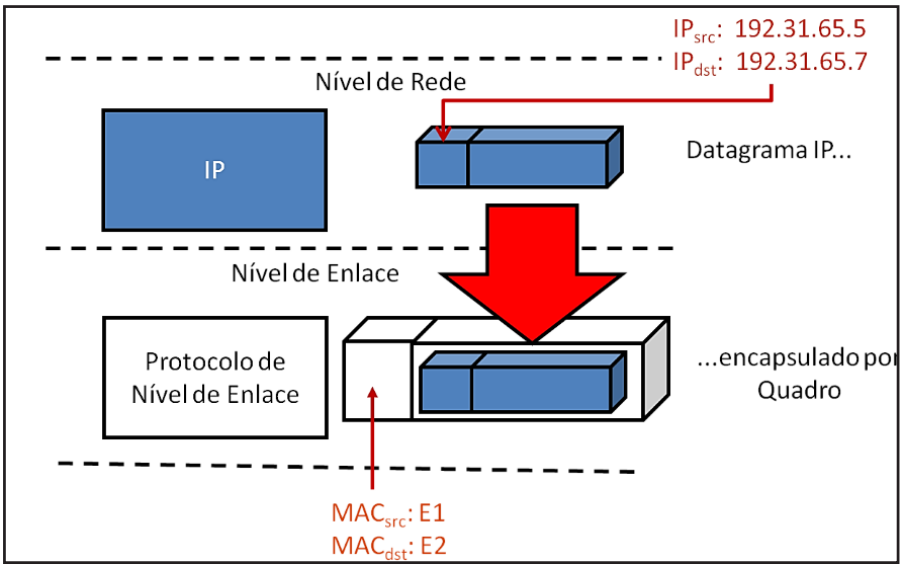
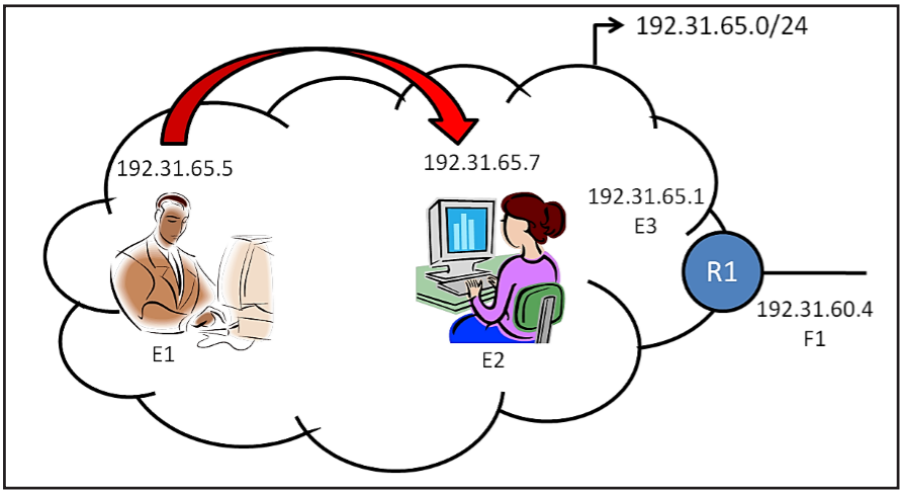
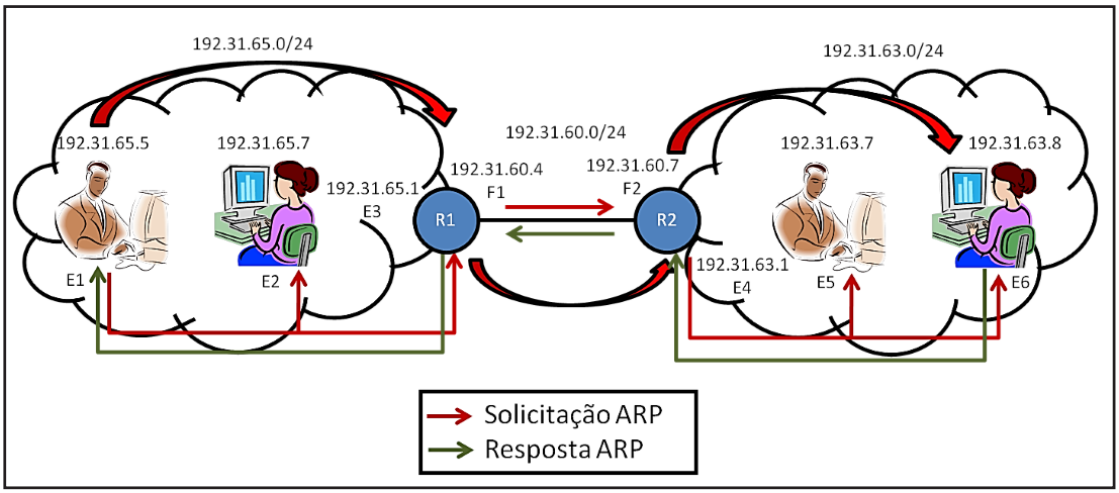


Figura 140: Envio do datagrama para o destinatário final



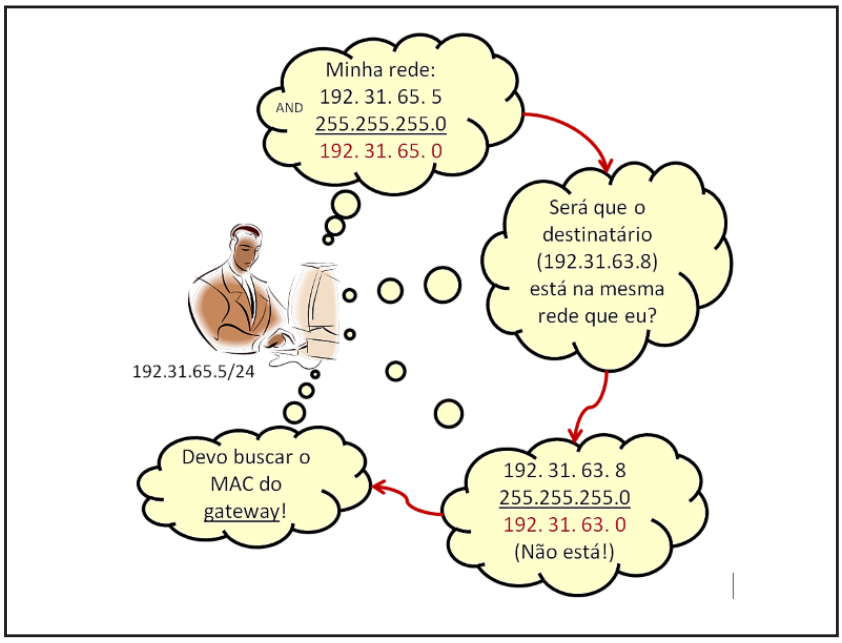
A Figura 141 ilustra as solicitações e respostas ARP envolvidas quando o *host* 192.31.65.5 deseja enviar um datagrama para o *host* 192.31.63.8. Trata-se de uma comunicação inter-redes, porque o destinatário está em uma rede diferente daquela do remetente.

Figura 141: Solicitações e respostas ARP em uma comunicação inter-redes



Mais uma vez o remetente deve descobrir se o destinatário se encontra na mesma rede em que ele. Este processo é ilustrado pela Figura 142. Com efeito, o processo de descoberta se um destinatário está ou não na mesma rede que o remetente explica o motivo de um host ser configurado com a máscara de rede. Ao verificar que o destinatário se encontra em outra rede, o remetente conclui que o endereço MAC a ser descoberto é o endereço do *gateway.*

Figura 142: Remetente conclui que destinatário se encontra em uma rede diferente



A Figura 143 ilustra o envio de uma solicitação ARP que busca descobrir qual é o endereço MAC do gateway da rede 192.31.65.0/24 – ou seja, 192.131.65.1. Ela também ilustra o envio da resposta associada à solicitação. Pode-se explicar neste contexto porque um *host* é configurado com o endereço IP do *gateway*: simplesmente para que o ARP seja utilizado para se descobrir qual é o endereço MAC associado àquele endereço IP. O endereço MAC do *gateway* é necessário

à geração do quadro que transportará o datagrama até ele, onde será roteado, conforme a Figura 144 ilustra.

Figura 143: Troca de mensagens ARP para descobrir endereço MAC do *gateway*

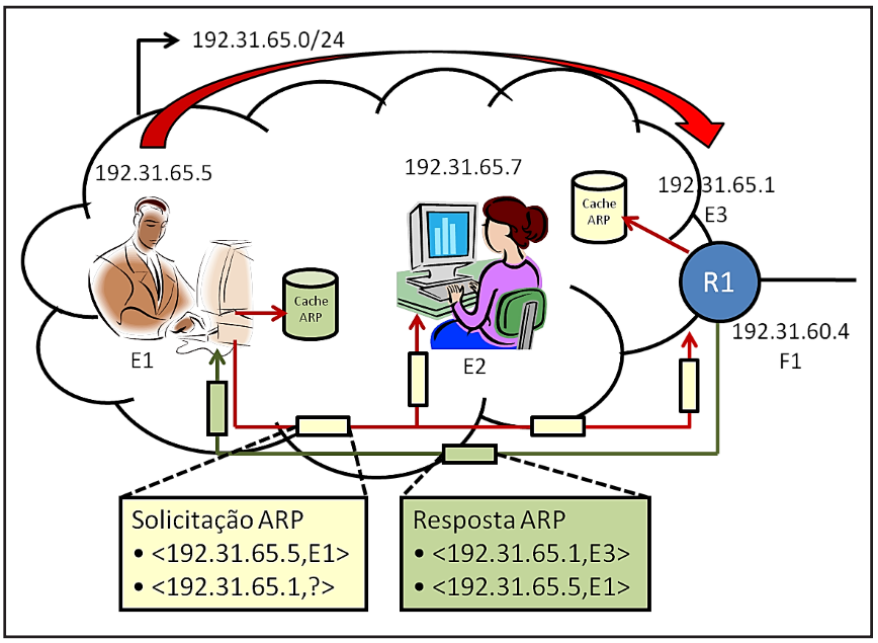
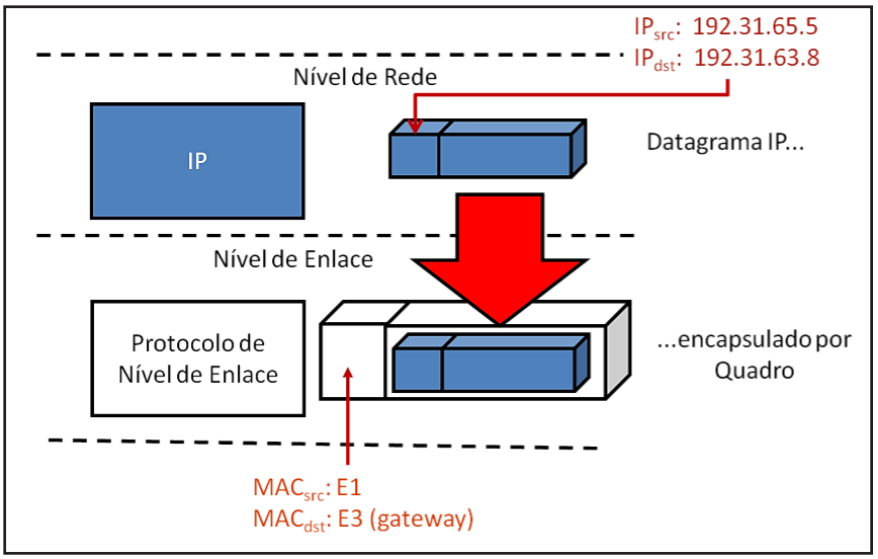
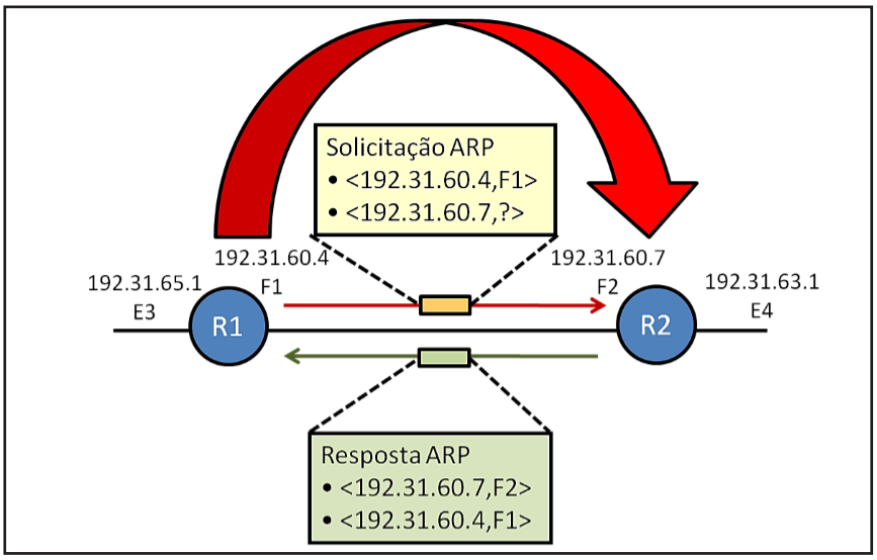


Figura 143: Datagrama encapsulado por um quadro endereçado ao *gateway*



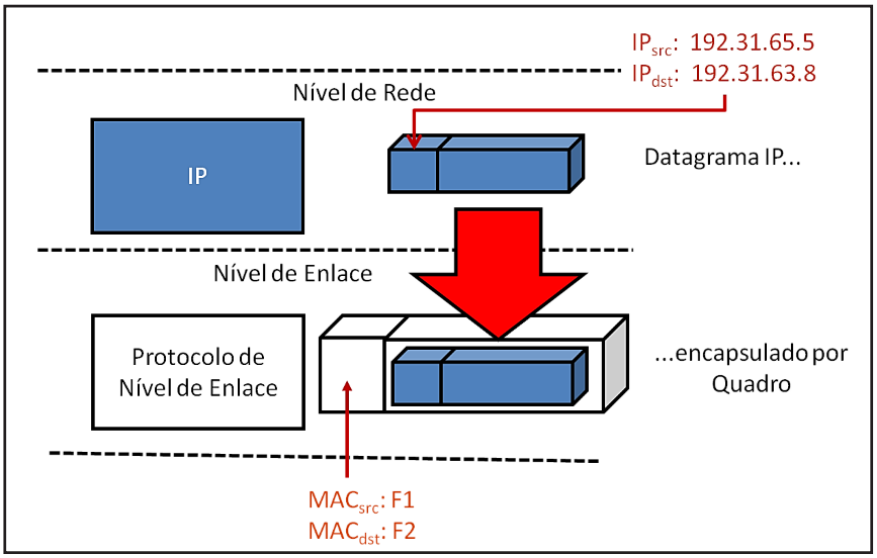
Ao realizar o roteamento do datagrama, R1 conclui que o próximo salto é 192.31.60.7 – endereço IP de R2 na tabela de rotas de R1. Isto implica o encapsulamento do datagrama em um quadro endereçado a R2. Porém, qual é o endereço MAC que corresponde a 192.31.60.7? Conforme ilustra a Figura 145, uma solicitação ARP é enviada por R1 e uma resposta é recebida, informando que o endereço MAC associado a 192.31.60.7 é F2.

Figura 145: Troca de mensagens ARP para descobrir endereço MAC do próximo salto



Neste contexto, explica-se por que existe a informação de endereço IP do próximo salto na tabela de rotas de um roteador: ela existe para que este endereço sirva como subsídio para o envio de uma solicitação ARP que busca descobrir o endereço MAC associado a ele. A Figura 146 ilustra o datagrama encapsulado em um quadro endereçado a R2.

Figura 146: Datagrama encapsulado por um quadro endereçado a R_2



Quando R2 roteia o datagrama, ele descobre que o destinatário encontra-se na rede 192.31.63.0/24. Por se encontrar diretamente conectado àquela rede, sabe que tem que encapsular o datagrama em um quadro endereçado diretamente a 192.31.63.7. Portanto, ele difunde uma solicitação ARP buscando descobrir qual é o endereço MAC associado àquele endereço IP. A resposta ARP informa que o endereço MAC associado a 192.31.63.7 é E6. A Figura 147 ilustra a difusão da solicitação e o envio da resposta ARP. Neste contexto, na ausência de cache em R2 e caso o destinatário esteja desligado, desconectado ou simplesmente não exista, aquele roteador cria uma mensagem ICMP *Destination Unreachable* endereçada e a encapsula em um datagrama endereçado ao remetente

do datagrama original, 192.31.65.5. A Figura 148 ilustra o datagrama encapsulado em um quadro endereçado ao destinatário final.

Figura 147: Troca de mensagens ARP para descobrir endereço MAC do destinatário final

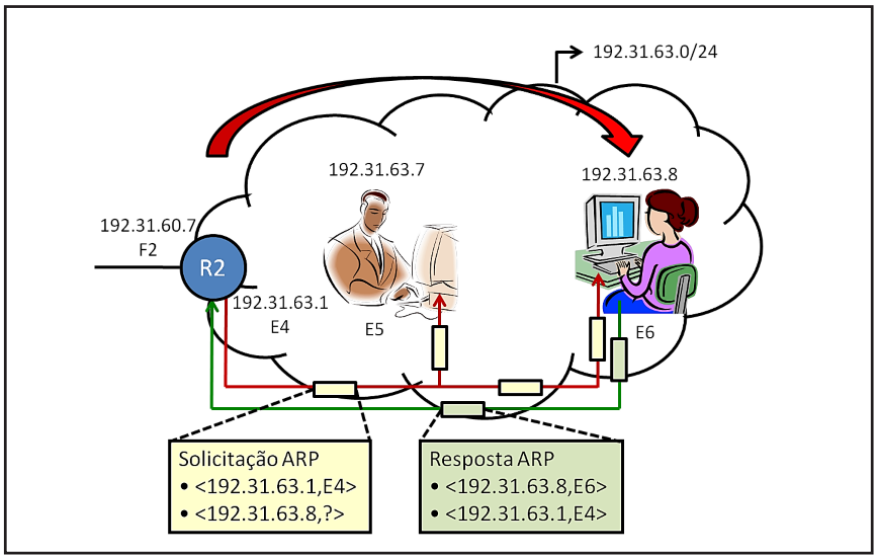
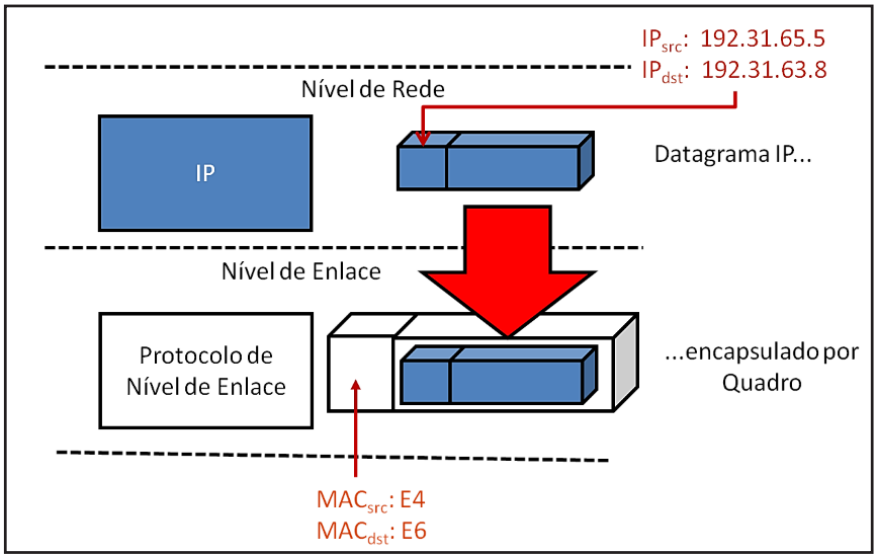


Figura 148: Datagrama encapsulado por um quadro endereçado ao destinatário final



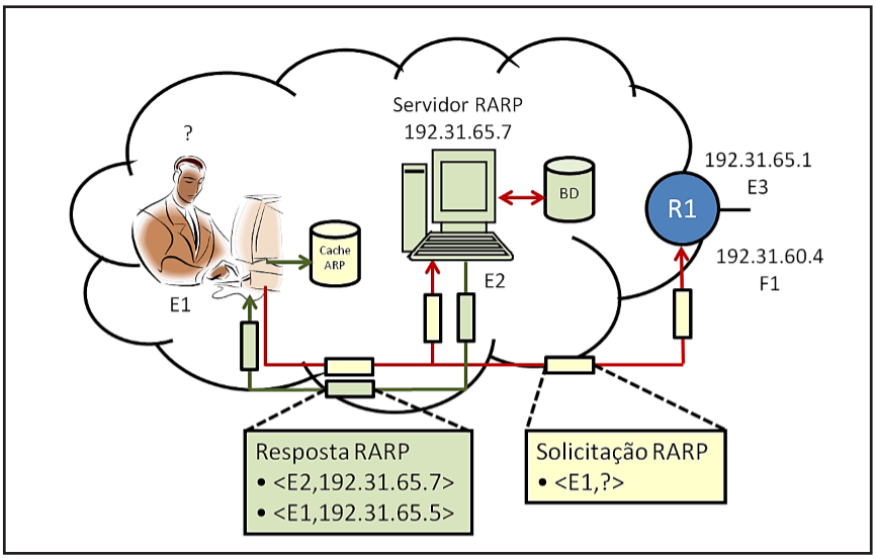
O RARP (Reverse ARP) é um protocolo projetado para ser utilizado na obtenção dinâmica de endereço IP. Um *host*, ao ser inicializado, comunicar-se-ia com um servidor RARP para obter seu endereço IP dinamicamente. Porém, o RARP já nasceu com uma grave limitação: apenas o endereço IP seria fornecido; assim, a máscara de rede e os endereços de gateway e dos servidores DNS não seriam fornecidos. A gravidade desta limitação vem da importância destes outros parâmetros.

Outra grave limitação do RARP era que, uma vez que um servidor RARP atribuía um endereço IP a um *host*, este endereço permaneceria indefinidamente associado àquele *host*. Mesmo que o *host* estivesse desligado ou não mais fizesse parte da rede a qual pertencesse o endereço atribuído, este endereço não poderia ser atribuído a mais ninguém. Poder-se-ia chegar a uma situação na qual não houvesse mais endereços a ser atribuídos, apesar de boa parte dos endereços atribuídos estar associada a *hosts* que não mais existem naquela rede.

Finalmente, endereços só seriam fornecidos por um servidor RARP caso o *host* solicitante estivesse na mesma rede local que ele. Em outras palavras, o raio de alcance do RARP era limitado à rede local do servidor. Por todas estas limitações, o RARP não é utilizado na prática – diferentemente do ARP. Em particular, a atribuição de endereços IP (e outros parâmetros associados) é realizada na prática por outro protocolo, a ser abordado posteriormente. Assim, o estudo do RARP é breve e possui caráter meramente histórico.

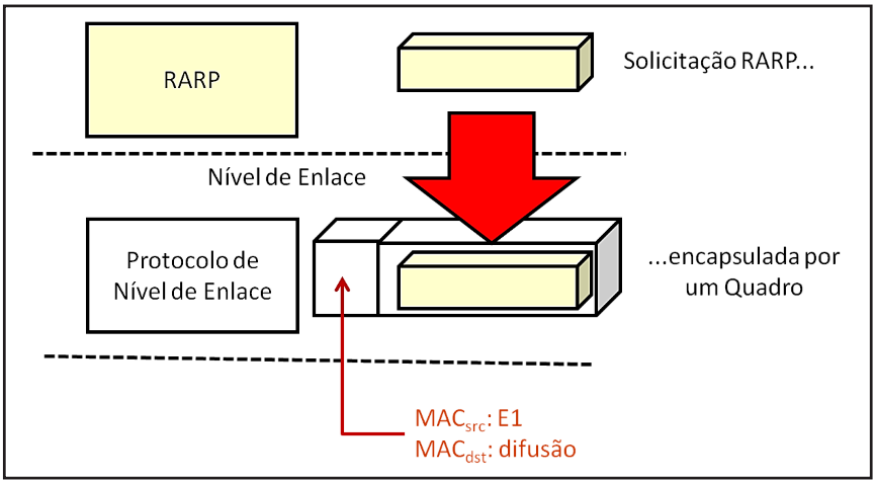
A Figura 149 ilustra como funciona a atribuição de endereço IP no RARP. Ao ser inicializado, um *host* difunde uma solicitação RARP através da rede. Esta solicitação contém o seu endereço MAC, E1. O servidor RARP responde a solicitação com o endereço IP 192.31.65.5. Este endereço IP encontra-se associado ao endereço MAC, internamente, no servidor RARP – *a priori*, como forma de controle de acesso, ou, posteriormente, ao ato da atribuição.

Figura 149: Troca de mensagens RARP para obtenção de endereço IP dinamicamente



O RARP, como indica sua própria denominação, é muito semelhante ao RARP. O próprio formato da PDU é o mesmo – apenas um campo indica se a PDU é uma PDU ARP ou RARP. Sua localização na pilha de protocolos seria ao lado do ARP, conforme ilustra a Figura 150, que exibe o encapsulamento de uma solicitação RARP em um quadro cujo endereço MAC de destino é o de difusão.

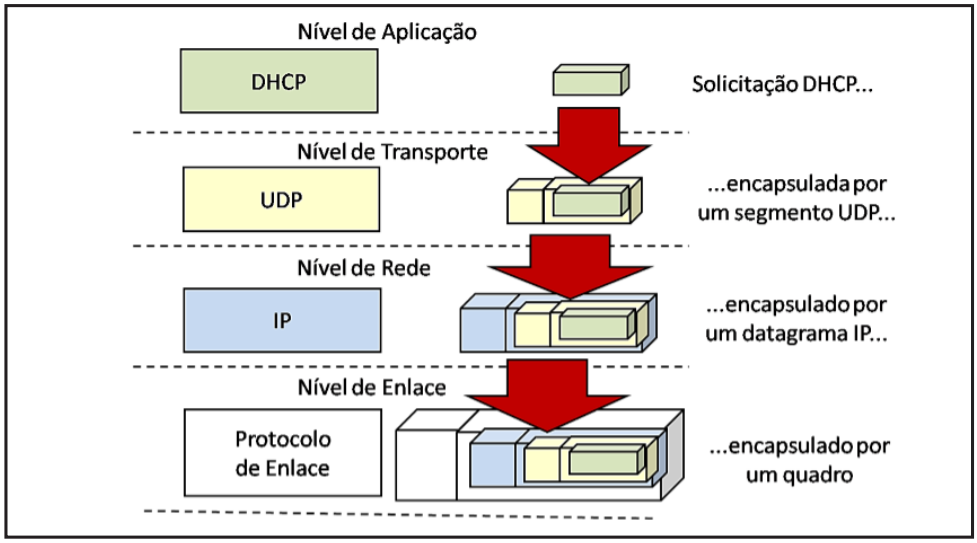
Figura 150: Encapsulamento de uma solicitação RARP em um quadro *broadcast*



DHCP

O DHCP (*Dynamic Host Configuration Protocol*), especificado pela RFC 2131, é o protocolo utilizado para que um *host* receba dinamicamente seus principais parâmetros: o endereço IP, a máscara de rede, o endereço de *gateway* e os endereços dos servidores DNS locais – primário e secundário. Trata-se de um protocolo cliente/servidor localizado no nível de aplicação, conforme ilustrado pela Figura 151.

Figura 151: Localização do DHCP na Arquitetura TCP/IP

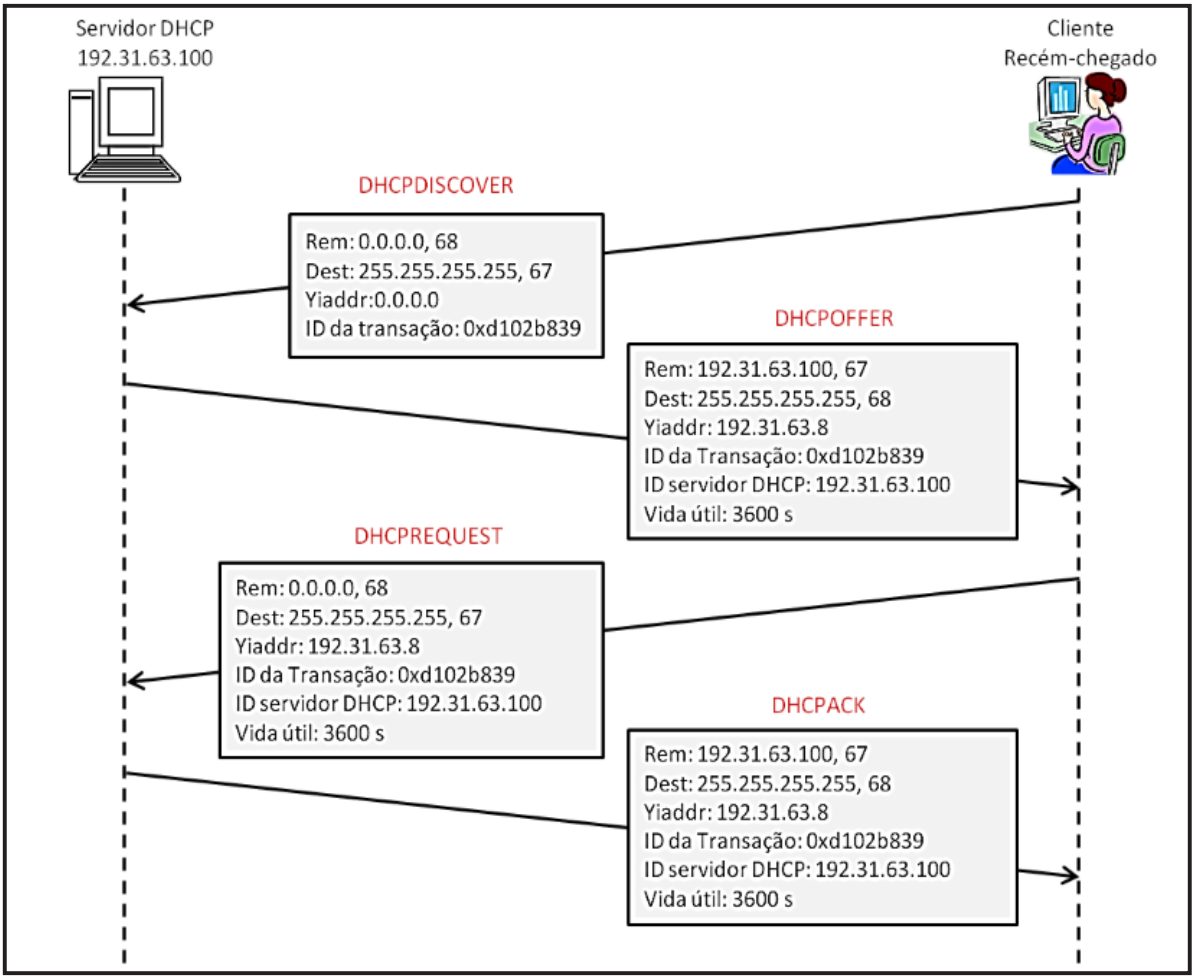


O DHCP utiliza o UDP. Mais especificamente, clientes e servidores DHCP utilizam localmente apenas as portas 68 e 67 dos seus respectivos *drivers* UDP. Os datagramas enviados por clientes durante o processo de aquisição de endereço IP possuem 0.0.0.0 como endereço de origem e 255.255.255.255 como endereço de destino. Os datagramas enviados por servidores DHCP também são endereçados a 255.255.255.255.

A Figura 152 ilustra o processo de aquisição de um endereço IP. Inicialmente, o cliente DHCP difunde mensagens DHCP*Discover*. O servidor DHCP instalado em 192.31.63.100 responde ao cliente com uma mensagem DHCP*Offer*. Ela já informa que o servidor deseja atribuir ao cliente o endereço 192.31.63.8 (valor do campo *Yiaddr,* que significa *"Your IP address"*), e que o período de validade sugerido para aquele endereço é de 3.600 segundos (uma hora). O cliente formaliza a solicitação com uma mensagem DHCP*Request*, que é respondida com uma mensagem DHCPAck. Somente a partir do momento em que recebe esta última mensagem é que o cliente DHCP tem a consciência de que possui um endereço IP.

Apesar de a Figura 152 ilustrar a obtenção apenas do endereço IP, conforme informado anteriormente, a máscara de rede, o endereço do *gateway* e os endereços dos servidores DNS locais (primário e secundário) também são fornecidos. Salienta-se que, durante todo o processo de aquisição, as mensagens DHCP possuem o mesmo identificador de transação para permitir a correlação entre elas. Uma última observação: é normal que um *host* obtenha sempre o mesmo endereço via DHCP sempre que é inicializado. Isto acontece porque aquele endereço está disponível para atribuição e porque o cliente DHCP o sugere, preenchendo com ele o campo *Yiaddr* na mensagem DHCP*Discover.*

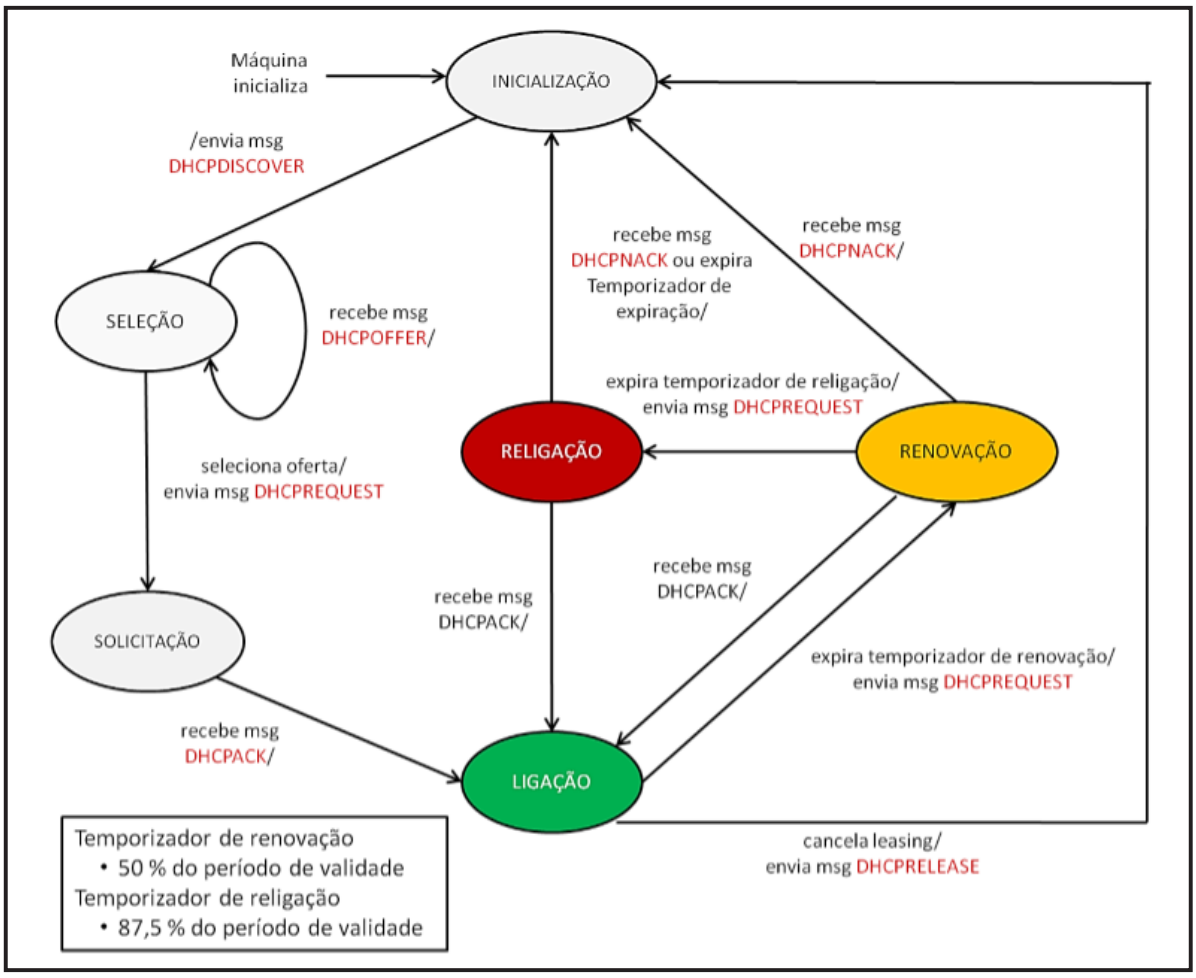
Figura 152: Processo de aquisição de um endereço IP



Uma vez atribuído, um endereço IP não permanece, indefinidamente, associado a um *host*. Conforme mencionado há pouco, é associado a um período de validade ao endereço IP fornecido a um cliente DCHP. Para continuar possuindo aquele endereço, ele deve renovar o interesse nele antes que expire o seu período de validade. Os detalhes deste processo são ilustrados pela Figura 153, uma máquina de estados que descreve o comportamento de um cliente DHCP.

O processo de aquisição do endereço IP passa pelos estados de Inicialização, Seleção (de um servidor DHCP, dentre possivelmente vários) e Solicitação (do endereço IP ao servidor selecionado). Uma vez recebido o endereço IP, o cliente encontra-se no estado de Ligação. Neste estado, ele inicializa dois temporizadores: o temporizador de renovação e o temporizador de religação.

Figura 153: Máquina de estados que descreve o comportamento de um cliente DHCP

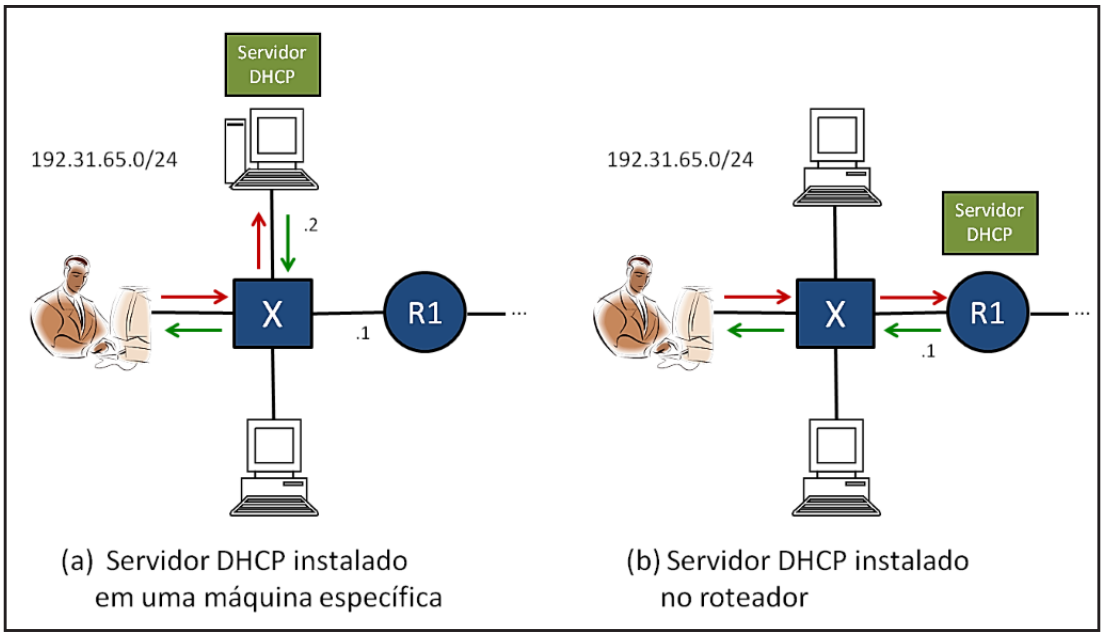


O temporizador de renovação é configurado para expirar na metade do período de validade associado ao endereço fornecido. Assim, se o período de validade for de 3.600 segundos (uma hora), o temporizador de renovação é configurado para expirar em 1.800 segundos (meia hora). Na expiração do temporizador de renovação, o cliente DHCP transmite uma nova mensagem DHCP*Request* e evolui para o estado de Renovação. Se tudo der certo, o cliente recebe uma nova mensagem DHC*PAck* e reinicializa o temporizador de renovação com base no período de validade fornecido na mensagem DHC*PAck* – tipicamente, o mesmo valor definido em mensagens anteriores.

Ao longo de toda a sua permanência com o endereço, o cliente DHCP mantém-se transitando entre os estados de Ligação e Renovação. É possível, embora improvável, que o servidor DHCP negue a renovação com uma mensagem DHCP*NAck*. De fato, há mais chances de que uma mensagem DHCP*Request* ou DHC*PAck* seja perdida ao longo do caminho, haja vista que o DHCP é um usuário do UDP – que, reconhecidamente, não realiza nenhum controle de erros. É neste contexto que expira o temporizador de religação, que é configurado para expirar em 87,5 % do período de validade. Assim, se o período de validade for de 3.600 segundos (uma hora), o temporizador de religação é configurado para expirar em 3.150 segundos (52,5 minutos). Após sua expiração, o cliente DHCP transmite uma nova mensagem DHCP*Request*. Caso esta mensagem seja respondida com uma mensagem DHC*PAck*, o cliente retorna para o estado de ligação. Por outro lado, caso ela seja respondida com uma mensagem DHCP*Nack* ou o período de validade expire, o cliente DHCP retorna ao estado de inicialização.

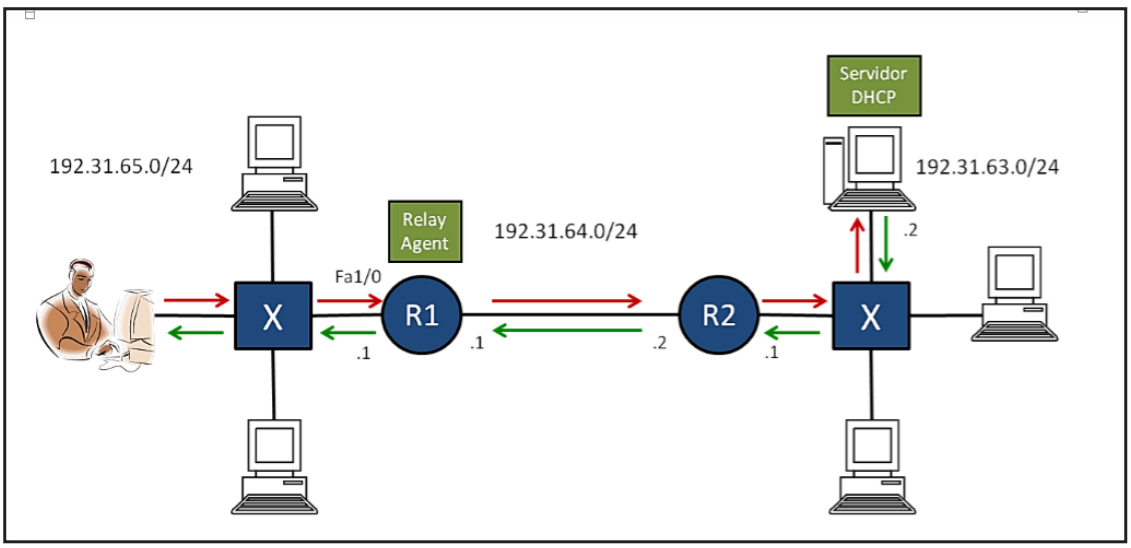
A Figura 154 ilustra as possibilidades de instalação de um servidor DHCP. Mais especificamente, o servidor DHCP pode ser instalado em uma das máquinas da rede ou (mais comumente) ser instalado em um roteador.

Figura 154: Opções de instalação para um servidor DHCP



A Figura 155 ilustra a presença de um agente de encaminhamento (*relay agent*) instalado no roteador R1. Este agente encaminha as solicitações DHCP para fora da rede onde o *host* solicitante está – na figura, o *host* solicitante está na rede 192.31.65.0/24 e o servidor DHCP pertence à rede 192.31.63.0/24. Os agentes de encaminhamento fazem com que o raio de alcance do DHCP estenda-se além da rede local a que o servidor pertence.

Figura 155: Utilização de agente de encaminhamento

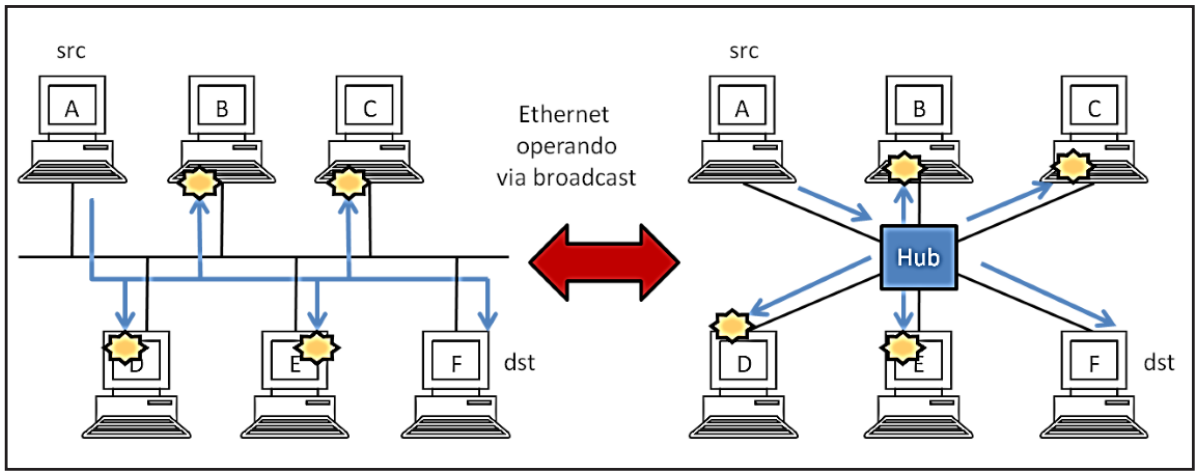


Redes Ethernet

As redes locais (LANs – *Local Area Networks*) são caracterizadas por possuírem um alcance geográfico limitado – uma sala ou uma pequena quantidade delas, por exemplo. A *Ethernet* é a tecnologia mais utilizada em redes locais cabeadas. Originalmente, ela utilizava o cabo coaxial como meio de transmissão, conforme ilustra a Figura 156. Um quadro – denominação dada à PDU do nível de enlace – endereçado apenas a um computador (*unicast*) era recebido por todos os demais computadores da rede. Aqueles computadores cujos endereços não correspondiam ao endereço de destino do quadro simplesmente o descartavam.

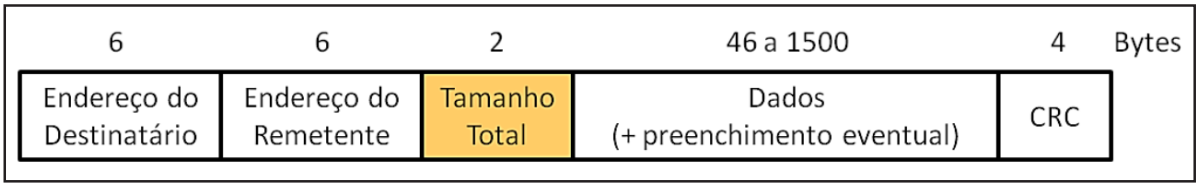
A topologia física inicial da *ethernet* possuía alguns inconvenientes: a saber, o cabo coaxial utilizado não era maleável o bastante – embora a última versão do cabo fosse mais delgada que a original; além disso, inserções e/ou remoções em um segmento requeriam a desconexão de todos os computadores. A versão seguinte da ethernet já fazia uso de um equipamento denominado *hub* para interconectar os computadores. O *hub* opera no nível físico, difundindo qualquer sinal eletromagnético recebido por uma porta para todas as suas demais portas. Estivessem os computadores conectados a um cabo coaxial ou a um hub, o fato é que estas versões eram denominadas Ethernet operando via *broadcast*.

Figura 156: Variantes da *Ethernet broadcast*



A Figura 157 ilustra o formato de um quadro *ethernet.* Os dois primeiros campos do quadro correspondem aos seus Endereços de Destinatário e Remetente. Em particular, endereços do nível de enlace são também denominados endereços MAC (*Medium Access Control*). Endereços MAC *Ethernet* são compostos por 6 (seis) octetos, expressos em notação hexadecimal. Um exemplo desta representação é o endereço 49-BD-D2-C7-56-2A. Diferentemente dos endereços IP, os endereços MAC *Ethernet* já vêm permanentemente gravados na placa de rede. Para evitar duplicidade de endereço, o IEEE designa o valor dos três primeiros octetos para cada fabricante de placas de redes; portanto, cada um daqueles fabricantes possui capacidade de distribuir até 2^24 endereços MAC diferentes. Quadros *broadcast*, necessários à transmissão de solicitações ARP e mensagens DHCP *Discover*, possuem como Endereço de Destinatário o endereço de *broadcast* FF-FF-FF-FF-FF-FF.

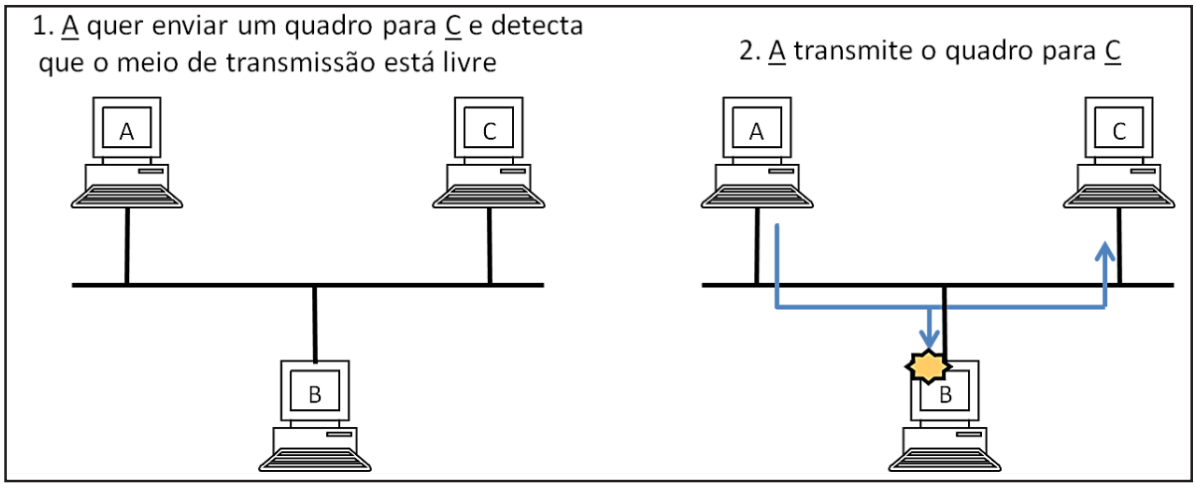
Figura 157: Formato do quadro *Ethernet*



Conforme se pode perceber na Figura 157, o campo de dados possui um tamanho que vai de 46 a 1.500 bytes. Se os dados encapsulados não totalizarem 46 bytes, bits de preenchimento são utilizados até que aquela marca seja atingida. Por outro lado, os 1.500 Bytes correspondem a MTU da *Ethernet.* Estes valores-limite fazem com que quadros Ethernet não possam ser menores que 64 bytes nem maiores que 1.518 bytes. O tamanho mínimo de 64 bytes foi calculado para que, no pior caso, a rede continue funcionando adequadamente – maiores detalhes sobre isso serão fornecidos mais adiante. Por outro lado, o tamanho máximo de 1.518 bytes foi estabelecido com base no custo da memória na época da especificação da Ethernet, 1978 – se a especificação fosse feita nos dias de hoje, certamente o tamanho máximo do quadro seria maior. O valor do campo Tamanho Total informa, como indica sua própria denominação, o tamanho do quadro. O campo CRC (*Cyclic Redundancy Check*) é utilizado apenas para a detecção de erros em um quadro – em particular, quadros *Ethernet* não são retransmitidos.

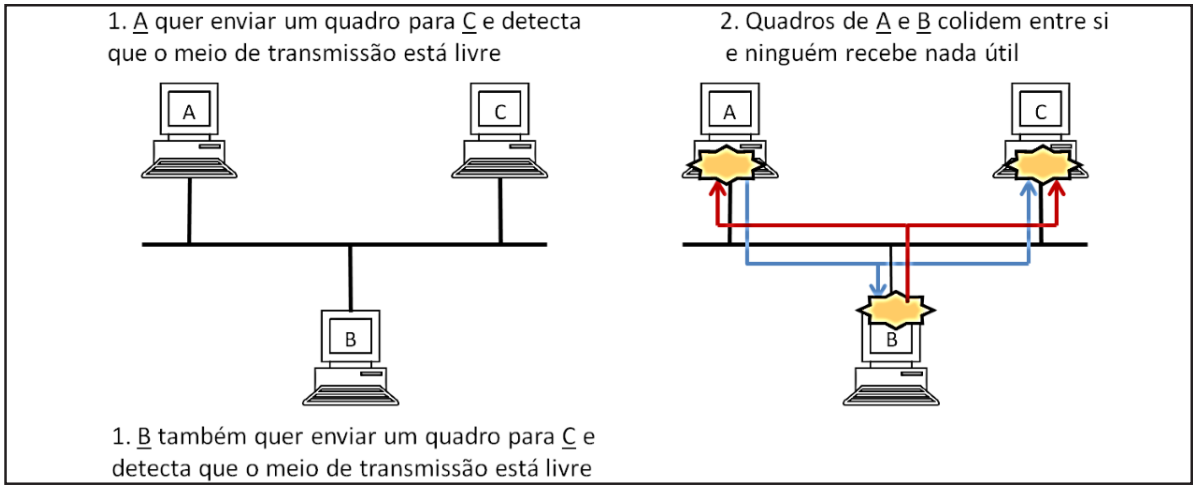
Os computadores que fazem parte de uma rede local *Ethernet* que opera através de *broadcast* utilizam uma estratégia denominada CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*) para transmitir seus quadros. A Figura 158 ilustra o quão simples é esta estratégia: quando um computador deseja transmitir um quadro, ele verifica se o meio de transmissão está livre – isto é, se não há nenhum outro computador transmitindo um quadro. Se o meio estiver livre, o computador transmite o quadro; por outro lado, se o meio estiver ocupado, o computador aborta a transmissão até que ele perceba que o meio está livre.

Figura 158: Exemplo de aplicação do CSMA/CD



Dois ou mais computadores podem querer transmitir um quadro. Eles verificam que o meio de transmissão está livre e transmitem seus quadros simultaneamente. Os quadros colidem entre si – os sinais eletromagnéticos se embaralham e nenhum dos quadros é compreendido. A Figura 159 ilustra um exemplo de colisão entre quadros.

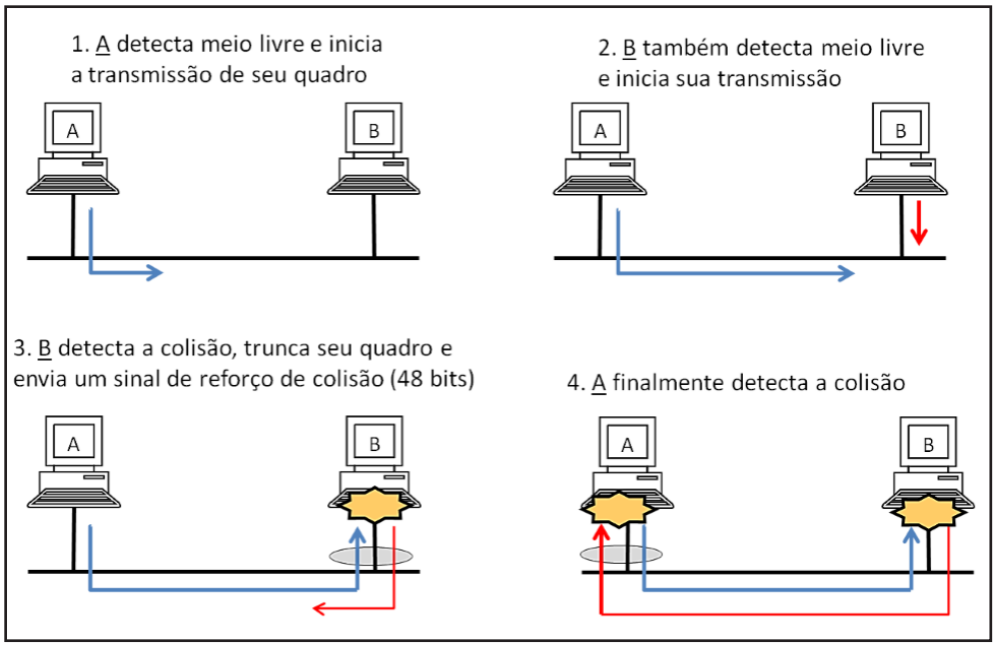
Figura 159: Exemplo de colisão entre quadros



Quando ocorre uma colisão, os computadores envolvidos executam o Algoritmo de Recuo Binário Exponencial. Após a colisão, os computadores dividem o tempo em *slots* discretos, cujo tamanho é igual ao pior tempo de propagação de ida e volta – que é de 51,2 μs, considerados a partir da extensão máxima de um segmento *Ethernet*. Depois da primeira colisão, cada computador espera 0 ou 1 *slot* antes de retransmitir o seu quadro. Naturalmente, se dois computadores escolherem a mesma quantidade de *slots*, seus quadros colidirão novamente. Caso haja uma segunda colisão, cada computador escolhe ao acaso 0, 1, 2 ou 3 *slots* e aguarda durante esse tempo até retransmitir seu quadro. Havendo uma terceira colisão, cada computador escolhe ao acaso 0, 1, 2, 3, 4, 5, 6 ou 7 *slots* e aguarda durante esse tempo até retransmitir seu quadro. Em termos gerais, após a i-ésima colisão, a quantidade de slots a esperar é um número aleatório entre 0 e 2^i-1 *slots*. Entretanto, após alcançadas dez colisões, a quantidade de opções para escolha será mantida em 1.023 *slots*. Finalmente, após 16 colisões consecutivas as tentativas de transmissão são definitivamente abortadas.

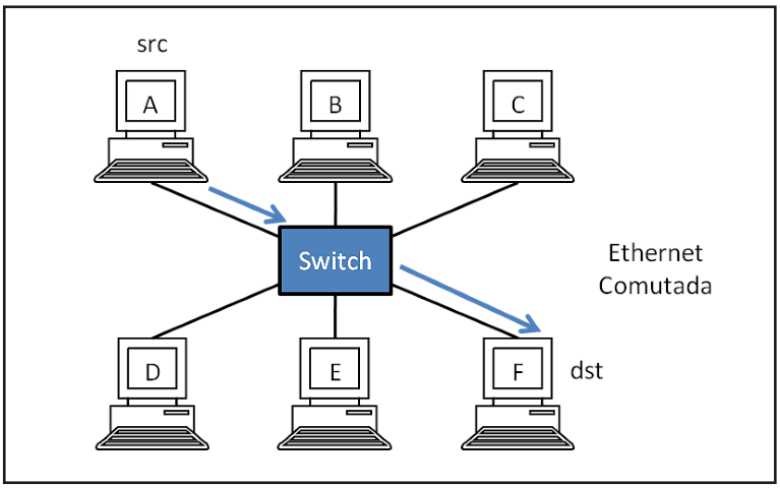
O valor de 64 bytes para o tamanho mínimo de um quadro *Ethernet* foi definido para que, mesmo quando dois computadores estiverem o mais afastado possível em um segmento *Ethernet*, qualquer um deles seja capaz de transmitir um quadro e detectar quando ele sofre uma eventual colisão. A Figura 160 ilustra este, que é considerado o pior caso para se transmitir um quadro e detectar a sua colisão. Ao observá-la, é possível perceber que, se um quadro transmitido for pequeno demais, uma eventual colisão pode não ser detectada. Assim, é importante que aquele quadro possua um tamanho mínimo. O valor de 64 bytes leva em consideração a distância máxima a que dois computadores podem estar um do outro em um segmento Ethernet e a taxa de transmissão naquele segmento.

Figura 159: Exemplo de colisão entre quadros



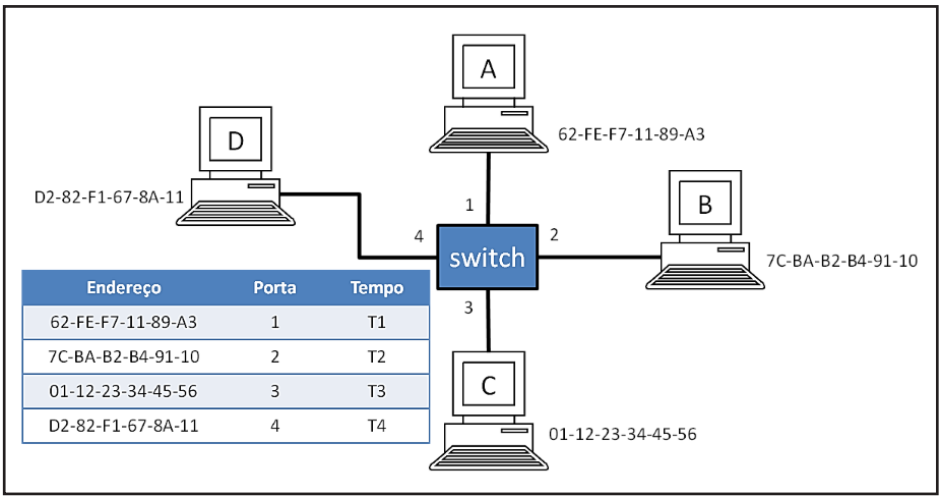
A última versão da *Ethernet*, ilustrada pela Figura 161, substituiu o *hub* por outro equipamento denominado switch, que opera no nível do enlace, examinando o endereço de destino de um quadro e, com base nele, encaminhando-o para apenas uma de suas portas. Este processo é denominado comutação, e por esta razão esta última versão da *Ethernet* é denominada *Ethernet* comutada. Pelo fato de o *switch* possuir capacidade de armazenamento temporário de quadros, o CSMA/CD não é utilizado em uma *Ethernet* comutada.

Figura 161: A presença de um *switch* caracteriza a *Ethernet comutada*



O *switch* realiza a comutação de quadros com base em sua tabela de comutação. Cada entrada desta tabela contém um endereço MAC associado a uma porta. Conforme ilustra a Figura 162, cada entrada contém também uma marca de tempo que é criada no momento da inserção da entrada, e atualizada sempre que um quadro destinado àquele endereço MAC é recebido pelo *switch;* se o *switch* não recebe um quadro destinado àquele endereço MAC até um determinado período (cinco minutos, em média), a entrada é removida. O motivo da existência da marca de tempo é evitar que uma entrada permaneça, indefinidamente, na tabela de comutação quando o computador associado não mais fizer parte do segmento *Ethernet.*

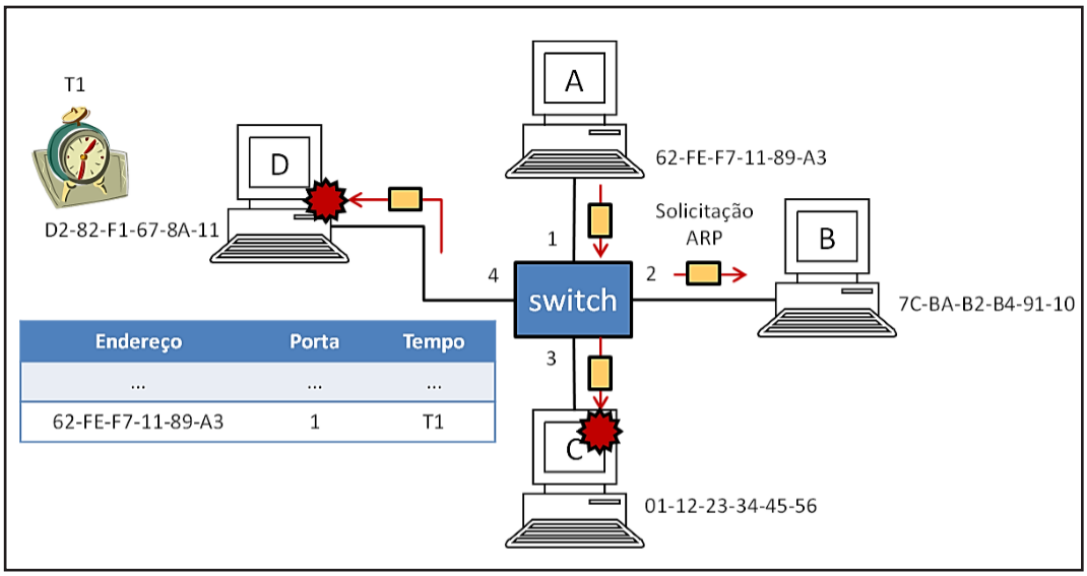
Figura 162: Exemplo de tabela de comutação



Endereço	Porta	Tempo
62-FE-F7-11-89-A3	1	T1
7C-BA-B2-B4-91-10	2	T2
01-12-23-34-45-56	3	T3
D2-82-F1-67-8A-11	4	T4

A tabela de comutação é construída com base nos endereços MAC dos remetentes. Na Figura 163, o computador A deseja enviar um datagrama destinado a B. Porém, ele não conhece o endereço MAC de B. Portanto, A envia uma solicitação ARP e o quadro que a encapsula é recebido pelo *switch.* Ele associa o endereço MAC de remetente do quadro (62-FE-F7-11-89-A3, o endereço MAC de A) à porta na qual ele foi recebido (1). Uma marca de tempo é criada para a entrada inserida. Em seguida, o *switch* difunde o quadro por todas as suas demais portas – afinal de contas, uma solicitação ARP é encapsulada em um quadro *broadcast.*

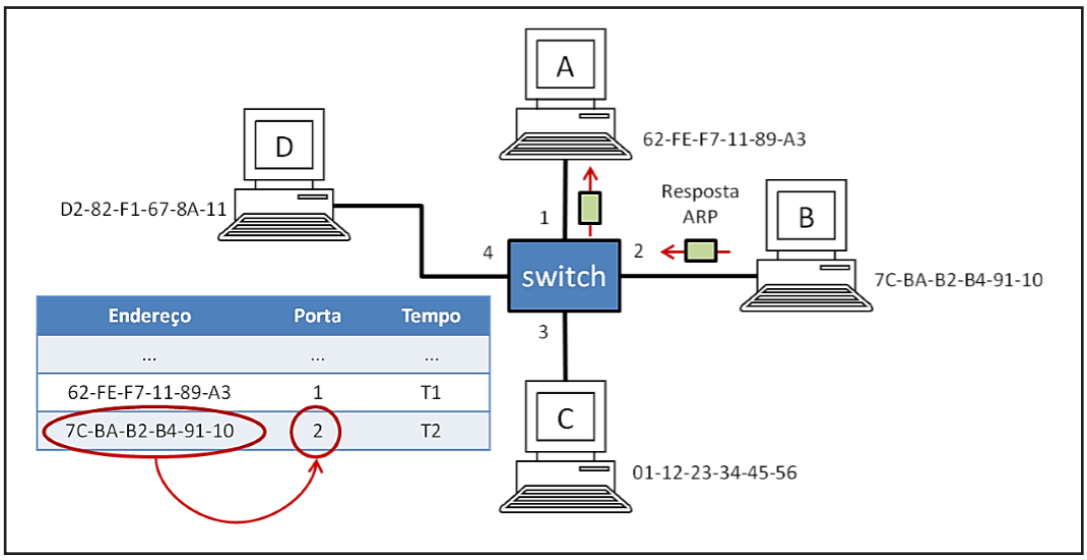
Figura 163: Construção da tabela de comutação: associação do endereço MAC de A a porta 1



A Figura 164 ilustra a transmissão da resposta ARP por parte de B, encapsulada por um quadro *unicast* destinado a A. Quando o *switch* recebe aquele quadro, ele associa seu endereço MAC de remetente (7C-BA-B2-B4-91-10, o endereço MAC de B) à porta na qual ele foi recebido (2). Uma marca de tempo também é associada à entrada recém-inserida.

A consulta de uma tabela de comutação é realizada com base no endereço MAC do destinatário. Assim, ao se deparar com o quadro *unicast* que encapsula a resposta ARP, o *switch* consulta sua tabela de comutação e verifica que o endereço MAC de destinatário do quadro (62-FE-F7-11-89-A3, o endereço MAC de A) está associado à porta 1. Assim, ele encaminha o quadro (somente) por aquela porta.

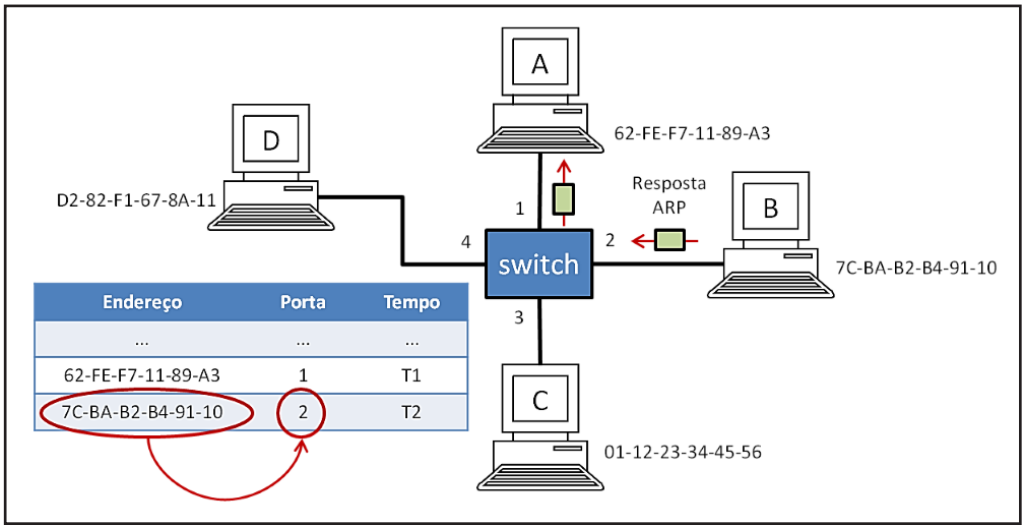
Figura 164: Construção da tabela de comutação: associação do endereço MAC de B a porta 2



Uma vez que a resposta ARP chega a A, ele pode transmitir agora o seu datagrama para B. A Figura 165 ilustra este procedimento. O datagrama é encapsulado em um quadro cujo endereço MAC de destinatário é 7C-BA-B2-B4-91-10 (o endereço MAC de B). Ao receber este quadro, o

switch consulta a sua tabela de comutação e verifica que aquele endereço MAC está associado à porta 2. Assim, ele encaminha o quadro (somente) por aquela porta.

Figura 165: Exemplo de consulta na tabela de comutação



A *Ethernet* vem sofrendo uma evolução contínua ao longo do tempo. A denominada *Ethernet* clássica recebeu do IEEE (*Institute of Electrical and Electronics Engineers*) a denominação 802.3. Ela operava *half-duplex* a 10 Mbps e permitia (com o auxílio de equipamentos denominados repetidores de sinal) que dois computadores estivessem separados a uma distância máxima de 2,5 Km. Esta distância é denominada diâmetro do segmento.

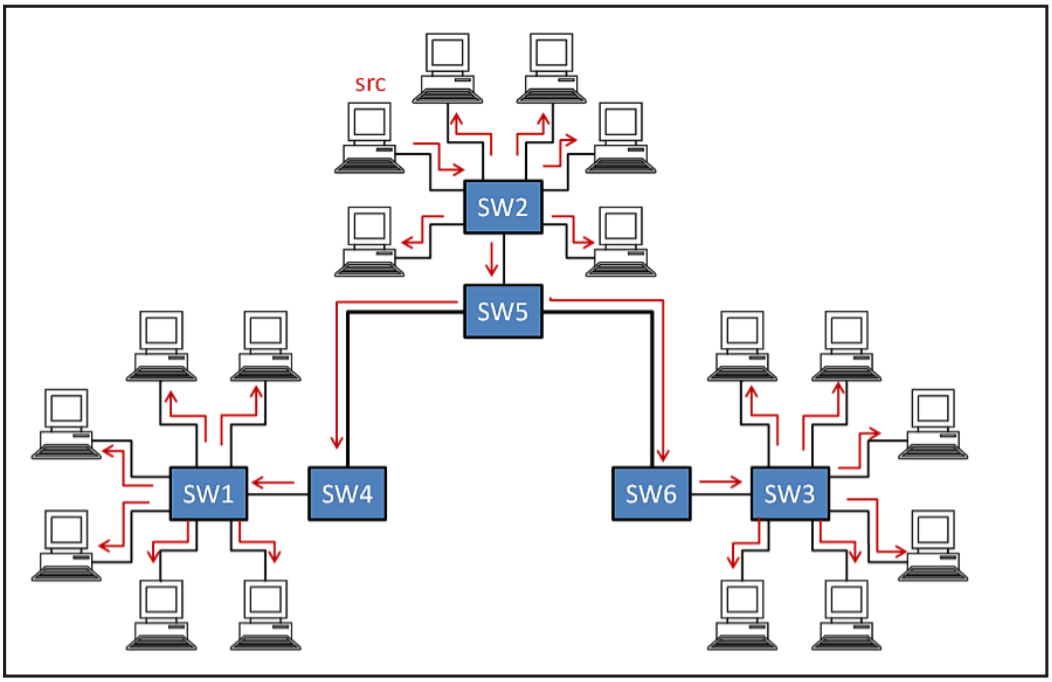
Novas técnicas de transmissão permitiram o surgimento da *Fast Ethernet*, que recebeu do IEEE a denominação 802.3u (a última letra é uma alusão à palavra *upgrade*). A *Fast Ethernet* é a versão da *Ethernet* mais popular em redes locais, operando *full-duplex* a 100 Mbps. Este aumento em dez vezes na taxa de transmissão resultou na redução proporcional no diâmetro do segmento, para manter o tamanho mínimo do quadro em 64 Bytes. Assim, o diâmetro de um segmento *Fast Ethernet* é de 250 metros.

A *Gigabit Ethernet* surgiu com um aumento em dez vezes na taxa de transmissão, com relação à *Fast Ethernet*. Ela opera, portanto, a 1 Gbps. Para manter o mesmo diâmetro de um segmento Fast *Ethernet*, o tamanho mínimo de um quadro foi aumentado para 512 bytes. Para conseguir este valor quando os dados encapsulados pelo quadro não são suficientes para tal, duas estratégias podem ser utilizadas: a extensão de portadora consiste simplesmente na inserção de bytes de preenchimento para se atingir o novo tamanho mínimo; outra estratégia é denominada de rajada de quadros, e consiste na retenção temporária de quadros para que a sequência transmitida atinja o tamanho mínimo. A *Gigabit Ethernet* recebeu a especificação 802.3z, e a última letra era uma clara alusão à ideia de que aquela seria a última versão da *Ethernet* – o que não corresponde à realidade, posto que hoje a 10 *Gigabit Ethernet* (especificada pelo IEEE como 802.3ab) é capaz de transmitir a 10 Gbps. A *Gigabit Ethernet* é uma tecnologia visando à utilização em *backbones.*

FTP

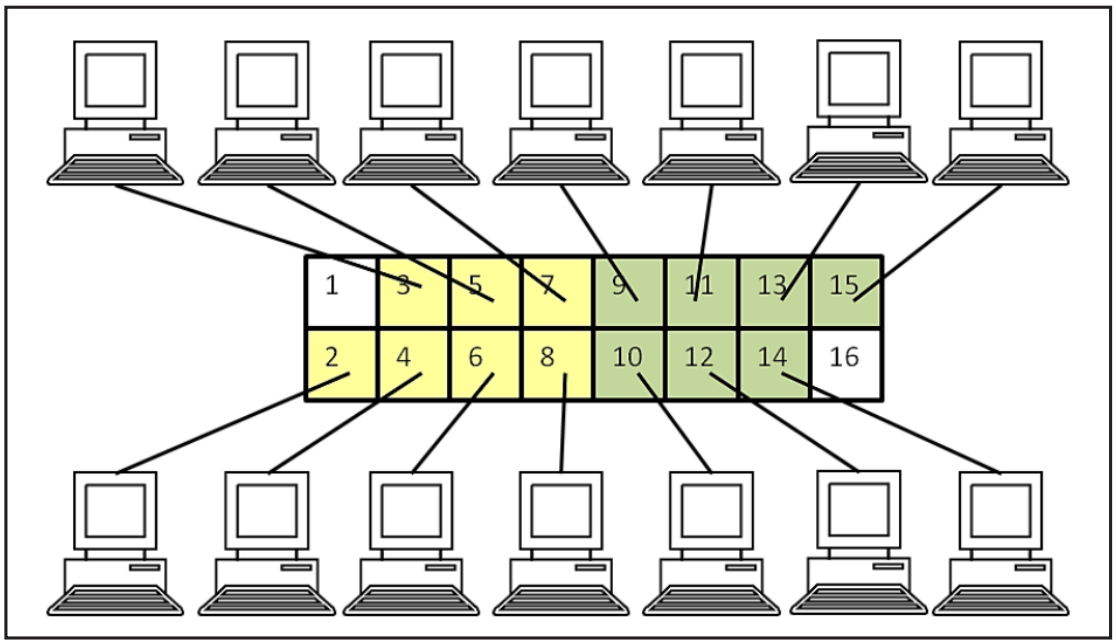
Qual é o melhor equipamento a ser utilizado? *Switches* ou roteadores? Há bons argumentos a favor dos primeiros: para começar, eles são *plug-and-play* – suas tabelas de comutação são preenchidas automaticamente; roteadores, por outro lado, necessitam de intervenção humana ou de protocolos de roteamento para preencher suas tabelas de rota. Outro argumento favorável aos *switches* é o fato de que, por eles operarem no nível de enlace, seriam conceitualmente mais rápidos que os roteadores, que operam no nível de rede. Porém, há um inconveniente com *switches* que não ocorre com os roteadores: os *switches* não filtram tráfego *broadcast* como os roteadores o fazem. Assim, quando um quadro broadcast – que encapsula diretamente uma solicitação ARP ou indiretamente uma mensagem DHCP *Discover* – chega a um *switch* por uma porta, ele o encaminha por todas as suas demais portas. A difusão indiscriminada de quadros broadcast pode dar origem à denominada tempestade de *broadcast*, ilustrada pela Figura 166.

Figura 166: Exemplo de tempestade de *broadcast*



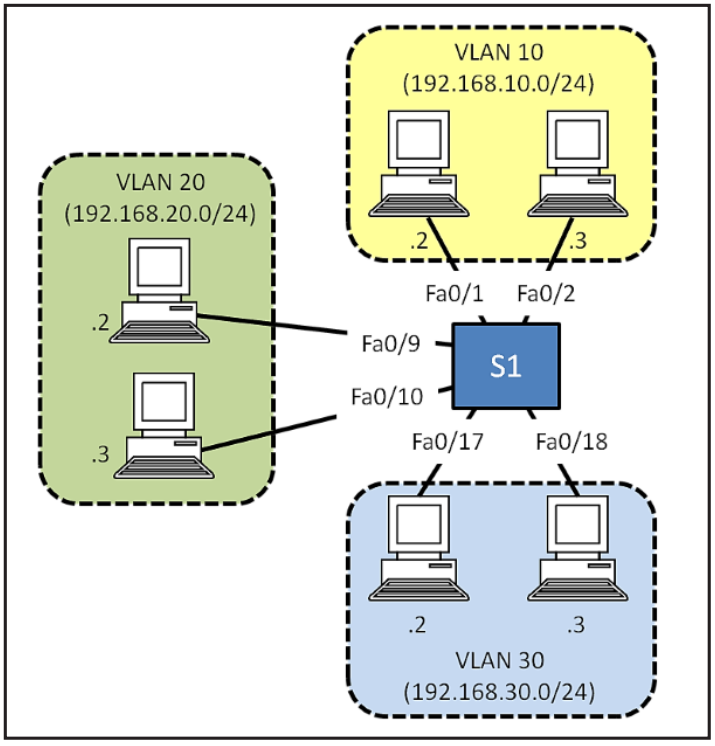
O conceito de VLAN (Virtual LAN) combate a tempestade de *broadcast*. Um *switch* que comporta VLANs possui suas portas divididas em diversas VLANs. A Figura 167 ilustra um *switch* que possui 16 portas, sendo sete delas pertencentes à VLAN amarela (as portas de 2 a 8) e outras sete pertencentes à VLAN verde (as portas de 9 a 15). Um quadro *broadcast* recebido através da porta 2, por exemplo, seria difundido apenas pelas demais portas da VLAN amarela (as portas 3, 4, 5, 6, 7 e 8). Um *switch* que comporta VLANs possui tantas tabelas de comutação quantas forem as VLANs comportadas – assim, o *switch* da Figura 167 possui duas tabelas de comutação: uma associada à VLAN amarela e outra associada à VLAN verde. Porém, a menos que o *switch* possua a característica de comutação entre VLANs ele não permite que quadros originados na VLAN amarela e destinados à VLAN verde cheguem a seu destino; o caminho inverso também não é possível, isto é, quadros originados na VLAN verde e destinados à VLAN amarela não chegam ao seu destino.

Figura 167: Um *switch* comportando duas VLANs



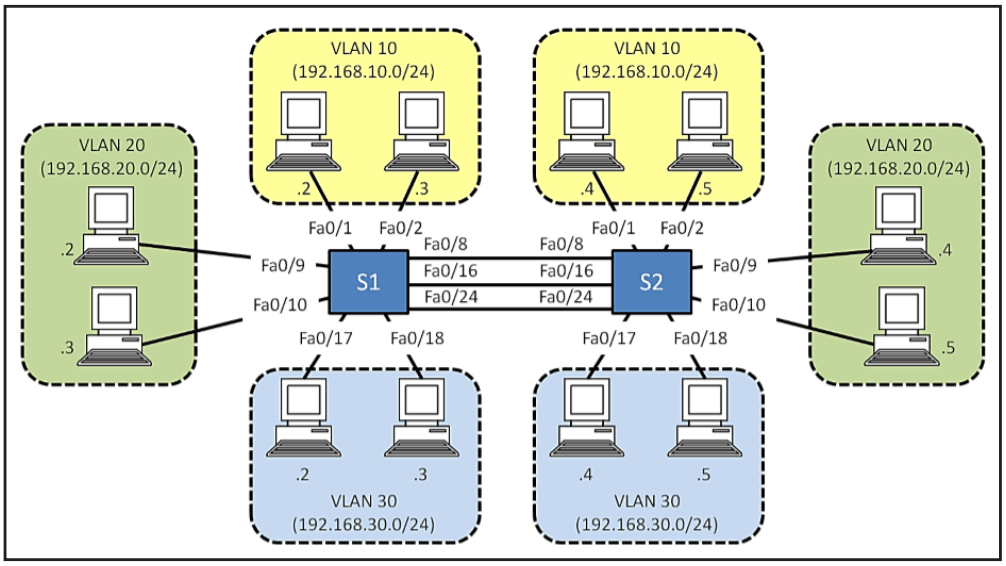
A Figura 168 ilustra um *switch* (S1) suportando três VLANs distintas, identificadas como 10, 20 e 30. Estas VLANs comportam as redes 192.168.10.0/24, 192.168.20.0/24 e 192.168.30.0/24, respectivamente. Até então, supõe-se que estas VLANs estejam incomunicáveis.

Figura 168: Um *switch* comportando três VLANs



A Figura 169 ilustra a extensão das VLANs 10, 20 e 30 através da adição do *switch* S2. Três portas são utilizadas em cada *switch* para a transmissão de quadros referentes às VLANs. Em particular, as portas Fa0/8, Fa0/16 e Fa0/24 de cada *switch* pertencem às VLANs 10, 20 e 30, respectivamente. Elas são utilizadas para encaminhar quadros entre computadores pertencentes a uma mesma VLAN que estão conectados a *switches* diferentes.

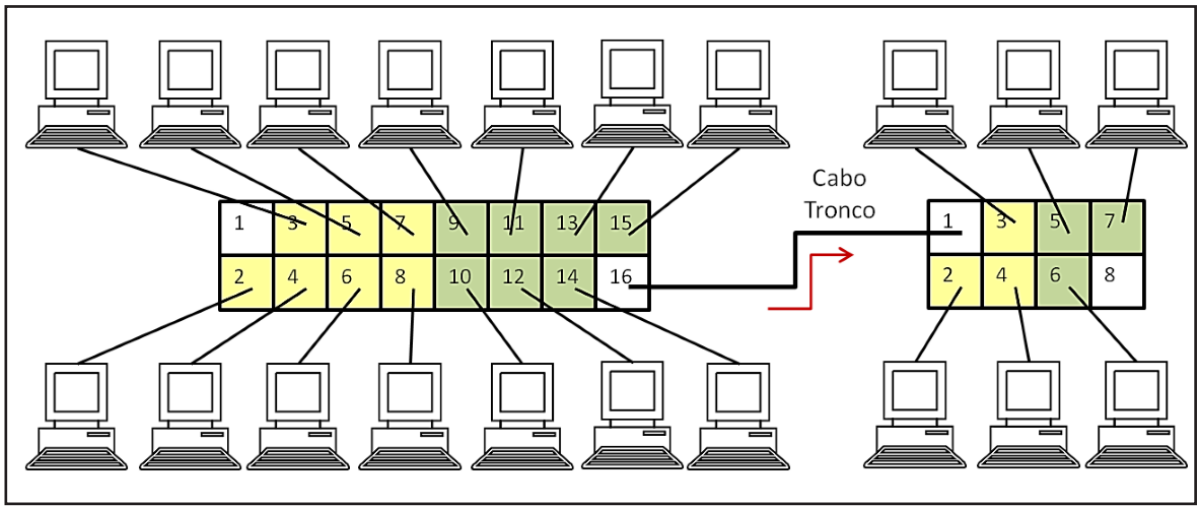
Figura 169: Extensão das três VLANs anteriores com a adição de S_2



A solução apresentada pela Figura 169 para a comunicação intra-VLAN possui o inconveniente de alocação de portas específicas para o tráfego de quadros entre computadores pertencentes à mesma VLAN, porém conectados a *switches* diferentes. Estas portas não estão, portanto, disponíveis para a conexão de computadores a elas. Além disso, a quantidade de portas necessárias aumenta proporcionalmente à quantidade de VLANs comportadas que se deseja estender.

Uma solução alternativa à extensão de VLANs é a utilização de portas tronco em cada *switch*, conforme ilustrado pela Figura 170. As portas tronco de cada *switch* são conectadas através de um cabo tronco, através do qual são transmitidos quadros das VLANs amarela e verde. Neste contexto, porém, como um *switch* que recebe um quadro através de uma porta tronco sabe a qual VLAN aquele quadro pertence?

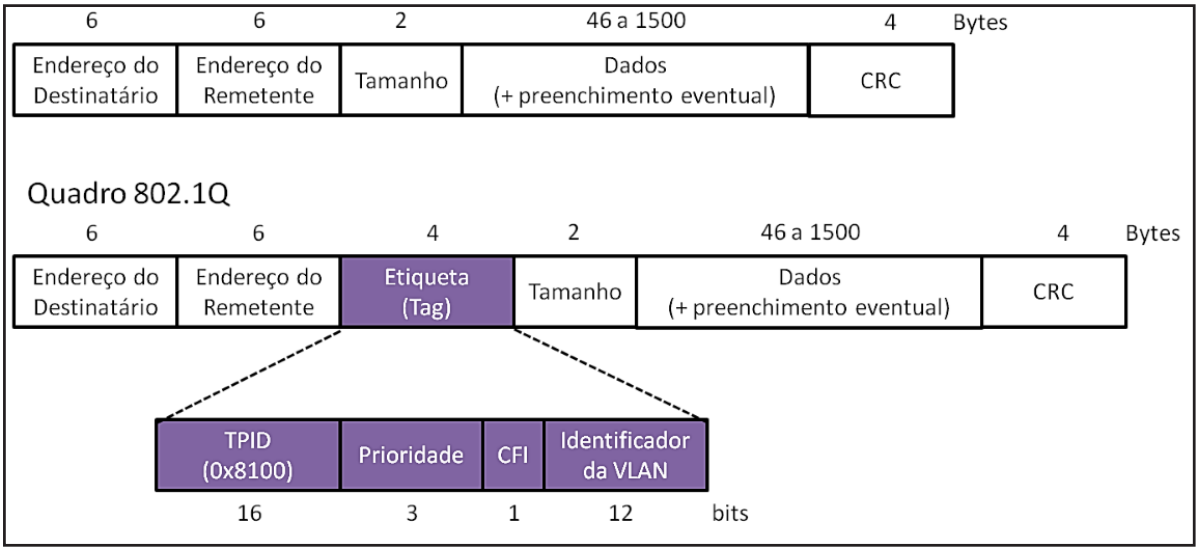
Figura 170: Portas tronco sendo utilizadas para a conexão de dois *switches*



O padrão IEEE 802.1Q responde àquela pergunta, inserindo um campo no quadro *Ethernet* para identificar a qual VLAN aquele quadro pertence. Conforme ilustra a Figura 171, este campo é comumente denominado etiqueta. Quadros etiquetados estão presentes apenas em cabos tronco. Assim, tomando como base a Figura 170, a etiqueta é inserida quando um quadro deixa a

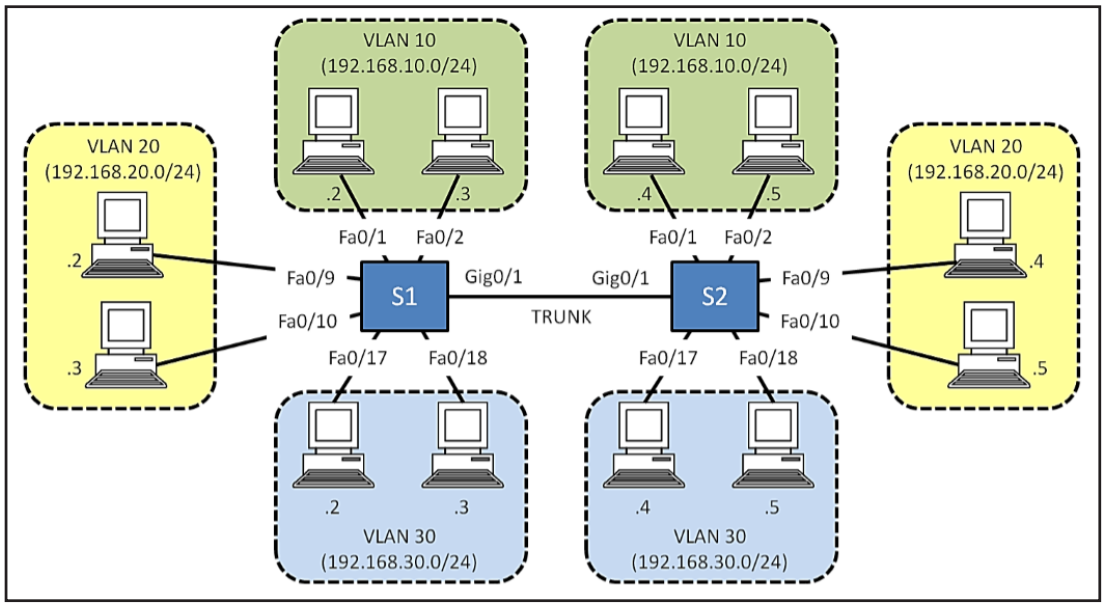
porta 16 do *switch* da esquerda e é removida quando ele chega à porta 1 do *switch* da direita. Um quadro trafegando no sentido contrário tem sua etiqueta inserida assim que deixa a porta 1 do *switch* da direita e removida assim que é recebida pela porta 16 do *switch* da esquerda.

Figura 171: Etiqueta 802.1Q



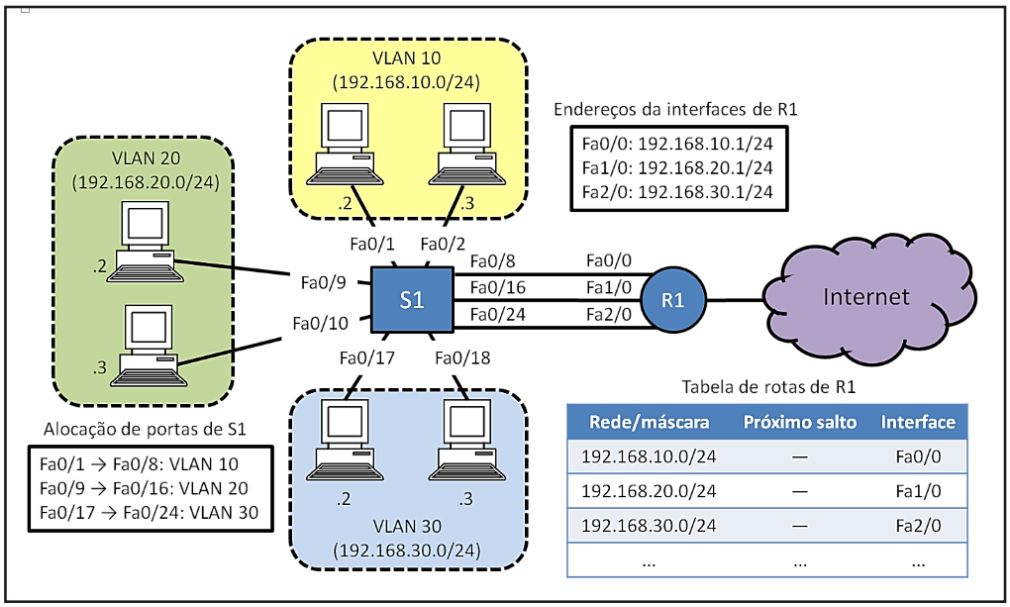
A Figura 172 ilustra a utilização de um cabo tronco para a extensão de VLANs a mais de um *switch*. Trata-se de uma solução mais eficiente que aquela ilustrada pela Figura 169, haja vista que apenas uma porta de cada *switch* é utilizada para a transmissão de quadros de todas as três VLANs.

Figura 172: Utilização de um cabo tronco para estender VLANs a mais de um *switch*



Em todas as configurações exibidas até então, as VLANs permaneceram incomunicáveis. A comunicação entre VLANs pode ser realizada de várias formas. Algumas delas incluem um roteador. Na Figura 173, o roteador R_1 possui suas interfaces conectadas a portas do *switch* S1 que fazem parte das VLANs 10, 20 e 30. As interfaces são configuradas com os endereços de *gateway* para as redes 192.168.10.0/24, 192.168.20.0/24 e 192.168.30.0/24. Datagramas originados em uma VLAN e destinados a outra VLAN são roteados por R1 com base nas entradas de sua tabela de rotas alusivas às redes diretamente conectadas.

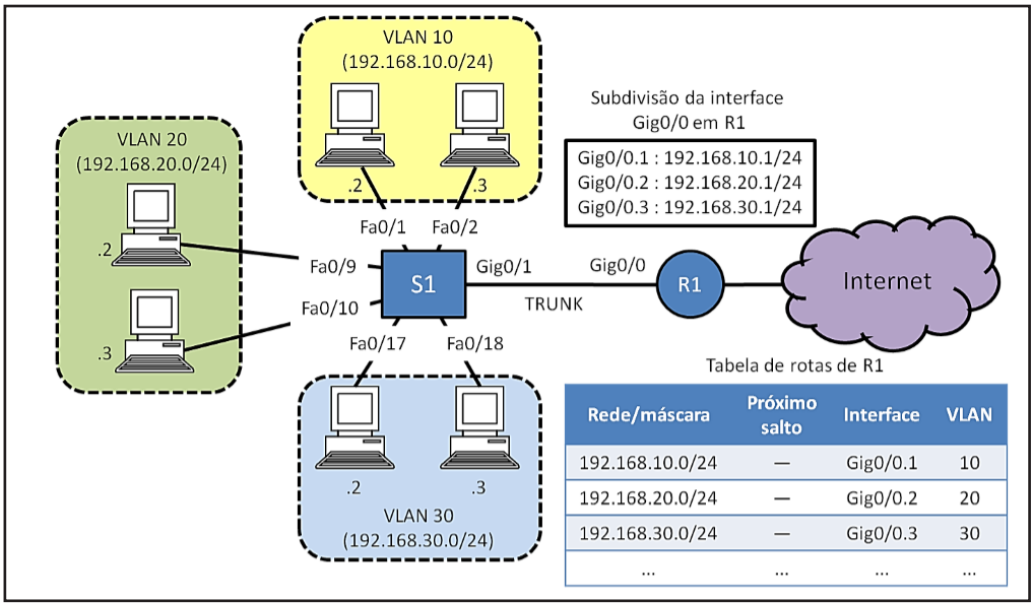
Figura 173: Comunicação entre VLANs intermediada por um roteador com três interfaces



A Figura 174 exibe uma versão mais eficiente da utilização de um roteador para intermediar a comunicação entre VLANs, baseada na utilização de um único cabo tronco e da subdivisão da interface Gig0/0 de R_1. Cada uma das subinterfaces está associada a uma das VLANs comportadas. Este arranjo é denominado *router on a stick.*

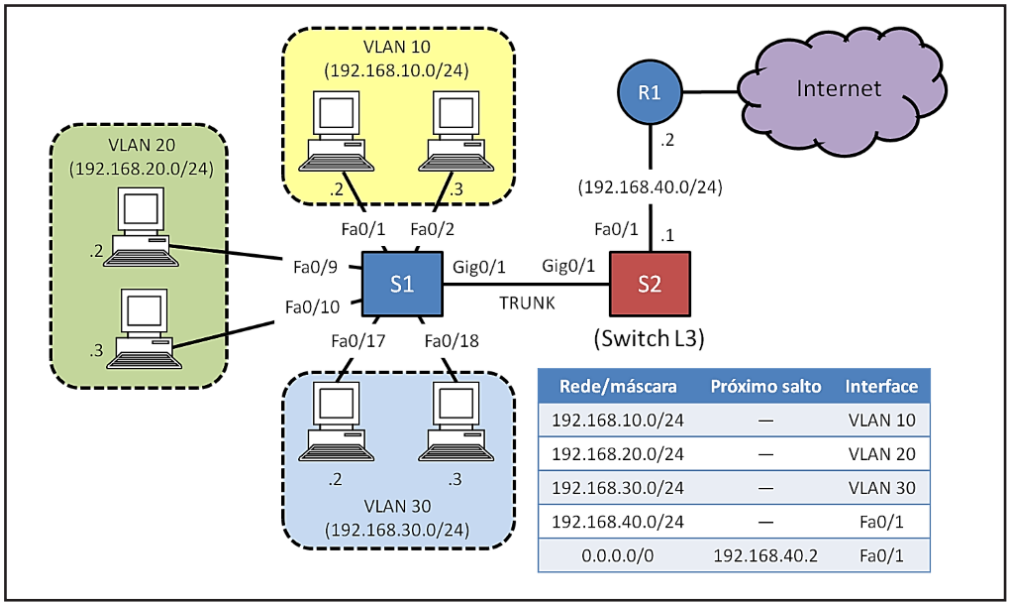
Supõe-se como exemplo o envio hipotético de um datagrama por 192.168.10.2 (localizado na VLAN 10) para 192.168.20.2 (localizado na VLAN 20). O quadro que encapsula este datagrama é transmitido por 192.168.10.2 e chega à porta Fa0/2 de S_1; o quadro em questão não é etiquetado. O *switch* S1 examina sua tabela de comutação associada à VLAN 10 e conclui que aquele quadro deve ser encaminhado pela sua porta tronco Gig0/1. Porém, antes de fazê-lo, S_1 insere uma etiqueta no quadro que o identifica como pertencente à VLAN 10. Quando o quadro chega ao roteador R_1, o datagrama é desencapsulado e roteado com base na tabela de rotas ilustrada pela Figura 174, segundo a qual o datagrama é encapsulado por um quadro cuja etiqueta agora o identifica como pertencente à VLAN 20. O quadro é transmitido através do cabo tronco e chega a S_1, que examina sua tabela de comutação associada à VLAN 20 e descobre que deve encaminhar o quadro pela porta Fa0/9. Antes de fazê-lo, porém, S_1 remove a sua etiqueta.

Figura 174: *Router on a stick*



Outra forma de fazer com que diversas VLANs se comuniquem entre si é por meio da substituição do roteador por um *switch* L3. Esta forma é ilustrada pela Figura 175. O *switch* L3 é diferente dos outros *switches*; além de ser capaz de etiquetar quadros, como todo switch que comporta VLANs, ele opera no nível de rede. Sua tabela de rotas inclui entradas associadas a interfaces virtuais que representam as VLANs comportadas. O *switch* L3 é uma solução menos dispendiosa que a utilização de roteadores.

Figura 175: *Switch* L3 substituindo o roteador

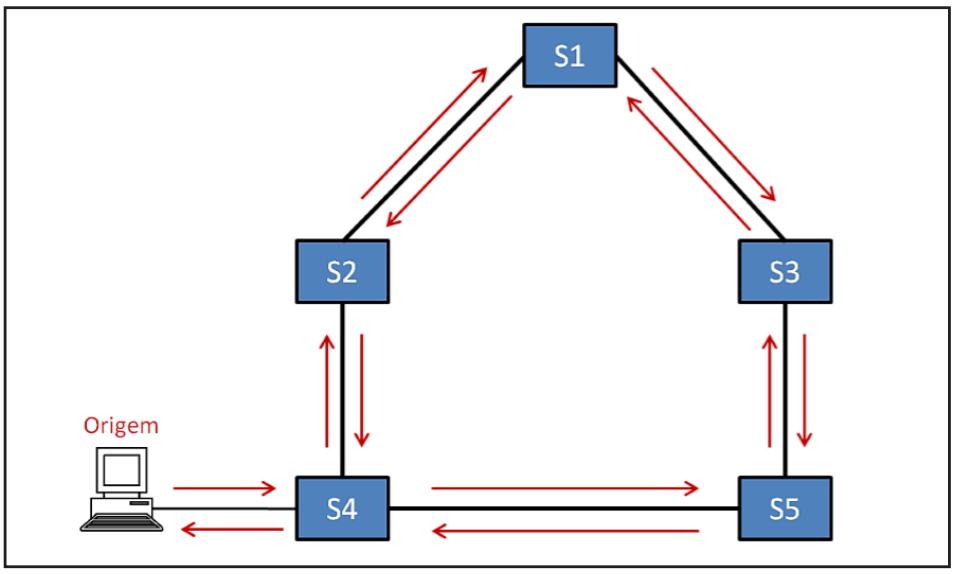


Rede/máscara	Próximo salto	Interface
192.168.10.0/24	—	VLAN 10
192.168.20.0/24	—	VLAN 20
192.168.30.0/24	—	VLAN 30
192.168.40.0/24	—	Fa0/1
0.0.0.0/0	192.168.40.2	Fa0/1

STP – Spanning Tree Protocol

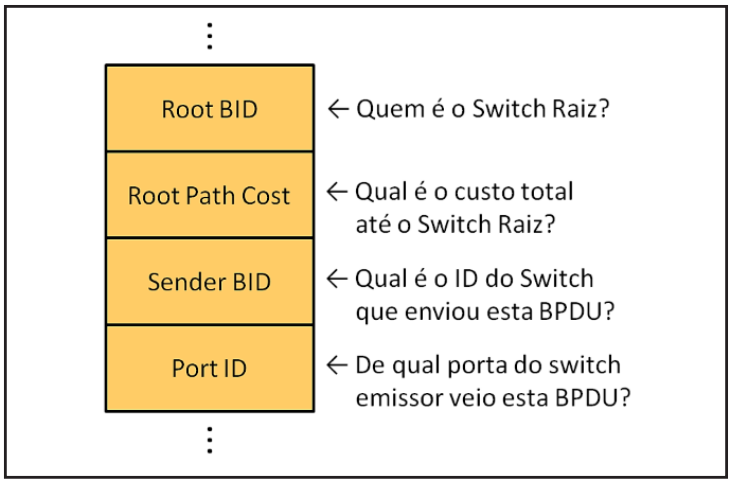
A Figura 176 ilustra uma topologia na qual um quadro *broadcast* é enviado por um computador. Em topologias como esta, em que há presença de um laço, o quadro ficaria vagando, indefinidamente, ao longo da topologia pelo fato de que seus *switches* o difundiriam ao longo de todas as suas portas.

Figura 176: Quadro *broadcast* vagando indefinidamente em uma topologia contendo um laço



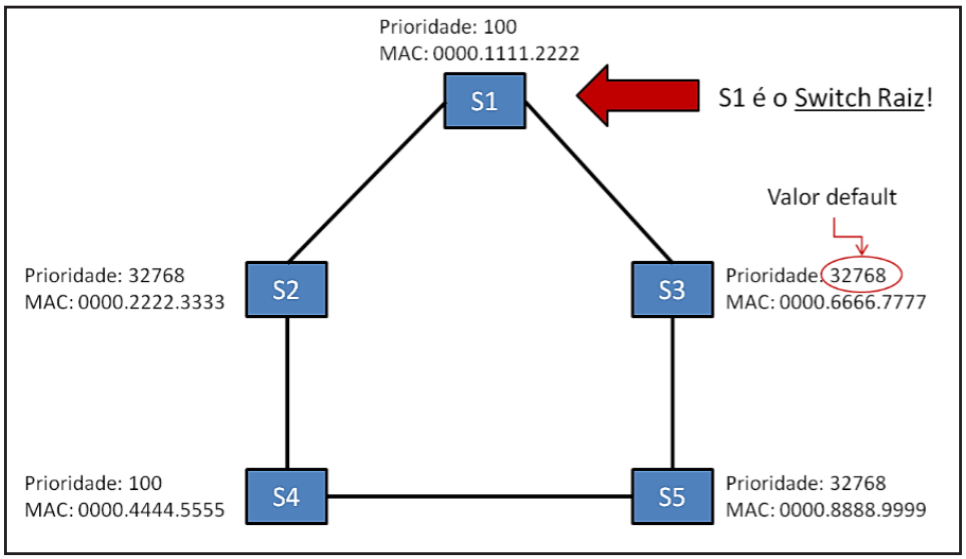
O STP (*Spanning Tree Protocol*), cuja especificação é a 802.1D, foi criado para evitar o surgimento destas tempestades de *broadcast* que desperdiçam a largura de banda dos enlaces. As mensagens STP são denominadas BPDUs (*Bridge Protocol Data Units*). Há dois tipos de BPDUs: as BPDUs de Configuração são utilizadas para definição e manutenção de uma árvore de espalhamento conceitual, formada a partir dos *switches*; as BPDUs TCN (*Topology Change Notification*) são utilizadas para a notificação de mudanças na topologia. Todas as BPDUs são enviadas para o endereço MAC *multicast* 0180.C200.0000 – esta é uma notação alternativa (um pouco mais resumida) à notação hexadecimal separada por dois pontos. A Figura 177 ilustra os principais campos de uma BPDUs e as questões às quais cada um deles procura responder.

Figura 177: Principais campos de uma BPDU



Cada *switch* possui um identificador denominado BID (*Bridge* ID). O BID é composto por dois campos: uma prioridade de 16 bits e um endereço MAC de 48 bits. Diferentemente do endereço MAC, a prioridade é ajustável – seu valor padrão é 32.768. A Figura 178 ilustra a mesma topologia ilustrada pela Figura 176, porém exibindo os BIDs de cada *switch*.

Figura 178: Bridge IDs dos switches



O processo de construção da árvore de espalhamento passa pela escolha de sua raiz. O *switch* raiz é aquele que possui o menor BID – seria, digamos, o *switch* mais antigo. Cada *switch* transmite uma BPDU de Configuração que informa aos demais qual é o seu BID – trata-se do campo BID,

ilustrado pela Figura 177. Assim, cada *switch* passa a conhecer os BIDs de todos os demais. A comparação entre os BIDs é feita inicialmente levando-se em conta os valores do campo prioridade. Na Figura 178, por exemplo, os switches S1 e S4 possuem os menores valores de prioridade (100). Havendo dois ou mais *switches* empatados com o mesmo valor de prioridade, como é o caso, o *switch* raiz é aquele que possui o menor endereço MAC – isto é, o endereço MAC define qual é o menor BID. Na Figura 178, S1 possui o menor endereço MAC, quando comparado a S4. Portanto, S1 é o *switch* raiz.

Na árvore de espalhamento definida pelo STP, cada porta de um *switch* desempenha um dos seguintes papéis: porta raiz, porta designada e porta bloqueada. Estes papéis são associados à construção e manutenção da árvore. Inicialmente, é necessário definir qual das portas de cada switch terá o papel de porta raiz. A porta raiz é aquela que possui o menor custo cumulativo até o switch raiz. O custo de uma porta é inversamente proporcional à sua largura de banda, conforme ilustrado pela Tabela 8.

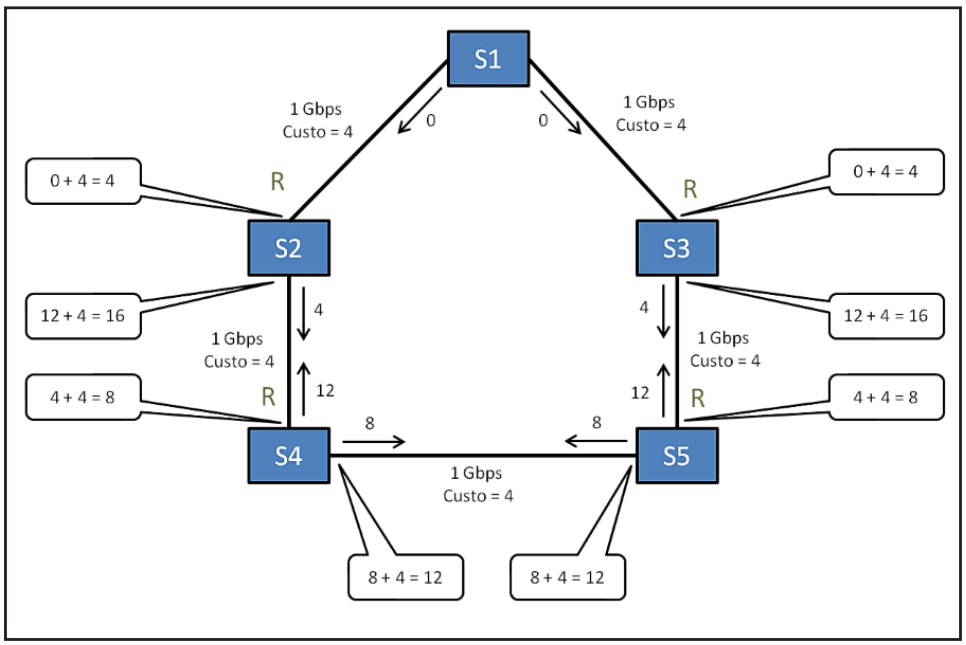
Tabela 8: Custo de uma porta em função da sua largura de banda

Largura de banda	Custo
4 Mbps	250
10 Mbps	100
16 Mbps	62
45 Mbps	39
100 Mbps	19
155 Mbps	14
1 Gbps	4
10 Gbps	2

Na Figura 179, o *switch* raiz transmite BPDUs cujo valor do campo *Root Path Cost* (ilustrado pela Figura 177) é 0 (zero). Nada mais natural, haja vista ser ele próprio (S1) o switch raiz. Quando S2 e S3 recebem aquela BPDU, eles adicionam aquele valor ao custo de suas próprias portas – que possuem, portanto, um custo cumulativo até o *switch* raiz igual a 4 (quatro). Eles então enviam BPDUs cujo valor do campo *Root Path Cost* é igual 4 (quatro). Quando as BPDUs chegam a S4 e S5, respectivamente, eles adicionam aquele valor ao custo de suas próprias portas – que possuem, portanto, um custo cumulativo até o *switch* raiz igual a 8 (oito).

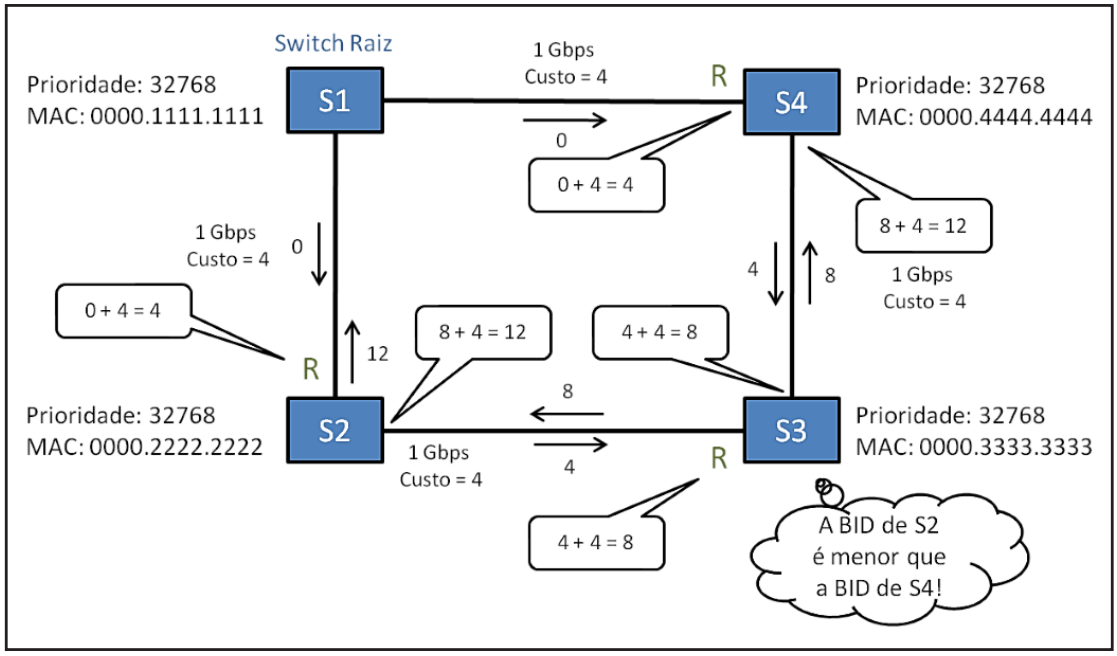
Agora, S4 e S5 enviam um ao outro BPDUs cujo valor do campo *Root Path Cost* é igual 8 (oito). Eles adicionam aquele valor ao custo de suas próprias portas – que possuem, portanto, um custo cumulativo até o *switch* raiz igual a 12. Finalmente, S4 e S5 enviam BPDUs cujo valor do campo *Root Path Cost* é igual a 12. Quando as BPDUs chegam a S2 e S3, respectivamente, eles adicionam aquele valor ao custo de suas próprias portas – que possuem, portanto, um custo cumulativo até o *switch* raiz igual a 16. Ao final, cada *switch* examina o custo cumulativo ao *switch* raiz de todas as suas portas; para cada *switch* a porta raiz será, conforme dito há pouco, a que possuir o menor valor para aquele custo cumulativo. Na Figura 179, aquelas portas são marcadas com a letra "R".

Figura 179: Definição da porta raiz de cada *switch*



A Figura 180 ilustra um cenário no qual, por conta da topologia, o *switch* S3 possui suas portas com o mesmo custo cumulativo (oito) até o *switch* raiz, S1. Em casos como este, o critério de desempate recai sobre o menor BID adjacente. Na figura, S3 possui como adjacentes S2 e S4; dentre estes, S2 possui o menor BID. Portanto, a porta raiz de S3 é definida como aquela na qual ele se conecta a S2.

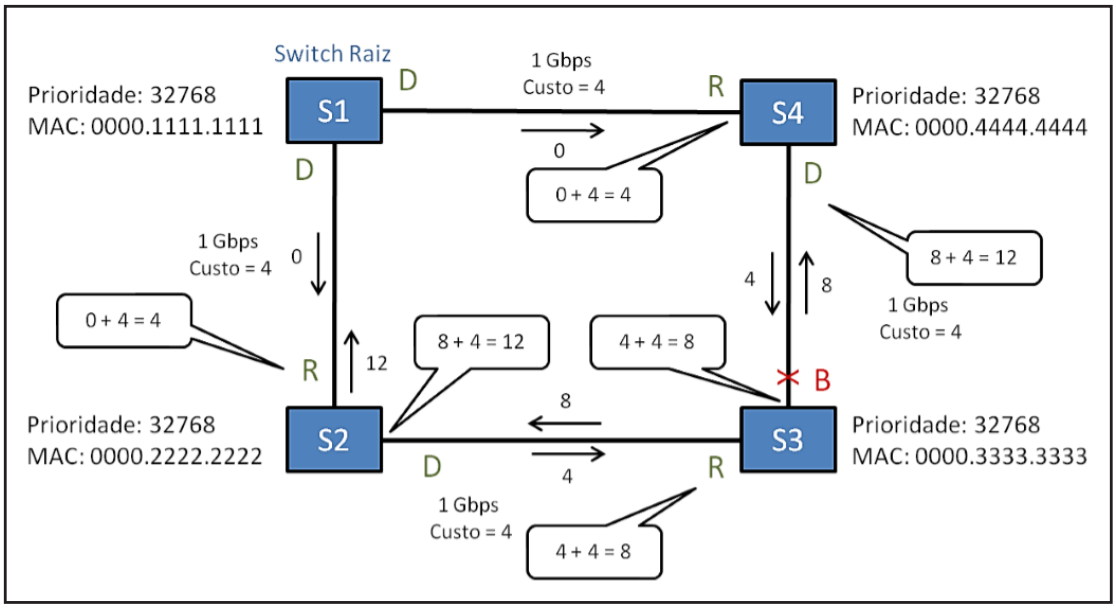
Figura 180: Definição da porta raiz de S3 com base no menor BID adjacente



O passo final na construção da árvore de espalhamento é a definição das portas designadas e das portas bloqueadas. Cada segmento deve possuir apenas uma porta designada. Em um segmento, o *switch* cuja BPDU contiver o menor custo cumulativo até o *switch* raiz terá sua porta definida como designada; o outro *switch* conectado pelo segmento terá sua porta definida como bloqueada. Na Figura 181, no enlace entre S3 e S4 circulam duas BPDUs: uma delas, enviada por S3, possui custo cumulativo até o *switch* raiz igual a 8 (oito); a outra, enviada por S4, possui custo

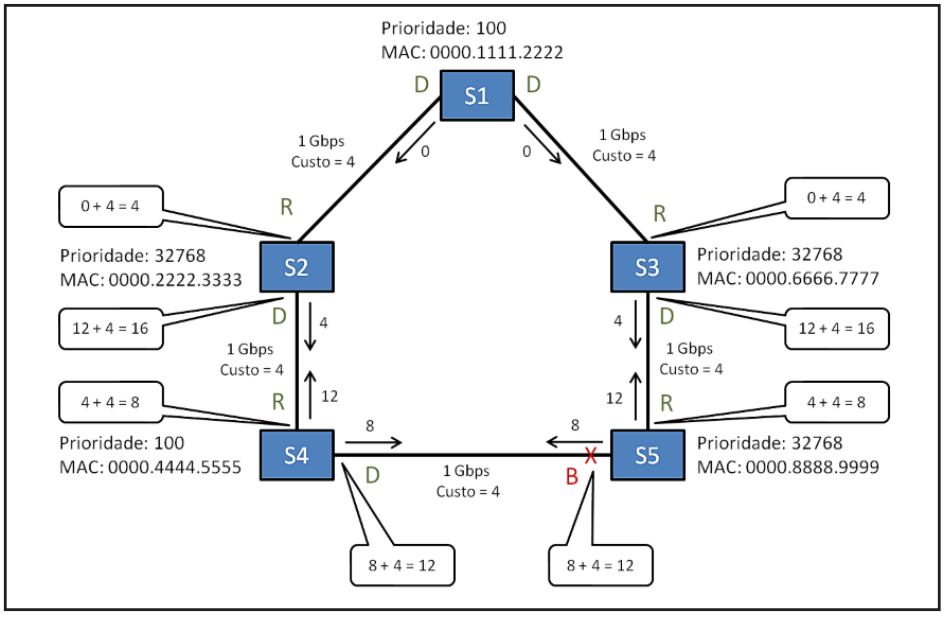
cumulativo igual a 4 (quatro). Assim, a porta de S4 associada ao segmento será definida como porta designada e a porta de S3 será definida como porta bloqueada.

Figura 181: Identificação de todas as portas na topologia apresentada na Figura 180



Havendo empate nos custos cumulativos, o *switch* que tiver a menor BID terá sua porta definida como porta designada, e o que tem a maior BID terá sua porta definida como porta bloqueada. Na Figura 182, por exemplo, os *switches* S4 e S5 transmitem BPDUs com custos cumulativos iguais a 8 (oito) no enlace que os conecta. Como S4 possui a menor BID dentre eles, a sua porta é definida como porta designada; a porta de S5 é definida como porta bloqueada.

Figura 182: Identificação de todas as portas na topologia apresentada na Figura 178

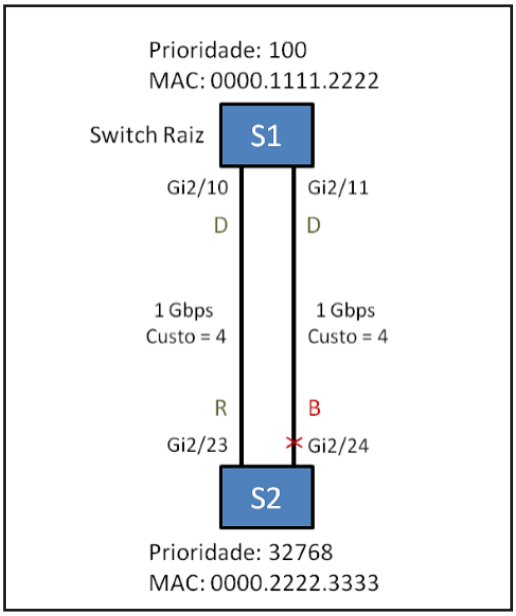


Na topologia ilustrada pela Figura 183, o *switch* raiz é S1 – eleito por possuir o menor valor do campo prioridade do BID. Para explicar qual das portas de S2 é definida como a porta raiz, há de se considerar inicialmente que ambas possuem o mesmo custo cumulativo (quatro) até S1. A aplicação do critério de desempate informado anteriormente (isto é, a menor BID adjacente) é

inútil, pois o mesmo *switch* (S1) é adjacente a S2 em ambas as portas. Em situações como esta, é utilizado o critério final de desempate – o PortID, que constitui um dos campos de uma BPDU, tendo sido ilustrado, anteriormente, pela Figura 177.

O PortID é composto pela prioridade da porta (que possui tamanho igual a quatro bits e tem como valor padrão 128) e pelo número da porta (que possui tamanho igual a 12 bits). O menor PortID define a porta raiz. Porém, salienta-se que os valores de PortID analisados referem-se ao emissor das BPDUs. Na Figura 183, S2 recebe BPDUs que contêm PortIDs das portas Gi2/10 e Gi2/11 de S1. O menor deles é o da porta Gi2/10 e, portanto, a porta de S2 que se conecta a ela é quem será a porta raiz de S2 – Gi2/23.

Figura 183: Critério de desempate final: o PortID

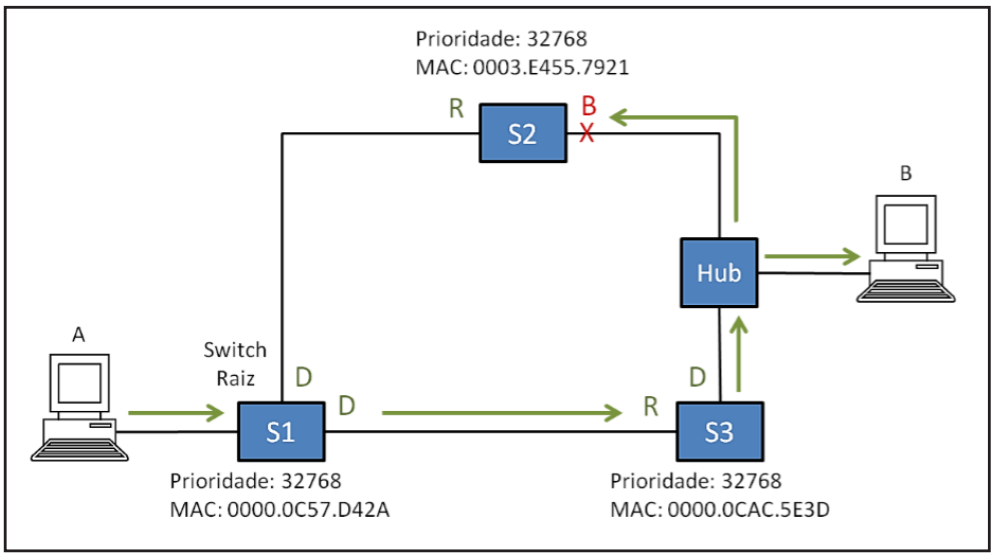


O estado de uma porta está relacionado à sua responsabilidade com relação ao encaminhamento de quadros de usuários. Uma porta desconectada está no estado *Disabled.* Quando um cabo é conectado a ela, seu estado passa a ser *Blocking*. Neste estado, ela não encaminha absolutamente nenhum quadro de usuário. Rapidamente, ela passa para o estado Listening, participando do STP e enviando e recebendo BPDUs. A porta demora 15 segundos no estado *Listening.* Uma vez definido seu papel como porta raiz ou designada, o estado da porta passa a ser *Learning.* Neste estado, ela recebe quadros, mas não os encaminha; ela apenas aprende seus endereços MAC de origem. A porta demora mais 15 segundos no estado *Learning.* Portanto, desde o instante em que um cabo é conectado a ela até o instante que quadros de usuário são encaminhados, são decorridos 30 segundos.

O STP possui três temporizadores. O Hello Time possui valor padrão de 2 (dois) segundos, e representa o intervalo de tempo entre o envio de BPDUs consecutivas por parte do *switch* raiz. Estas BPDUs são necessárias à manutenção da árvore de espalhamento definida pelo STP. Outro temporizador é o *Forward Delay*, cujo valor padrão é de 15 segundos, e que representa o tempo gasto por uma porta nos estados *Listening* e *Learning*. Por fim, o temporizador *Max Age* (cujo valor padrão é de 20 segundos) define o tempo máximo durante o qual uma BPDU recebida é armazenada. Os valores padrão apresentados podem ser alterados pelo administrador da rede, e aconselha-se que – uma vez decidido fazê-lo – seja feito no *switch* raiz.

A Figura 184 ilustra uma topologia na qual se descreve o caminho percorrido pelos quadros transmitidos de A para B. Considere, porém, que haja também um fluxo contrário – isto é, quadros transmitidos de B para A. As BPDUs de Configuração não são mostradas, apenas os papéis das portas de cada *switch*. Neste contexto, S2 possui uma de suas portas bloqueadas e está, portanto, no estado *Blocking*. Por aquela porta, S2 recebe os quadros difundidos pelo *hub*; porém, S2 não os encaminha.

Figura 184: Caminho percorrido pelos quadros transmitidos de A para B



A Figura 185 ilustra o que acontece quando a porta designada de S3 é desconectada. Espera-se que o estado da porta bloqueada de S2 evolua de *Blocking* para *Forwarding*, e que aquela porta assuma o papel de porta designada. Este processo dura 50 segundos, divididos nos seguintes períodos:

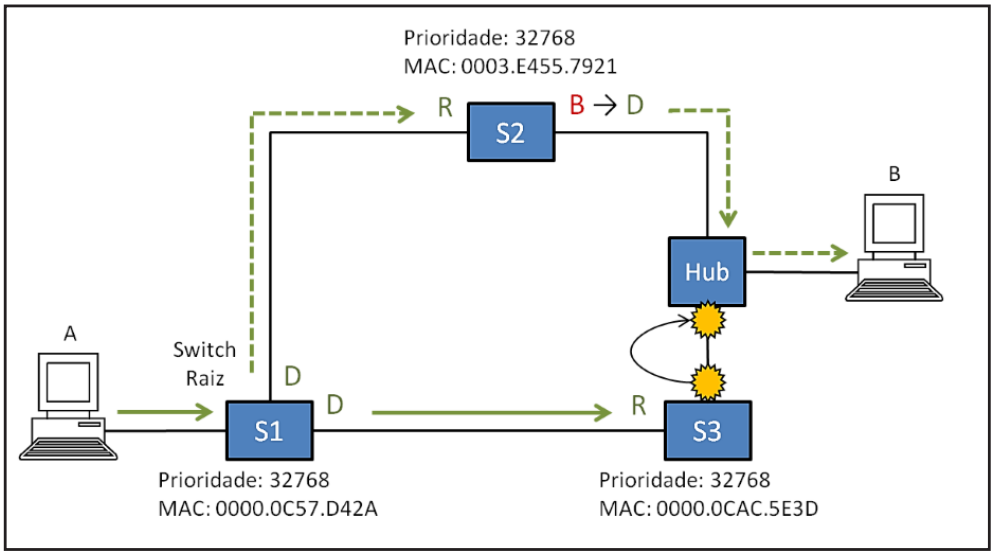
• Durante os 20 segundos iniciais, S2 armazena a última BPDU enviada por S3 – que informa que S3 possui um BID maior que S2. Este intervalo de tempo corresponde ao valor padrão do temporizador *Max Age*;

• Durante os próximos 15 segundos, a porta passa a ocupar-se do STP durante o estado *Listening*, ao final do qual a porta assume o papel de porta designada. Este intervalo de tempo corresponde ao valor padrão do temporizador *Forward Delay*;

• Durante os 15 segundos finais, a porta aprende endereços MAC de origem durante o estado Learning. Este intervalo de tempo corresponde ao valor padrão do temporizador *Forward Delay*. Ao final destes 15 segundos, a porta entra no estado *Forwarding*.

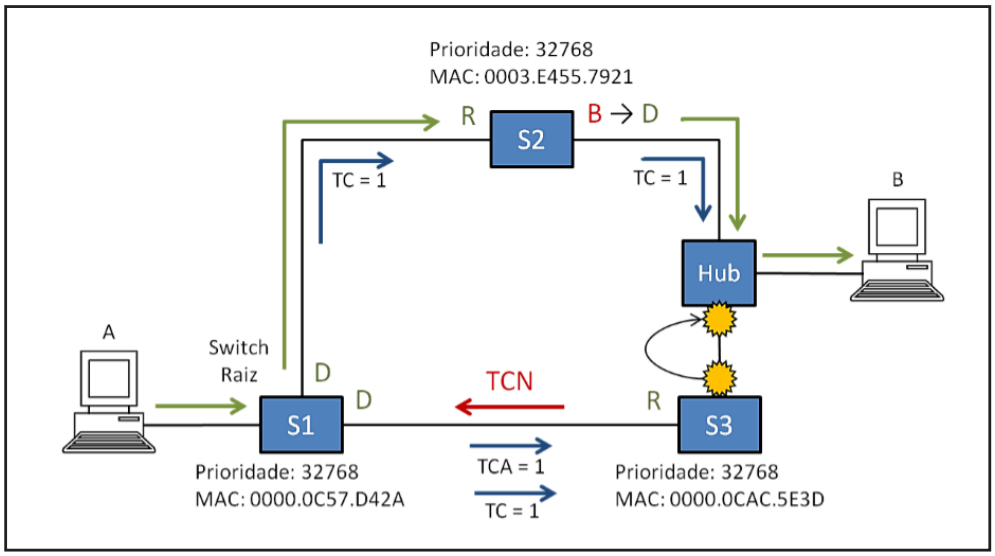
Ao final da transição de estado descrita, porém, os quadros permaneceriam sendo encaminhados por S1 para S3 por mais 250 segundos. E por quê? Ora, porque a entrada na tabela de comutação de S1 que associa o endereço MAC de B ao encaminhamento para S3 permanece lá por cinco minutos – ou seja, 300 segundos.

Figura 185: Desconexão da porta designada de S3



É neste contexto que a BPDU TCN (*Topology Change Notification*) é transmitida. Esta BPDU é enviada por um *switch* através da sua porta raiz ao perceber uma alteração na topologia. A BPDU TCN é propagada no sentido de alcançar o *switch* raiz – ela é reconhecida ao longo de cada enlace por uma BPDU de Configuração com a flag TCA (*Topology Change Acknowledgement*) levantada. Quando o *switch* raiz recebe a BPDU TCN, ele passa a enviar BPDUs de Configuração com a flag TC (*Topology Change*) levantada. Quando os demais *switches* recebem esta BPDUs, eles diminuem o tempo de permanência nas suas respectivas tabelas de comutação para 15 segundos. A Figura 186 ilustra a transmissão da BPDU TCN, seu reconhecimento por uma BPDU de Configuração com a flag TCA levantada e o envio de uma BPDU de Configuração com a flag TC levantada. O efeito do envio da BPDU TCN na figura é redirecionar o tráfego em 50 segundos, não mais os 300 segundos de outrora.

Figura 186: Transmissão da BPDU TCN e das BPDUs de Configuração subsequentes



Redes WiFi

A comunicação sem fio (*wireless*) possui vantagens e desvantagens em relação à comunicação cabeada. As vantagens imediatas são uma maior flexibilidade e um menor custo: computadores podem ser movidos com relativa facilidade e não é mais necessário gastar dinheiro com cabos e conectores. Porém, as desvantagens de uma comunicação sem fio dizem respeito ao aumento de restrições que já se aplicam à comunicação cabeada, bem como ao surgimento de restrições exclusivas à comunicação sem fio.

Toda comunicação sofre da perda de energia do sinal ao longo da distância que separa o transmissor do receptor. Dá-se o nome de atenuação a este fenômeno, e ele se manifesta com mais força na comunicação sem fio. Além da distância, que já contribui para seu surgimento, a atenuação acentua-se na comunicação sem fio pela transposição de obstáculos (muros e paredes, por exemplo) e mesmo pela absorção da energia por parte de moléculas de água em dias chuvosos.

Outra restrição acentuada nas comunicações sem fio é a susceptibilidade ao ruído, que é um sinal capaz de interferir na inteligibilidade daquele sinal que se pretende transmitir. O ruído pode ser originar de descargas elétricas atmosféricas, motores elétricos ou equipamentos eletrônicos – telefones celulares, por exemplo. Na comunicação cabeada, é possível isolar o sinal de ruído através de coberturas metálicas – que se comportam como uma gaiola de Faraday – ou através da utilização de cabos de fibra óptica. Estas estratégias, naturalmente, não podem ser utilizadas

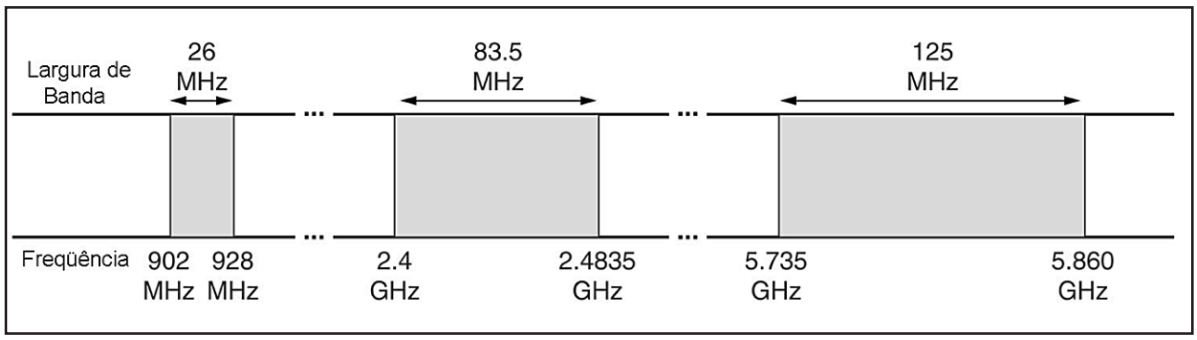
na comunicação sem fio.

Um tipo de restrição exclusiva à comunicação sem fio é a propagação multipercurso, que consiste na presença de vários modos de propagação de um sinal, ocasionados pelo fato de aquele sinal não ser guiado. A propagação multipercurso é benéfica no sentido de que permite atingir destinatários localizados em diferentes posições. Porém, ela pode fazer com que, pelo fenômeno da reflexão, vários modos convirjam para um mesmo destino e interfiram entre si negativamente, contribuindo para a ininteligibilidade do sinal recebido.

O padrão para redes locais *wireless* (WLANs – *Wireless Local Area Networks*) atende pela denominação WiFi. Trata-se da especificação IEEE 802.11. Há diferentes variantes de WiFi; todas elas possuem o mesmo protocolo de enlace, e diferenciam-se entre si apenas pelo Nível Físico empregado. Mais especificamente, a diferenciação pode ser caracterizada pela faixa de frequência utilizada e pela taxa de transmissão alcançada naquela frequência.

O WiFi utiliza faixas de frequência não licenciadas para a transmissão de dados. Também conhecidas como faixas ISM (*Industrial, Scientific and Medical*), estas faixas não são alocadas a ninguém especificamente e, por isto, qualquer pessoa pode utilizá-las em seus projetos particulares. A única recomendação é a de que haja moderação na potência do sinal transmitido, exatamente para evitar a ocorrência de interferência entre sinais que porventura utilizem a mesma frequência. A Figura 187 ilustra três faixas de frequência não licenciadas, cada qual com seu respectivo tamanho – também denominado largura de banda.

Figura 187: Faixas ISM



É fato que a capacidade máxima teórica de transmissão de um canal é diretamente proporcional à largura de banda utilizada. Há, de fato, uma equação matemática que representa esta proporcionalidade. Assim, a faixa que se inicia em 2,4 GHz possui uma capacidade máxima teórica maior que a que se inicia em 902 MHz. Pelo mesmo raciocínio, a faixa que se inicia em 5,735 GHz possui uma capacidade máxima teórica maior que a faixa que se inicia em 2,4 GHz. Conforme ilustra a Tabela 9, as variantes existentes do WiFi utilizam as faixas iniciadas em 2,4 GHz (802.11b e 802.11g) e 5,735 GHz (802.11a). Porém, trata-se da capacidade máxima teórica. Os valores reais dependem da estratégia de transmissão – também conhecida como modulação do sinal. É por isto que na Tabela 9 as variantes 802.11b e 802.11g possuem taxas de transmissão diferentes, apesar de utilizarem a mesma faixa de frequência.

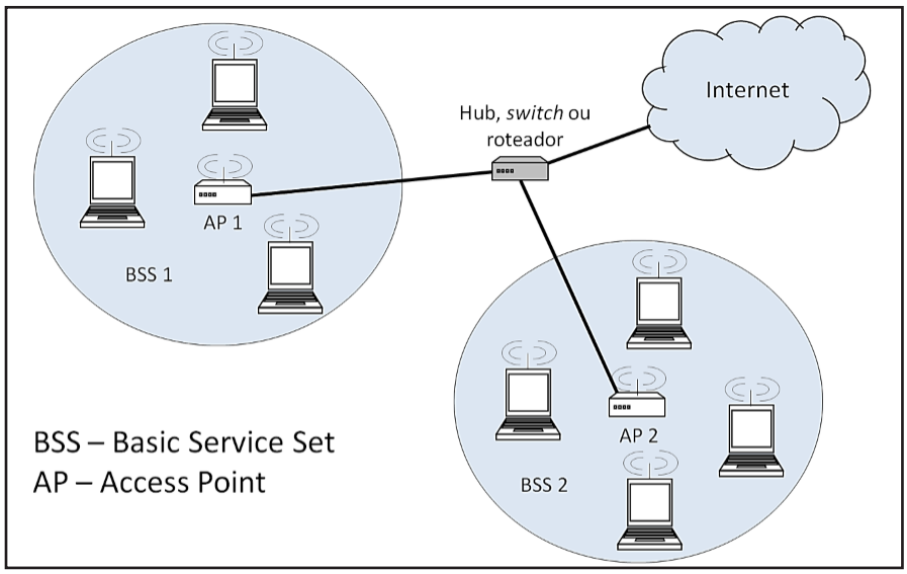
Neste contexto, a história relata que o surgimento do 802.11g deu-se graça ao apelo para se conseguir transmitir na faixa de frequência iniciada em 2,4 GHz (assim como a 802.11b) tendo a mesma taxa de transmissão conseguida pelo 802.11a. A transmissão na faixa de frequência iniciada em 2,4 GHz era preferida por custar menos que a transmissão na faixa de frequência iniciada em 5,735 GHz. A despeito da faixa de frequência utilizada e da taxa de transmissão máxima obtida, todas as variantes do WiFi utilizam antenas omnidirecionais – isto é, que irradiam em todas as direções.

Tabela 9: Variantes do WiFi

Variante	Faixa de freqüência	Taxa de transmissão
802.11b	2,4 – 2,4835 GHz	até 11 Mbps
802.11a	5,735 – 5,860 GHz	até 54 Mbps
802.11g	2,4 – 2,4835 GHz	até 54 Mbps

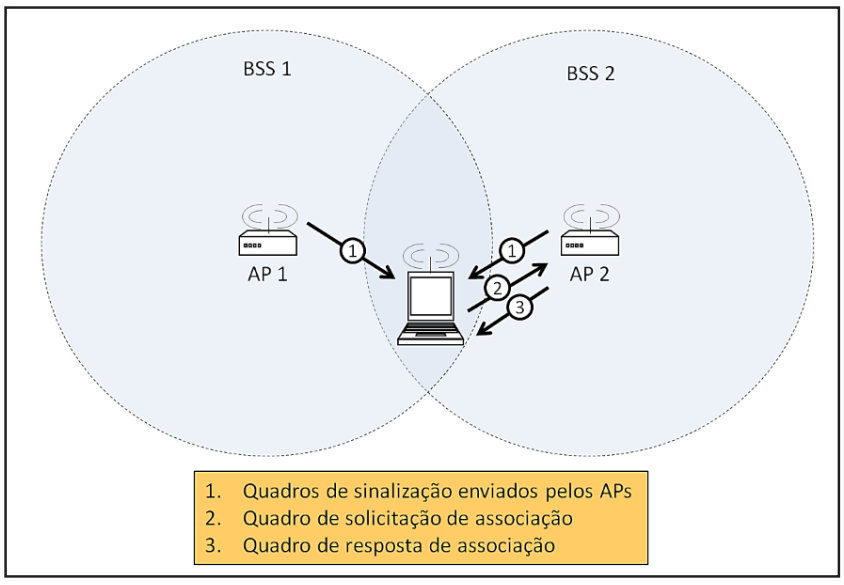
A Figura 188 ilustra a arquitetura mais comum para redes WiFi, comumente referenciada como WiFi estruturada. Ela pressupõe a presença de um AP (*Access Point*) que ficaria responsável pela comunicação entre *hosts* associados a uma BSS – *Basic Service Set*. Há também a arquitetura *ad-hoc*, muito menos utilizada e caracterizada pela ausência de um AP.

Figura 188: WiFi estruturada



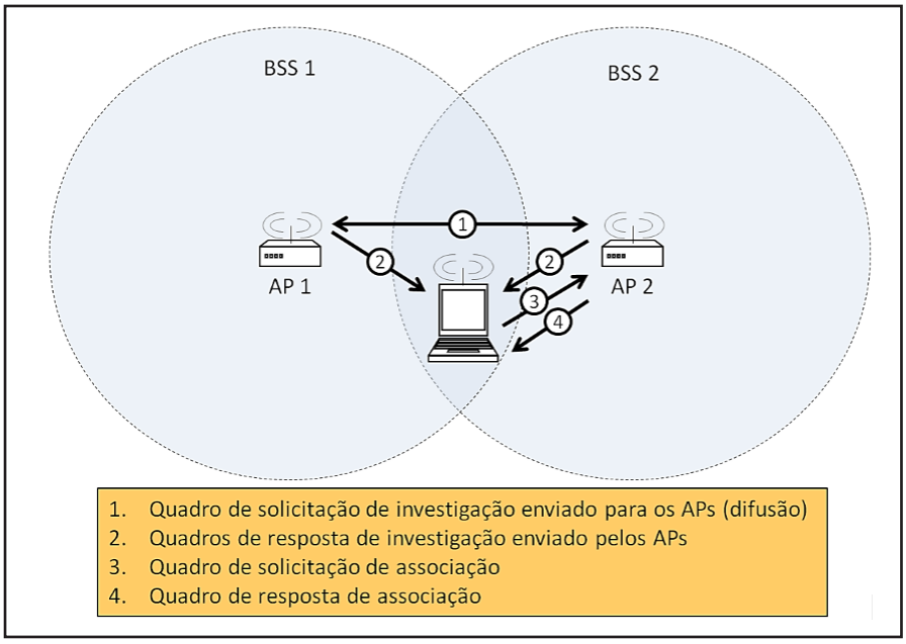
No processo de associação a um determinado AP, é comum que um *host* receba quadros de sinalização de vários APs, os quais carregam o SSID (*Service Set IDentifier*) da rede, bem como o endereço MAC do AP. Na técnica de varredura passiva, ilustrada pela Figura 189, o host recebe passivamente os quadros de sinalização enviados periodicamente pelos APs, escolhe a qual deles quer se associar e responde àquele AP.

Figura 189: Processo de varredura passiva



Uma técnica alternativa é a varredura ativa, ilustrada pela Figura 190. Nela, o *host* ativamente envia quadros de investigação imediatamente respondidos pelos quadros de sinalização dos APs. O *host* escolhe a qual dos APs deseja se associar e responde ao seu respectivo quadro de sinalização. A varredura ativa é, em média, mais rápida que a varredura passiva porque os quadros de sinalização dos APs são enviados mais rapidamente – não há necessidade de se esperar pela periodicidade dos APs para enviá-los.

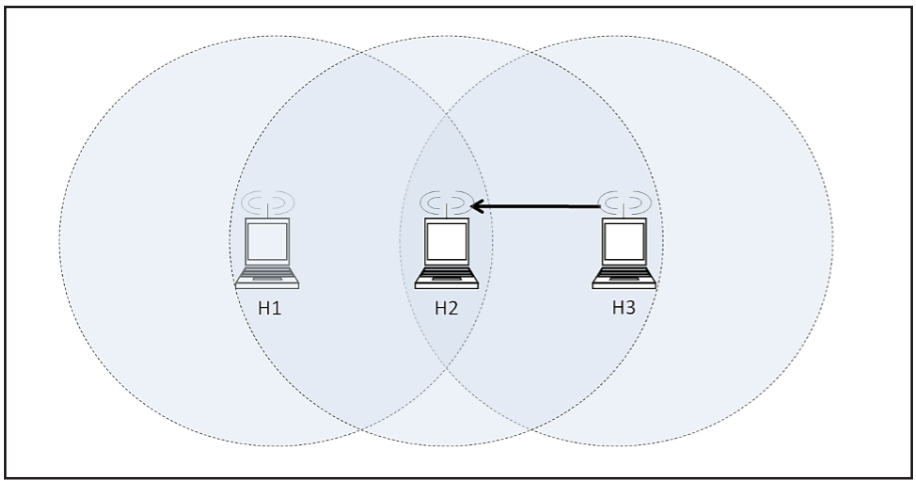
Figura 190: Processo de varredura ativa



O processo de associação a um dos APs é, por vezes, intermediado pelo usuário e pode requerer autenticação na forma de um endereço MAC (do host) ou, mais comumente, na forma de uma senha.

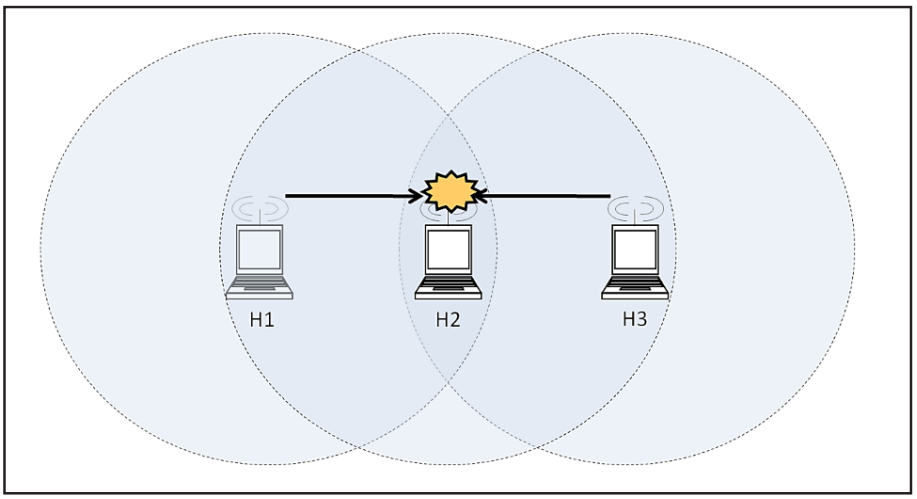
O protocolo utilizado em redes *Ethernet* é, sem dúvida, inadequado à utilização em ambientes sem fio. De cara, trata-se de um ambiente altamente susceptível a erros; portanto, reconhecimento e retransmissões são desejáveis. Além disso, outras particularidades são observadas. A Figura 191 ilustra três *hosts* (H1, H2 e H3) e seus respectivos raios de alcance. Nela, H3 está transmitindo quadros para H2. Esta transmissão não é percebida por H1; do ponto de vista de H1, H3 está oculta.

Figura 191: H3 está transmitindo quadros para H2, e H1 não detecta esta transmissão



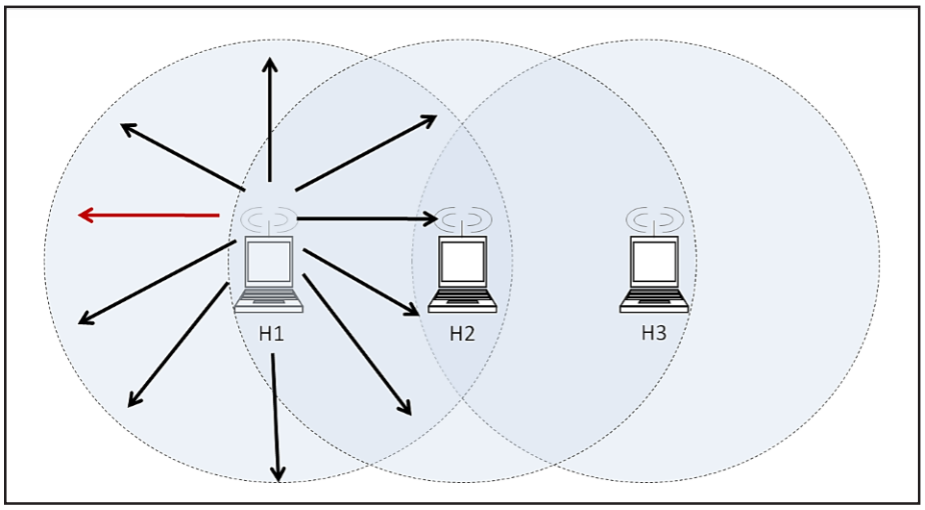
Na Figura 192, por não detectar a transmissão de H3 a H2, o host H1 transmite seus quadros para H2. Estes quadros colidem com os quadros enviados por H3 a H2, e H1 não detecta esta colisão. Assim, nem todas as colisões são detectáveis em ambientes sem fio, e o fenômeno relatado é denominado problema da estação oculta.

Figura 192: Quadros transmitidos por H1 colidem com aqueles transmitidos por H3



Outro fenômeno observável em ambientes sem fio é ilustrado pela Figura 193. Nela, o *host* H1 está transmitindo quadros para um destinatário não mostrado na figura. Por transmiti-los, omnidirecionalmente, estes quadros chegam a H2. Este host, por sua vez, deseja transmitir quadros para H3; porém, não o faz achando que aqueles quadros colidiriam com os quadros recebidos por H1. Assim, em ambientes sem fio algumas transmissões são abordadas desnecessariamente, e o fenômeno relato é denominado problema da estação exposta.

Figura 193: H2 deseja transmitir a H3, mas não o faz achando que haveria uma colisão



Pelos motivos mencionados até então, o CSMA/CD é inadequado à operação em ambientes sem fio. Há dois modos de operação para o WiFi: no modo PCF (*Point Coordination Function*), o AP controla via *polling* o acesso em um BSS; todos os host aguardam pacientemente que o AP permita a eles transmitir seus quadros – se houver quadros a serem transmitidos, naturalmente. Embora de concepção inicial simples, alguns parâmetros de funcionamento de modo de operação não foram devidamente padronizados, e isto desencorajou sua adoção. Assim, o modo de operação utilizado na prática é o DCF (Distributed Coordination Function). Nele, *hosts* e AP competem pelo acesso ao meio de transmissão utilizando o protocolo CSMA/CA (*Carrier Sense with Collision Avoidance*).

A Figura 194 ilustra o funcionamento detalhado do CSMA/CA em termos algorítmicos. Este protocolo tem sua semelhança com o CSMA/CD no sentido de que ele se baseia no fato de que um *host* (ou o AP) tem que ouvir o meio de transmissão e detectar se ele está livre antes de transmitir um quadro. Porém, conforme indica seu próprio nome, o CSMA/CA tenta evitar a ocorrência de colisões.

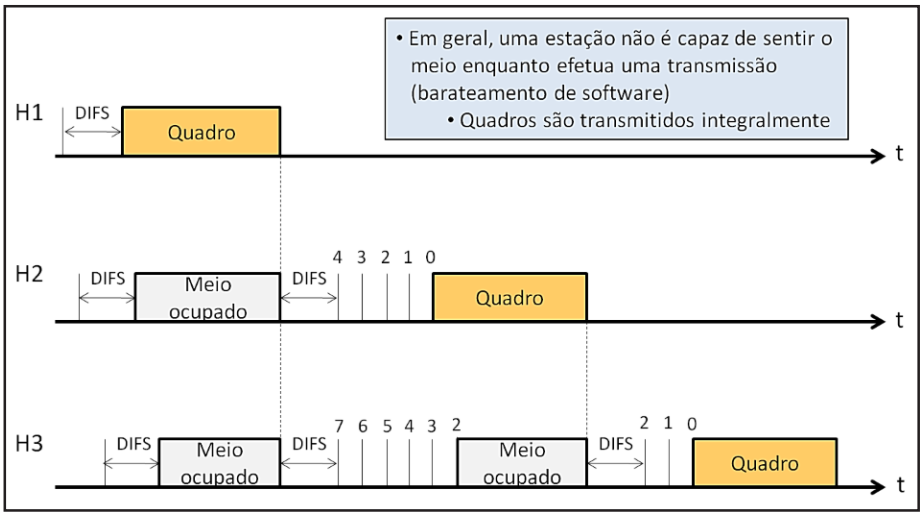
Esta tentativa se dá pela escolha antecipada de uma quantidade aleatória de *slots* a esperar quando o meio está ocupado. A Figura 195 provê um exemplo prático de execução daquele protocolo. Em momentos ligeiramente distintos, três *hosts* (H1, H2 e H3) desejam transmitir um quadro e ouvem o meio de transmissão durante um intervalo denominado DIFS (*Distributed InterFrame Space*). Ao final deste intervalo, H1 encontra o meio de transmissão livre e transmite seu quadro. H2 e H3, por sua vez, percebem que o meio de transmissão está ocupado e aguardam até que o quadro de H1 seja transmitido integralmente. Ao final deste processo, H2 e H3 ouvem o meio de transmissão por mais um DIFS e escolhem aleatoriamente, cada um, uma quantidade de *slots* de tempo.

Durante o transcorrer daqueles *slots*, H2 e H3 ouvem o meio de transmissão. Haja vista que as quantidades selecionadas por H2 e H3 foram de 4 (quatro) e 7 (sete) *slots*, respectivamente, H2 chega ao final de sua espera antes que H3. Ao sentir que o meio de transmissão está livre, H2 transmite seu quadro. H3 detecta esta transmissão após o quinto slots de tempo e interrompe sua espera até que o quadro de H3 seja transmitido integralmente e que mais um período DIFS seja atingido. Ao final deste período, H3 retoma a sua espera. Ao final dos dois últimos *slots*, H3 detecta que o meio de transmissão está livre e, enfim, transmite seu quadro.

Figura 194: Funcionamento do CSMA/CA

```
L0  sinta o meio durante DIFS
    SE (meio livre) ENTÃO transmita quadro
    SENÃO
L1      sinta o meio até ficar livre
        aguarde DIFS
        SE (meio livre) ENTÃO
            SE (r não foi escolhido) ENTÃO
               escolha r em [0,N]
L2          ENQUANTO (r > 0)
               sinta o meio durante 1 slot
               SE (meio livre) ENTÃO
                  r = r - 1
                  VÁ PARA L2
              SENÃO
                  VÁ PARA L1
            transmita quadro
            VÁ PARA L0
        SENÃO
            VÁ PARA L1
```

Figura 195: Exemplo de aplicação do algoritmo ilustrado pela Figura 194

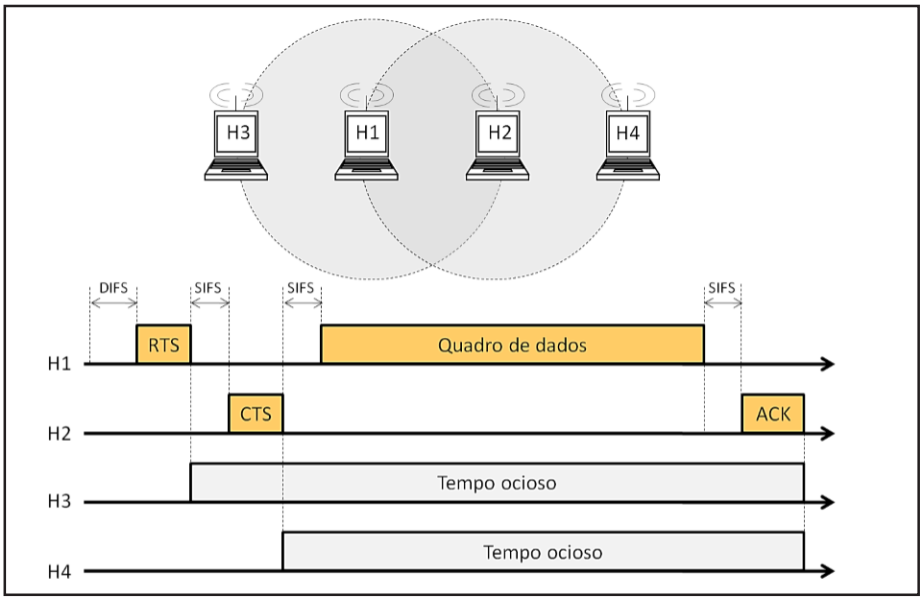


Uma estratégia alternativa utilizada, opcionalmente, pelo CSMA/CA para a prevenção de colisões é o emprego de quadros de controle. A Figura 196 ilustra um cenário no qual H1 deseja enviar um quadro para H2. H3 encontra-se no alcance de H1, e H4 encontra-se no alcance de H2. Antes de transmitir seu quadro, H1 aguarda durante DIFS, verifica que o meio está livre e envia um quadro de controle.

Este quadro não contém dados de usuário: ele informa apenas que um quadro será transmitido e o intervalo de tempo compreendido desde a sua transmissão até o seu reconhecimento. H2 e H3 recebem o quadro RTS, pois estão no alcance de H1. Durante o intervalo de tempo especificado no quadro, H3 ficará ocioso – isto é, não transmitirá nada. Após receber o quadro RTS, H2 aguarda por um intervalo de tempo denominado SIFS (*Short Interfase Space*) e o responde com o quadro de controle CTS (*Cleared To Send*).

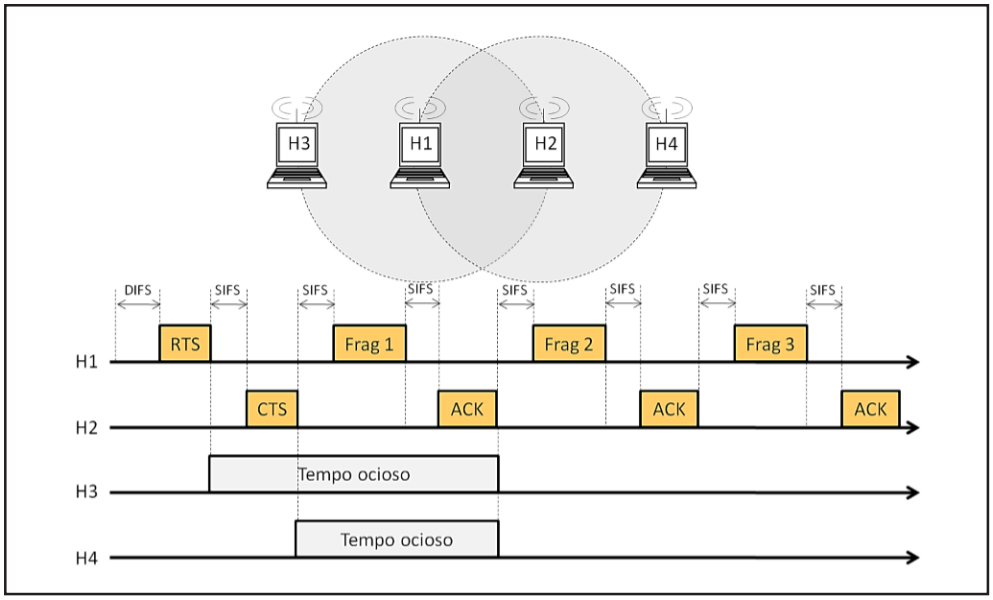
Este quadro confirma que um quadro de dados será transmitido e que informa quanto tempo será transcorrido desde o envio até o seu respectivo reconhecimento. H1 e H4 recebem o quadro CTS, pois estão no alcance de H2. Durante o intervalo de tempo especificado no quadro, H4 ficará ocioso – isto é, não transmitirá nada. H1 aguarda por mais um SIFS e transmite o quadro de dados. Após a recepção do quadro, H2 aguarda por mais um SIFS e envia um quadro de reconhecimento.

Figura 196: Utilização dos quadros de controle RTS e CTS



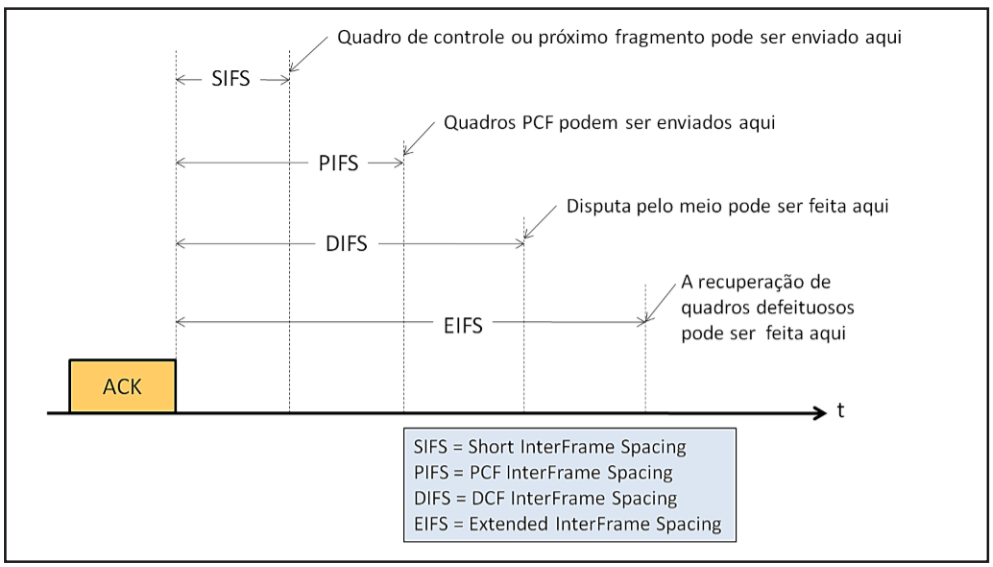
A Figura 197 ilustra a utilização dos quadros de controle RTS e CTS na transmissão de um conjunto de fragmentos – em particular, um quadro é fragmentado para que, na presença de ruído, ele não tenha que ser integralmente retransmitido. Na figura, observa-se que o tempo de ociosidade considerado por H3 e H4 está relacionado à transmissão apenas do primeiro fragmento e do seu respectivo reconhecimento. Explica-se: após o tempo de ociosidade, H3 e H4 devem esperar um tempo correspondente ao DIFS para transmitir seus quadros; antes, porém (após um tempo correspondente ao SIFS), o próximo fragmento é transmitido e as transmissões de H3 e H4 são adiadas.

Figura 197: Utilização dos quadros de controle na transmissão de um conjunto de fragmentos



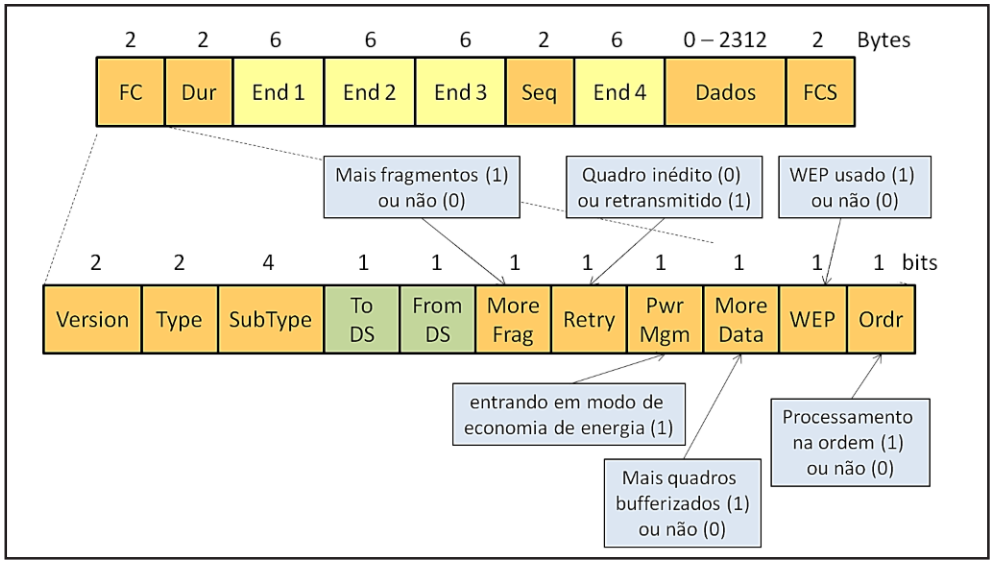
A Figura 198 ilustra todos os intervalos de tempo padrão do WiFi. Além do DIFS e do SIFS, há mais dois tempos padrão: o PIFS está relacionado ao modo de operação PCF – cuja adoção não se verifica na prática, conforme mencionado anteriormente. O EIFS destina-se à recuperação de quadros defeituosos, e é o maior dos tempos padrão.

Figura 198: Intervalos de tempo padrão do WiFi



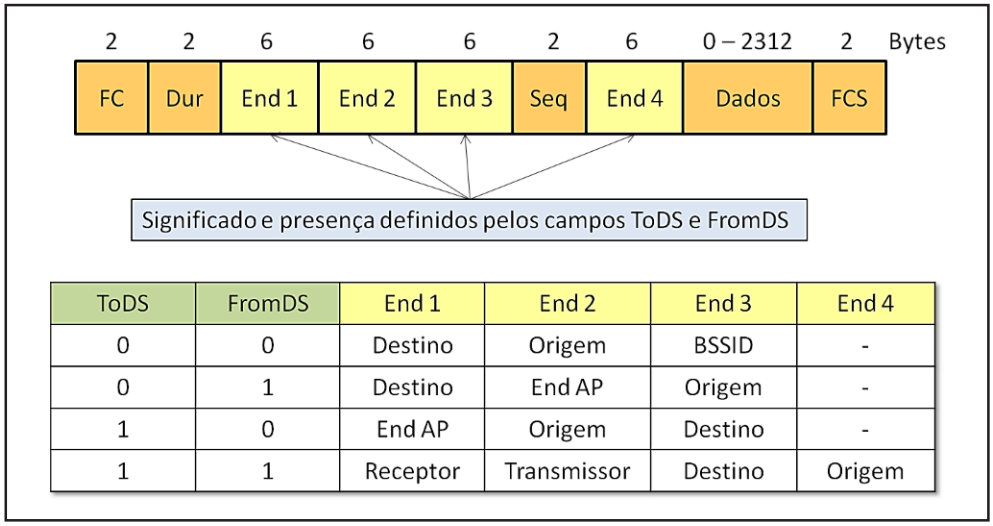
A Figura 199 ilustra o formato do quadro WiFi. Percebe-se que ele é bem mais complexo que o quadro *Ethernet*. Os dois primeiros bytes definem o campo FC (*Frame Control*), que especifica se o quadro é de controle ou de dados, se ele vai para (ou vem do) AP, se ele é o último de uma série de fragmentos, se ele é um quadro inédito ou um quadro retransmitido e se segurança está sendo utilizada, dentre outras coisas.

Figura 199: Formato do quadro WiFi



O que chama atenção de imediato na Figura 199 é o fato de o quadro WiFi possuir quatro campos de endereço, ao invés de dois. O conteúdo destes campos está intimamente relacionado às *flags* ToDS e FromDS. Neste caso, DS significa *Distribution System* – outra denominação para o AP. A Figura 200 ilustra a relação entre os valores destas flags e o conteúdo dos endereços do quadro WiFi.

Figura 200: Relação entre os valores das *flags* ToDS e FromDS
e o conteúdo dos campos de endereço do quadro WiFi

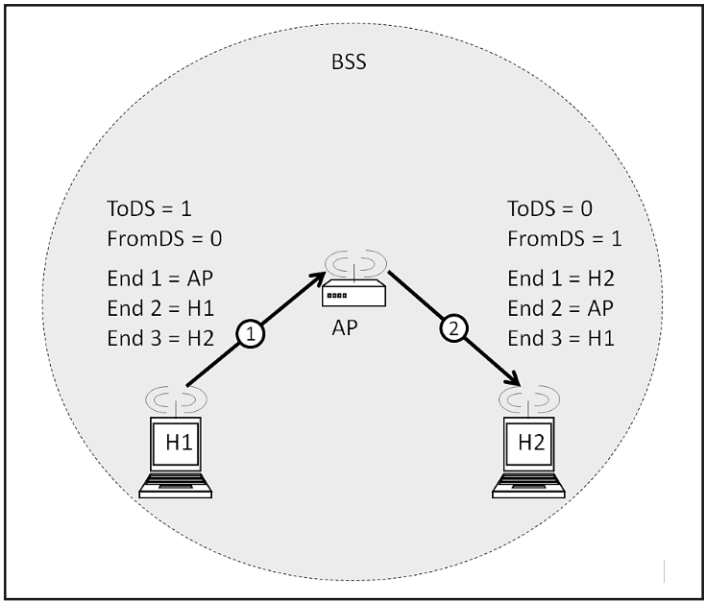


ToDS	FromDS	End 1	End 2	End 3	End 4
0	0	Destino	Origem	BSSID	-
0	1	Destino	End AP	Origem	-
1	0	End AP	Origem	Destino	-
1	1	Receptor	Transmissor	Destino	Origem

A Figura 201 ilustra o caso em que um quadro está sendo transmitido em uma WiFi estruturada. O *host* H1 deseja transmitir um quadro para H2. Este quadro possui a *flag* ToDS levantada e três endereços: o primeiro deles identifica o destinatário imediato (o AP). O segundo identifica o remetente original (H1) e o terceiro identifica o destinatário final (H2).

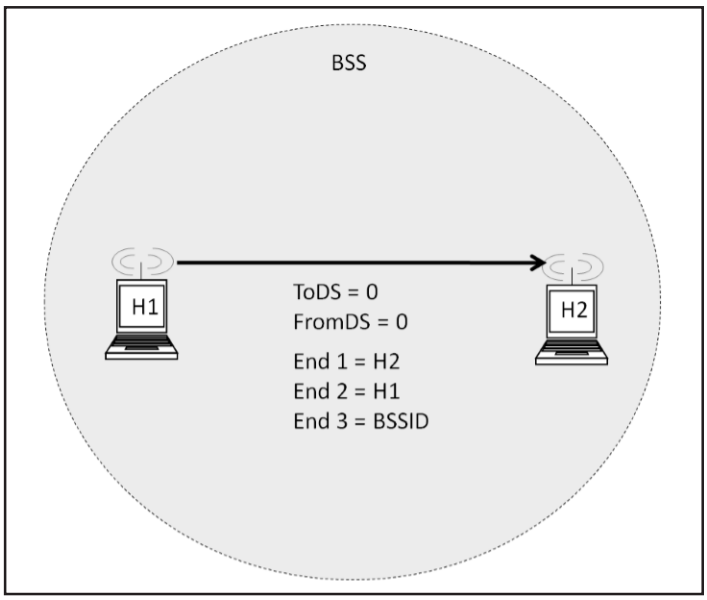
Quando chega ao AP, ele o transmite a H2. Porém, agora é a *flag* FromDS que está levantada. O valor dos endereços também mudou; o primeiro deles identifica o destinatário final (H2), o segundo identifica o remetente imediato (o AP) e o terceiro identifica o remetente original (H1).

Figura 201: Valores dos campos de endereço no quadro transmitido em uma WiFi estruturada



A Figura 202 ilustra os valores dos campos de endereço em uma WiFi *Ad-hoc*. Conforme se observa, nenhuma das *flags* está levantada (afinal de contas, não há nenhum AP). O primeiro endereço identifica o destinatário, o segundo identifica o remetente e o terceiro identifica o BSSID da rede.

Figura 202: Valores dos campos de endereço em um quadro transmitido em uma WiFi *Ad-hoc*

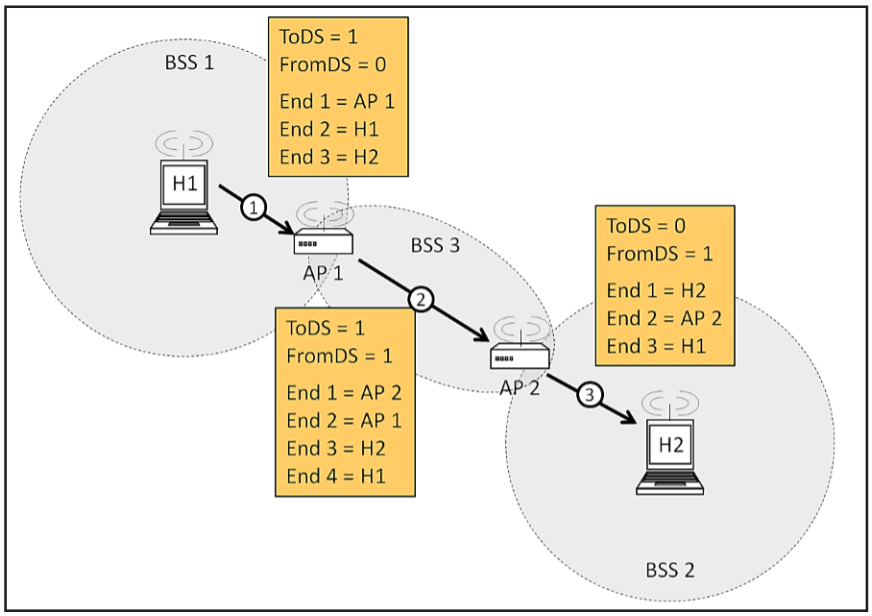


A Figura 203 ilustra o único caso em que os quatro endereços estão presentes. O remetente original (H1) e o destinatário final (H2) estão em BSSs diferentes. Quando H1 envia o quadro, a *flag* ToDS está levantada. O primeiro endereço identifica o destinatário imediato (o AP 1), o segundo endereço identifica o remetente original (H1) e o destinatário final (H2). Quando o AP 1 transmite o quadro para o AP 2, ambas as flags estão levantadas (afinal de contas, o quadro vem de um AP e vai para um AP).

O primeiro endereço identifica o destinatário imediato (o AP 2), o segundo endereço identifica o remetente imediato (o AP 1), o terceiro endereço identifica o destinatário final (H2) e o

quarto endereço identifica o remetente original (H1). Quando o AP 2 transmite o quadro para H2, a flag FromDS está levantada. O primeiro endereço identifica o destinatário final (H2), o segundo endereço identifica o remetente imediato (o AP 2) e o terceiro endereço identifica o remetente original (H1).

Figura 203: Valores dos campos de endereço quando remetente e destinatário estão em BSSs diferentes

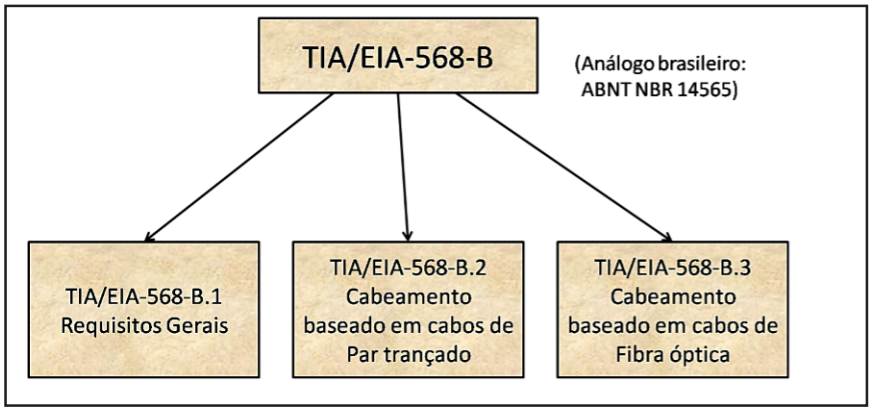


Noções de Cabeamento Estruturado

Quando uma infraestrutura de rede de um prédio faz uso de meios de transmissão guiados, é necessário obedecer às normas de cabeamento estruturado. A Figura 204 ilustra a especificação TIA/EIA-568-B, criada pela TIA (*Telecommunications Industry Association*) e pela EIA (*Eletronic Industries Association*). Esta especificação possui uma análoga nacional, a NBR 14.565 da ABNT (Associação Brasileira de Normas Técnicas).

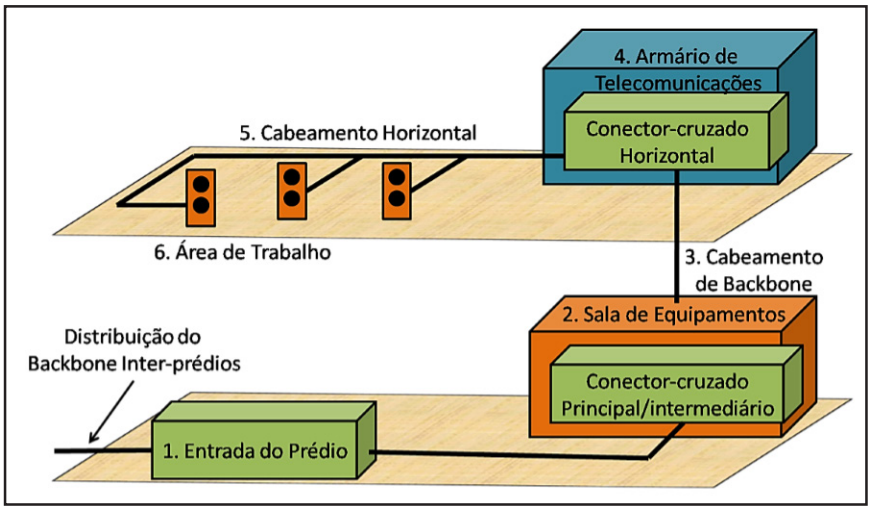
A TIA/EIA-568-B é, na verdade, uma especificação guarda-chuva: ela subdivide-se em três outras especificações: a TIA/EIA-568-B.1 define os requisitos gerais do cabeamento estruturado. É nesta especificação que são definidos os componentes funcionais do cabeamento estruturado, abordados a seguir. As especificações TIA/EIA-568-B.2 e TIA/EIA-568-B.3 estabelecem de fato as normas a serem seguidas para cabeamento baseado em cabos de par trançado e cabos de fibra óptica, respectivamente.

Figura 204: Normas de cabeamento estruturado



Os cinco componentes funcionais do cabeamento estruturado são: Entrada do Prédio, Sala de Equipamentos, Cabeamento de *Backbone*, Armário de Telecomunicações, Cabeamento Horizontal e Área de Trabalho. A Figura 205 ilustra a disposição física destes componentes em um prédio de dois pavimentos – para um prédio de três ou mais pavimentos, é válido considerar que os demais pavimentos possuem disposição idêntica ao do pavimento superior da Figura 205.

Figura 205: Componentes funcionais do cabeamento estruturado

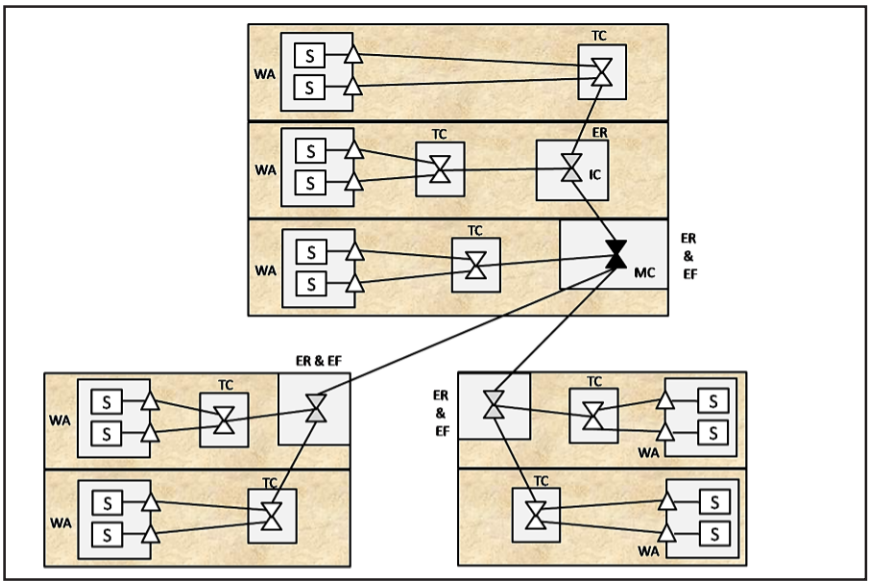


Um projeto de cabeamento estruturado define com precisão a disposição física dos componentes funcionais citados há pouco, bem como de computadores, suas respectivas tomadas e de conectores cruzados (conhecidos popularmente como *patch panels*, ou painéis de fixação). Todos estes elementos possuem representações gráficas padronizadas, ilustradas pela Figura 206. A Figura 207 ilustra um exemplo de utilização destas representações.

Figura 206: Representação gráfica padronizada dos elementos presentes em um projeto de cabeamento estruturado

- MC: Conector-cruzado Principal
- IC: Conector-cruzado Intermediário
- HC: Conector-cruzado Horizontal
- ER: Sala de Equipamentos
- TC: Armário de Telecomunicações
- EF: Entrada do Prédio
- S: Computador
- O: Tomada de Computador

Figura 207: Exemplo de projeto de cabeamento estruturado



A Entrada do Prédio (EF – *Entrance Facilities*) representa a junção entre o cabeamento externo (inter-prédios) e o cabeamento interno (inter-prédios). Ela contém *hardware* de interconexão, dispositivos de proteção e outros equipamentos, e é o ponto de demarcação entre o provedor do serviço e o cabeamento do cliente.

A Sala de Equipamentos (ER – *Equipment Room*) está localizada normalmente no pavimento térreo de um prédio. Ela possui acesso restrito e abriga servidores, roteadores, *switches*, e o conector cruzado principal (MC – *Main Cross-Connect*) e, quando necessário, o conector cruzado intermediário (IC – *Intermediate Cross-Connect*). A Sala de Equipamentos raramente experimenta mudanças estruturais.

O Cabeamento de *Backbone*, também denominado Cabeamento Vertical ou Rede Primária, é planejado para sofrer alterações somente após um período que vai de 3 (três) a 10 anos. O Cabeamento de *Backbone* pode ser formado por cabos de par trançados não blindados (UTP – *Unshielded Twisted Pair*), cabos de fibra óptica multimodo 62,5/125 µm ou 50/125 µm, ou cabos de fibra óptica monomodo. A topologia utilizada, com relação aos conectores cruzados, é em estrela; isto quer dizer que todos os conectores cruzados presentes em cada pavimento (HCs – *Horizontal Cross-Connects*) devem ser conectados diretamente ao conector cruzado principal ou intermediário e que todos os conectores cruzados intermediários devem ser conectados diretamente ao conector cruzado principal. Esta última assertiva garante que haja uma hierarquia de, no máximo, dois níveis entre os conectores cruzados. Todas estas nuances topológicas podem ser observadas no exemplo ilustrado pela Figura 207.

A distância máxima do cabeamento de *Backbone* depende do meio de transmissão utilizado. Quanto menor for a atenuação proporcionada por um determinado meio de transmissão, maior será a distância máxima do cabeamento de *Backbone* quando ele for utilizado. A Figura 208 ilustra a distância máxima definida quando cabos UTP são utilizados. Esta distância é aumentada quando cabos de fibra óptica multimodo são utilizados, conforme ilustrado pela Figura 209. Um aumento ainda mais significativo é verificado quando cabos de fibra óptica monomodo são utilizados, conforme ilustrado pela Figura 210.

Figura 208: Distância máxima para o cabeamento de *Backbone* utilizando cabos UTP

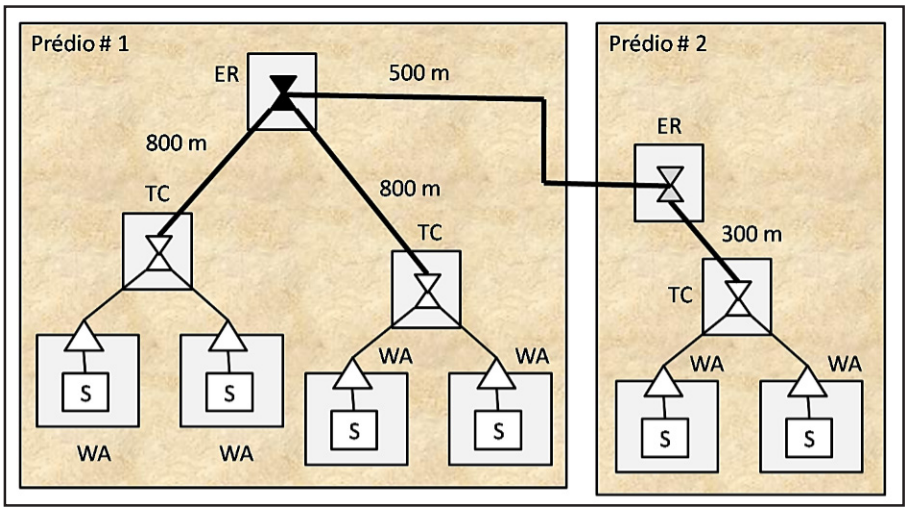


Figura 209: Distância máxima para o Cabeamento de *Backbone* utilizando cabos de fibra óptica multimodo

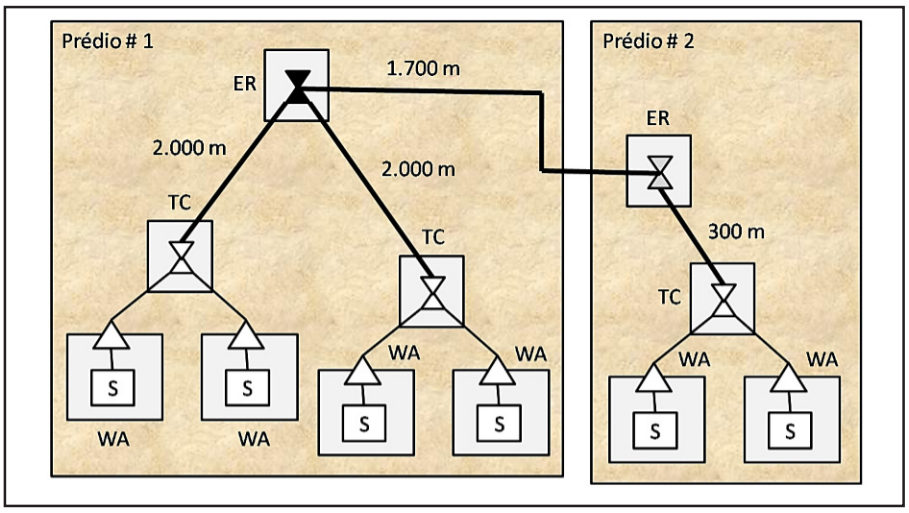
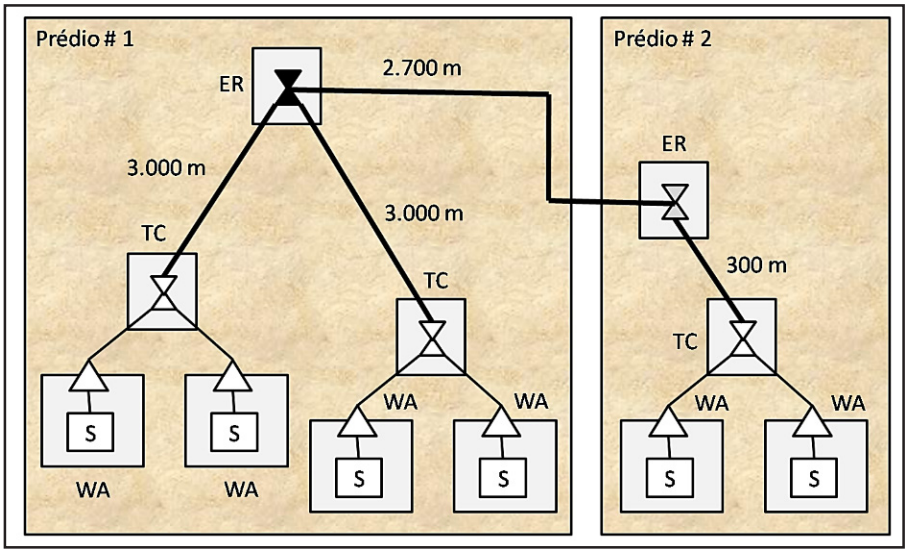


Figura 210: Distância máxima para o Cabeamento de *Backbone* utilizando cabos de fibra óptica monomodo



O Armário de Telecommunicações (TC – *Telecommunications Closet*) tem a mesma função da sala de equipamentos; porém, possui menor complexidade. Normalmente, há no mínimo um TC por pavimento em um prédio.

Os conectores cruzados funcionam como intermediários entre os equipamentos de interconexão (roteadores ou *switches*, por exemplo) e as tomadas de telecomunicação. Eles melhoram a organização dos cabos e facilitam a reconfiguração da rede, além de serem construídos para fixação em racks. Há dois tipos padronizados de conexão via conectores cruzados, denominados de conexão cruzada e interconexão, ilustrados pela Figura 211 e pela

Figura 211: Conexão cruzada

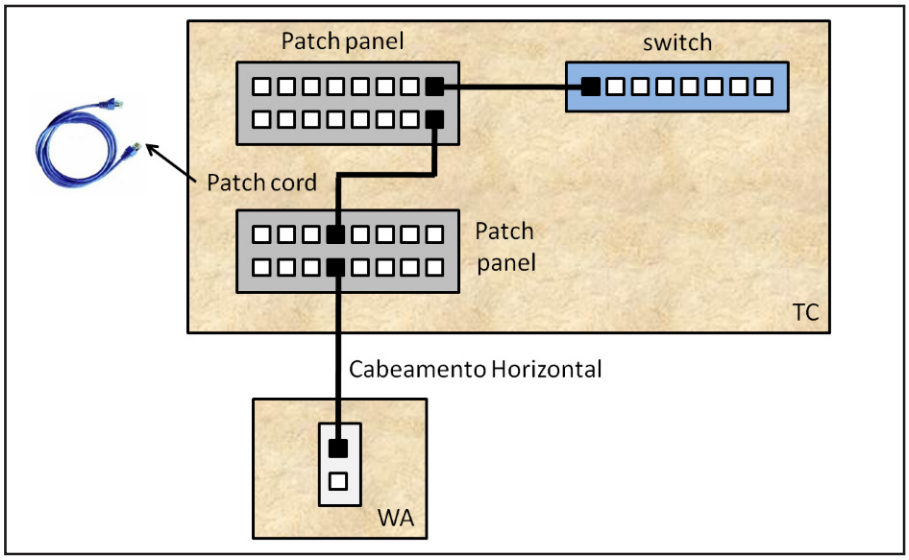
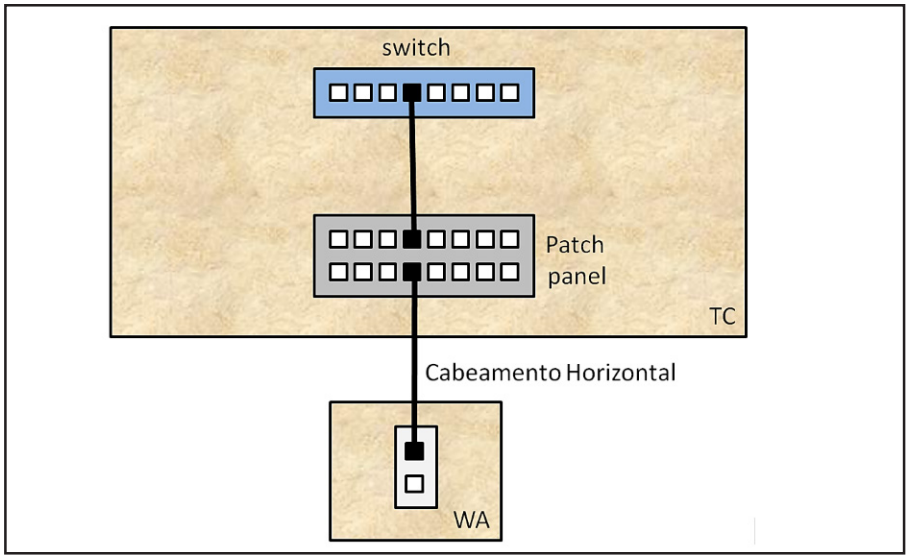
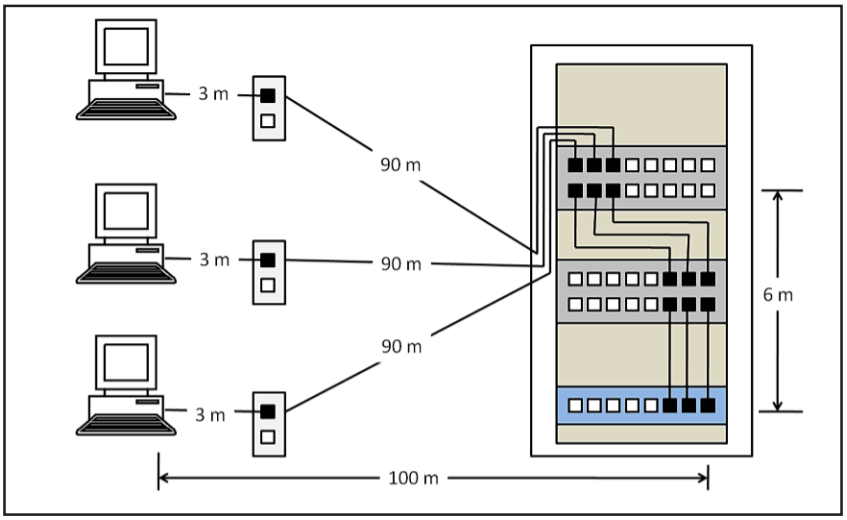


Figura 212: Interconexão



O Cabeamento Horizontal, também denominado Rede Secundária, estende-se do HC até a tomada de telecomunicações. Possui topologia em estrela, centrada no HC, e experimenta mudanças com mais frequência do que no Cabeamento de *Backbone*. Os meios de transmissão que podem utilizados são cabos de par trançado UTP e cabos de fibra óptica multimodo (62,5/125 µm ou 50/125 µm). A distância máxima é de 100 m, distribuídos de acordo com o ilustrado pela Figura 213.

Figura 213: Distâncias máximas nos segmentos que compõem o Cabeamento Horizontal



Há algumas disposições físicas específicas ao Cabeamento Horizontal. Uma delas é o MUTOA (*Multi-User Telecommunications Outlet Assembly*), ilustrada pela Figura 214. Trata-se de uma concentração de tomadas de telecomunicações (12 delas, no máximo) em colunas. A Tabela 10 ilustra as distâncias máximas em um segmento de Cabeamento Horizontal onde o MUTOA é utilizado.

Figura 214: MUTOA

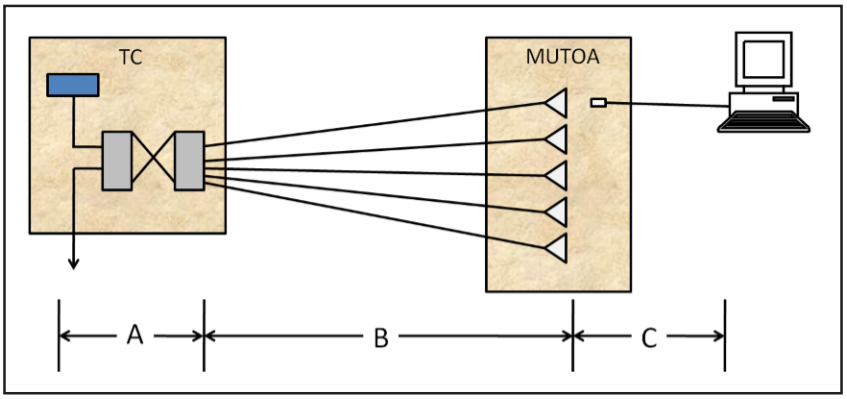
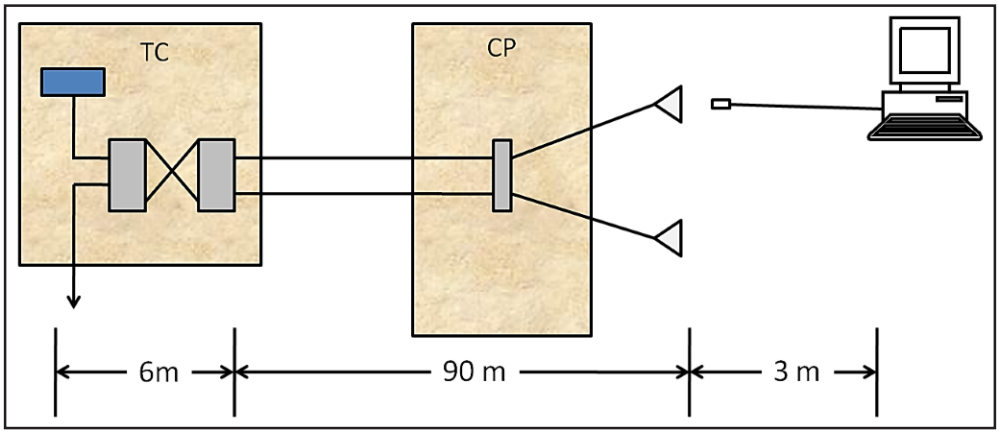


Tabela 10: Distâncias máximas em um segmento onde o MUTOA é utilizado

A	B	C	Total
5	90	5	100
5	85	9	99
5	80	13	98
5	75	17	97
5	70	22	97

Outra disposição física específica é o Ponto de Consolidação (PC – *Point of Consolidation*). Conforme ilustrado pela Figura 215, trata-se da concentração inicial do Cabeamento Horizontal no teto para, em seguida, ser distribuído.

Figura 215: Ponto de Consolidação



A Área de Trabalho (WA – *Working Area*) compreende desde a tomada de telecomunicações até o computador. No cabeamento estruturado baseado em cabos UTP, há dois padrões de crimpagem para cabos, denominados T568-A e T568-B, que são definidos pela sequência de cores dos fios a serem crimpados junto ao conector RJ-45. A Figura 216 ilustra a sequência definida pelo padrão T568-A e os sinais transmitidos/enviados em cada fio considerando a variante 10 Base-T da *Ethernet* Clássica e a variante 100 Base-TX da *Fast Ethernet.* A Figura 217 faz o mesmo com relação ao padrão T568-B.

Figura 216: Sequência de cores definida pelo padrão T568-A

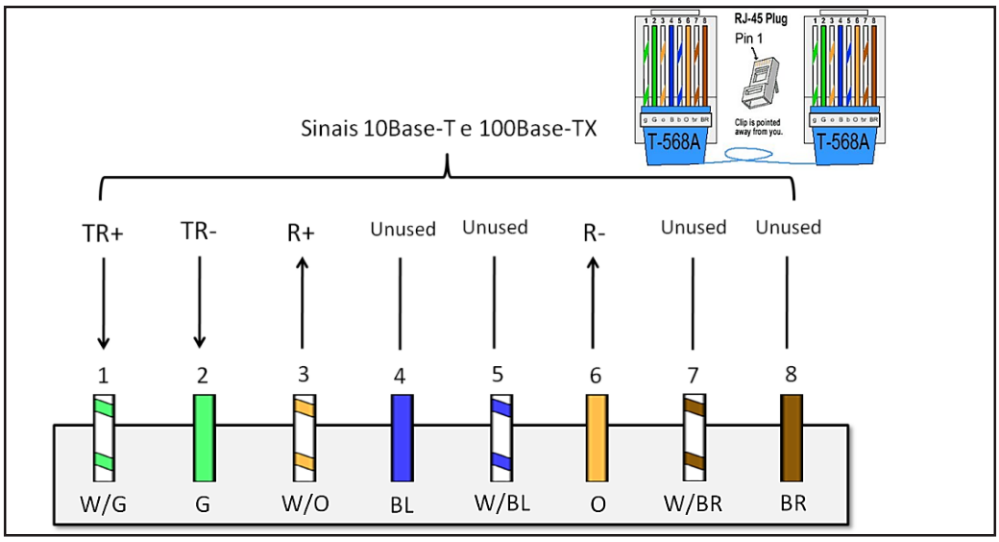
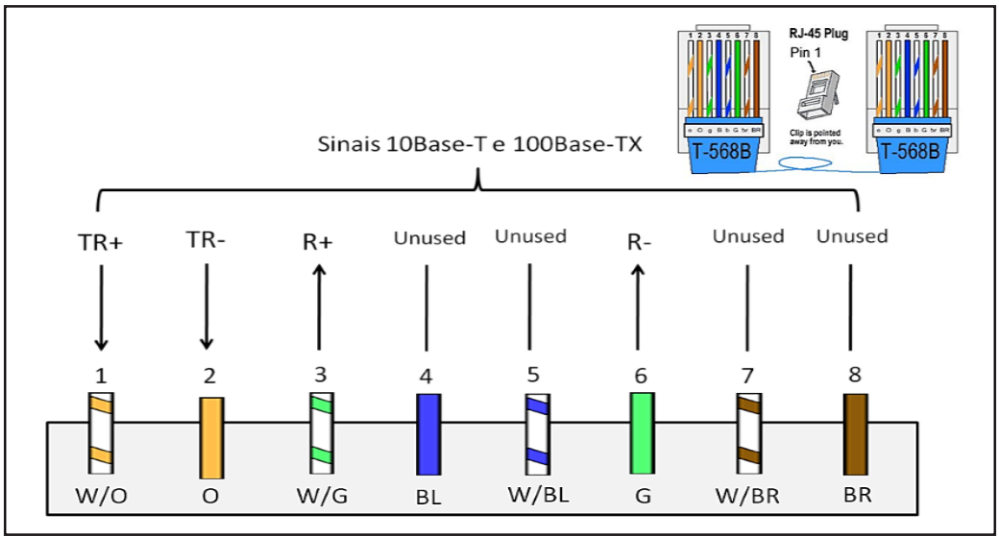
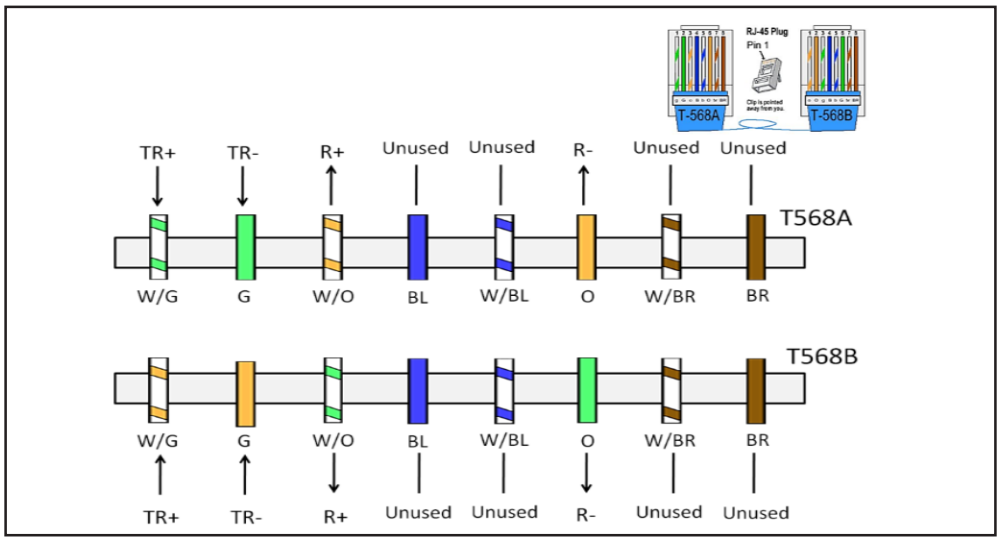


Figura 217: Sequência de cores definida pelo padrão T568-B



Cabos que são crimpados com o mesmo padrão nas duas extremidades, quer seja o T-568A ou o T-568B, são denominados cabos diretos (ou *straightforwarding*). Eles são utilizados em ligações de computadores a *switches,* ou de *switches* a roteadores. Por outro lado, cabos que são crimpados com padrões diferentes nas suas extremidades são denominados cabos cruzados (ou *crossover*). Eles são utilizados em ligações de computadores a computadores ou de *switches a switches*. A Figura 218 ilustra um cabo crimpado para ser um cabo *crossover*.

Figura 218: Cabo crossover



SAIBA MAIS

Neste caderno, apenas algumas noções acerca do cabeamento estruturado são providas. Para saber mais, consulte Lima Filho (2014).

REFERÊNCIAS

• BRITO, Samuel H. Bucke. IPv6: – o novo protocolo da Internet. Novatec. 2013.

• COMER, Douglas E. **Redes de Computadores e a Internet.** 2. ed. Porto Alegre: Bookman, 2000.

• DOYLE, Jeff. **Routing TCP/IP,** Volume I (CCIE Professional Development). Cisco Press, 1998.

• KUROSE, J. F e ROSS, Keith W. **Redes de computadores e a Internet:** uma abordagem top-down. 5. ed. São Paulo: Pearson, 2010.

• LIMA FILHO, Eduardo Corrêa. **Fundamentos de rede e cabeamento estruturado.** São Paulo: Pearson Education do Brasil, 2014.

• TANENBAUM, Andrew; WETHERALL, D. **Redes de computadores.** 5. ed. São Paulo: Pearson, 2011.

ANOTAÇÕES

ANOTAÇÕES

EDITORA

EDUFERSA - Editora da Universidade Federal Rural do Semi-Árido
Campus Leste da UFERSA
Av. Francisco Mota, 572 - Bairro Costa e Silva
Mossoró-RN | CEP: 59.625-900
edufersa@ufersa.edu.br

COMPOSIÇÃO

Formato: 21cm x 29,7cm
Número de páginas: 154

UNIVERSIDADE FEDERAL
UFERSA
RURAL DO SEMI-ÁRIDO

PROGRAD
PRÓ-REITORIA DE GRADUAÇÃO

NEaD
núcleo de educação a distância

Ministério da Educação
GOVERNO FEDERAL
BRASIL
PAÍS RICO É PAÍS SEM POBREZA

CAPES

UAB
UNIVERSIDADE ABERTA DO BRASIL